SWIPE: A SAWTOOTH WAVEFORM INSPIRED PITCH ESTIMATOR FOR SPEECH AND MUSIC

By

ARTURO CAMACHO

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2007

Dedico esta disertación a mis queridos abuelos
Hugo y Flory

ACKNOWLEDGMENTS

I thank my grandparents for all the support they have given to me during my life, my wife for her support during the years in graduate school, and my daughter who was in my arms when the most important ideas expressed here came to my mind. I also thank Dr. John Harris for his guidance during my research and for always pushing me to do more, Dr. Rahul Shrivastav for his support and for introducing me to the world of auditory system models, and Dr. Manuel Bermudez for his unconditional support all these years.

TABLE OF CONTENTS

page

CHAPTER

APPENDIX

LIST OF TABLES

LIST OF FIGURES

Figure page

LIST OF OBJECTS

LIST OF ABBREVIATIONS

AC	Autocorrelation
AMDF	Average magnitude difference function
APVD	Average peak-to-valley distance
ASDF	Average squared difference function
BAC	Biased autocorrelation
CEP	Cepstrum
ERB	Equivalent rectangular bandwidth
ERBs	Equivalent-rectangular-bandwidth scale
FFT	Fast Fourier transform
HPS	Harmonic product spectrum
HS	Harmonic sieve
IP	Inner product
IT	Integral transform
ISL	Idealized spectral lobe
K^+-NIP	K^+-normalized inner product
NIP	Normalized inner product
O-WS	Optimal window size
P2-WS	Power-of-two window size
SHS	Subharmonic-summation
SHR	Subharmonic-to-harmonic ratio
STFT	Short-time Fourier transform
SWIPE	Sawtooth Waveform Inspired Pitch Estimator
UAC	Unbiased autocorrelation
WS	Window size

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

SWIPE: A SAWTOOTH WAVEFORM INSPIRED PITCH ESTIMATOR FOR SPEECH AND MUSIC

By

Arturo Camacho

December 2007

Chair: John G. Harris
Major: Computer Engineering

A Sawtooth Waveform Inspired Pitch Estimator (SWIPE) has been developed for processing speech and music. SWIPE is shown to outperform existing algorithms on several publicly available speech/musical-instruments databases and a disordered speech database. SWIPE estimates the pitch as the fundamental frequency of the sawtooth waveform whose spectrum best matches the spectrum of the input signal. A decaying cosine kernel provides an extension to older frequency-based, sieve-type estimation algorithms by providing smooth peaks with decaying amplitudes to correlate with the harmonics of the signal. An improvement on the algorithm is achieved by using only the first and prime harmonics, which significantly reduces subharmonic errors commonly found in other pitch estimation algorithms.

CHAPTER 1
INTRODUCTION

Pitch is an important characteristic of sound, providing information about the sound's source. In speech, pitch helps to identify the gender of the speaker (pitch tends to be higher for females than for males) (Wang and Lin, 2004), gives additional meaning to words (e.g., a group of words can be interpreted as a question depending on whether the pitch is rising or not), and may help to identify the emotional state of the speaker (e.g., joy produces high pitch and a wide pitch range, while sadness produce normal to low pitch and a narrow pitch range) (Murray and Arnott, 1993). Pitch is also important in music because it determines the names of the notes (Sethares, 1998).

Pitch estimation also has applications in many areas that involve processing of sound: music, communications, linguistics, and speech pathology. In music, one of the main applications of pitch estimation is automatic music transcription. Musicologists are often faced with music for which no transcription exists. Therefore, automated tools that extract the pitch of a melody, and from there the individual musical notes, are invaluable tools for musicologists (Askenfelt, 1979). Automated transcription has also been used in query-by-humming systems (e.g., Dannenberg *et al.*, 2004). These systems allow people to search for music in databases by singing or humming the melody rather than typing the title of the song, which may be unknown for the user or the database.

In communications, pitch estimation is used for speech coding (Spanias, 1994). Many speech coding systems are based on the source-filter model (Fant, 1960), which models speech as a filtered source signal. In some implementations, the source is either a periodic sequence of glottal pulses (for voiced sound) or white noise (for unvoiced sound). Therefore, the correct estimation of the glottal pulse rate is crucial for the correct coding of voiced speech.

Pitch estimators are useful in linguistics for the recognition of intonation patterns, which are used, for example, in the acquisition of a second language (de Bot, 1983). Pitch estimators are also used in speech pathology to determine speech disorders, which are characterized by high levels of noise in the voice. Since most methods used to estimate noise are based on the fundamental frequency of the signal (e.g., Yumoto, Gould, and Baer, 1982), pitch estimators are of vital importance in this area.

The goal of our work is to develop an automatic pitch estimator that operates on both speech and music. The algorithm should be competitive with the best known pitch estimators, and therefore be suitable for the many applications mentioned above. Furthermore, the algorithm should provide a measure to determine if a pitch exists or not in each region of the signal. The remaining sections of this chapter present several psychoacoustics definitions and phenomena that will be used to explain the operation and rationale of the algorithm.

1.1 Pitch Background

1.1.1 Conceptual Definition

Several conceptual definitions of pitch have been proposed. The American Standard Association (ASA, 1960) definition of pitch is

> "Pitch is that attribute of auditory sensation in terms of which sounds may be ordered on a musical scale,"

and the American National Standards Institute (ANSI, 1994) definition of pitch is

> "Pitch is that auditory attribute of sound according to which sounds can be ordered on a scale from low to high. Pitch depends mainly on the frequency content of the sound stimulus, but it also depends on the sound pressure and the waveform of the stimulus."

These definitions mention an attribute that allows ordering sounds in a scale, but they say nothing about what that attribute is.

We will propose another definition of pitch, which is based on the fundamental frequency of a signal. The fundamental frequency f_0 of a signal (sound or no sound) exists only for periodic signals, and is defined as the inverse of the period of the signal, where the period T_0 of the signal (a.k.a. fundamental period) is the minimum repetition interval of the signal $x(t)$, i.e.,

$$T_0 = \min\{T > 0 \mid \forall t : x(t) = x(t+T)\}. \qquad (1\text{-}1)$$

It is also possible, to define the fundamental frequency in the frequency domain:

$$f_0 = \max\left\{ f \geq 0 \mid \exists\{c_k\}, \exists\{\phi_k\} : x(t) = \sum_{k=0}^{\infty} c_k \sin(2\pi kft + \phi_k) \right\}. \qquad (1\text{-}2)$$

Although both equations are mathematically equivalent (i.e., it can be shown that $f_0 = 1/T_0$), they are conceptually different: Equation 1-1 looks at the signal in the time domain, while Equation 1-2 looks at the signal as a combination of sinusoids using a Fourier series expansion. The key element for periodicity in Equation 1-1 is the *equality* in $x(t) = x(t + T)$, and the key element for periodicity in Equation 1-2 is the existence of components only at *multiples* of the fundamental frequency. Unfortunately, no signal in nature is perfectly periodic because of natural variations in frequency and amplitude, and contamination produced by noise. Nevertheless, when listening to many natural signals, we perceive pitch. This suggests that, to determine pitch, the brain probably uses either a modified version of Equation 1-1, where the equality $x(t) = x(t + T)$ is substituted by an approximation, or a modified version of Equation 1-2, where noise and fluctuations in the frequency of the components are allowed. Based on this suggestion, we define pitch as the perceived "fundamental frequency" of a sound, in other words, as the estimate our brain does of the (quasi) fundamental frequency of a sound.

1.1.2 Operational Definition

Since the previous definitions of pitch do not indicate how to measure it, they are of no practical use, and an operational definition of pitch is required. The usual way in which pitch is measured is the following. A listener is presented with two sounds: a target sound, for which the pitch is to be determined, and a matching sound. The matching sound is usually a pure tone, although sometimes harmonic complex tones are used as well. The levels of the target and the matching sounds are usually equalized to avoid any effect of differences in level in the perception of pitch. The sounds are presented sequentially, simultaneously, or in any combination of them, depending on the design of the experiment. The listener is asked to adjust the fundamental frequency of the matching sound such that it matches the target sound, in the sense of the conceptual definitions of pitch presented above. The fundamental frequency of the matching sound is recorded and the experiment is repeated several times and with different listeners. The data is summarized, and if the distribution of fundamental frequencies shows a clear peak around a certain frequency, the target sound is said to have a pitch corresponding to that frequency.

1.1.3 Strength

Some sounds elicit a strong pitch sensation, and some do not. For example, when we speak, some sounds are highly periodic and elicit a strong pitch sensation (e.g., vowels), but some do not (e.g., some consonants: /*s*/, /*sh*/, /*p*/, and /*k*/). In the case of musical instruments, the attack tends to contain transient components that obscure the pitch, but they disappear quickly letting the pitch show more clearly. The quality of the sound that allows us to determine whether pitch exists is called *pitch strength*. Pitch strength is not a categorical variable but a continuum. Also, pitch strength is independent of pitch: two sounds may have the same pitch and differ in pitch strength. For example, a pure tone and a narrow band of noise centered at the frequency of the tone have the same pitch, however, the pure tone elicit a stronger pitch sensation than the noise.

Unfortunately, not much research exists on pitch strength, and the few studies that exist have concentrated mostly on noise (Yost, 1996; Wiegrebe and Patterson, 1998), although some have explored harmonic sounds as well (Fastl and Stoll, 1979; Shofner and Selas, 2002). In terms of variety of sounds, the most complete study is probably Fastl and Stoll's, which included pure tones, complex tones, and several types of noises. In that study, pure tones were reported to have the strongest pitch among all sounds. However, other studies have found that pitch identification improves as harmonics are added (Houtsma, 1990), which suggests that pitch strength increases as well.

We hypothesize that our brain determines pitch by searching for a match between our voice, produced or imagined, and the target signal for which pitch is to be determined, probably based on their spectra. This hypothesis agrees with studies of pitch determination in which subjects have been allowed to hum the target sound to facilitate pitch matching tasks (Houtgast, 1976). Based on this hypothesis, we believe that the higher the similarity of the target signal with our voice, the higher its pitch strength. If the similarity is based on the spectrum, a signal will have maximum pitch strength when its spectrum is closest to the spectrum of a voiced sound. If we assume that voiced sounds have harmonic spectra with envelopes that decay on average as $1/f$ (i.e., inversely proportional to frequency) (Fant, 1960), then a signal will have maximum pitch strength if its spectrum has that structure.

An example of a signal with such property is a sawtooth waveform, which is exemplified in Figure 1-1. A sawtooth waveform is formed by adding sines with frequencies that are multiples of a common fundamental f_0, and whose amplitude decays inversely proportional to frequency:

$$x(t) = \sum_{k=1}^{\infty} \frac{1}{k} \sin 2\pi k f_0 t \ . \qquad (1\text{-}3)$$

Though sawtooth waveforms play a key role in our research, their importance resides in their spectrum, and not in their time-domain waveform. In particular, the phase of the components can be manipulated (destroying its sawtooth waveform shape) and the signal would still play the same role in our work as the sawtooth waveform. In other words, it is assumed in this work that what matters to estimate pitch and its strength is the amplitude of the spectral components of the sound, and not their phase, which in fact is ignored here. However, phase *does* play a role in pitch perception, as have been shown by some researchers (Moore, 1977; Shackleton and Carlyon, 1994; Galembo, et al., 2001). These researchers have created pairs of sounds that have the same spectral amplitudes but significantly different pitches, by choosing the phases of the components appropriately. Nevertheless, it is not the aim of this research to cover the whole range of pitch phenomena, but to concentrate only on the most common speech and

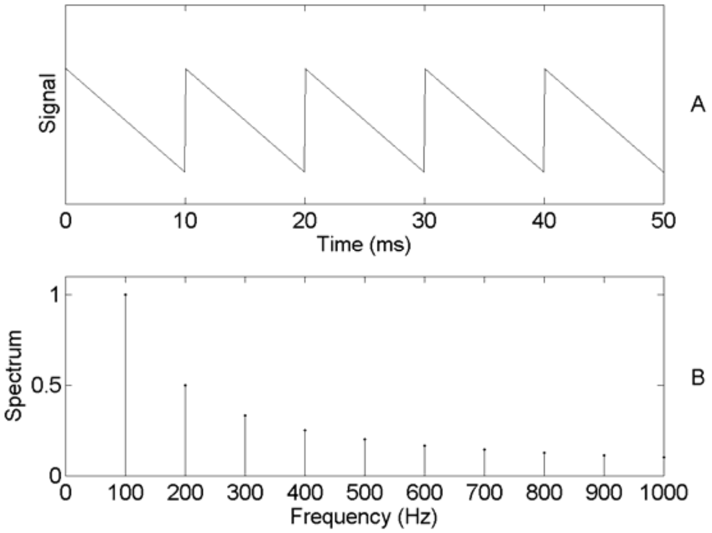

Figure 1-1. Sawtooth waveform. A) Signal. B) Spectrum.

Object 1-1. Sawtooth waveform (WAV file, 32 KB).

musical instruments sounds. As we will see later, good pitch predictions are obtained for these types of sounds based solely on the amplitude of their spectral components.

1.1.4 Duration Threshold

Doughty and Garner (1947) studied the minimum duration required to perceive a pitch for a pure tone. They found that there are two duration thresholds with two different properties. Tones with durations below the shorter threshold are perceived as a click, and no pitch is perceived. Tones with durations between the two thresholds are perceived as having pitch, and an increase in their duration causes an increase in their pitch strength. Tones with durations above the largest threshold are also perceived as having pitch, but further increases in their duration do not increase their pitch strength.

These thresholds are not constant, but approximately proportional to the pitch period of the tone. In other words, the threshold corresponds to a certain number of periods of the tone. However, there is some interaction between pitch and the minimum number of cycles required to perceive it (lower frequencies have a tendency to require fewer cycles to elicit a pitch). The shorter threshold is approximately two to four cycles, and the larger threshold is approximately three to ten cycles. For frequencies above 1 kHz the thresholds become constant: 4 ms the shorter and 10 ms the larger, regardless of their corresponding number of cycles.

Robinson and Patterson (1995a; 1995b) studied note discriminability as a function of the number of cycles in the sound using strings, brass, flutes, and harpsichords. A large increase in discriminability can be observed in their data as the number of cycles increases from one to about ten, but beyond ten cycles the discriminability of the notes does not seem to increase much. This trend agrees with the thresholds for pure tones mentioned above, which suggests that the thresholds are also valid for musical instruments, and probably for sawtooth waveforms as well.

1.2 Illustrative Examples and Pitch Determination Hypotheses

In previous sections, conceptual and operational definitions of pitch were given. From a practical point of view, both types of definitions are of limited use since the conceptual definitions are too abstract and the operational definition requires a human to determine the pitch. In this section we propose more algorithmic ways for determining pitch, through the search for cues that may give us hints regarding the pitch. These cues, hereafter referred as hypotheses, are illustrated with examples of sounds in which they are valid, and examples in which they are not.

1.1.2 Pure Tone

From a frequency domain point of view, the simplest periodic sound is a pure tone. A pure tone with a frequency of 100 Hz and its spectrum is shown in Figure 1-2. Based on our operational definition of pitch (i.e., the one that uses a pure tone as matching tone presented at the same intensity level as the testing tone), the pitch of a pure tone is its frequency, and

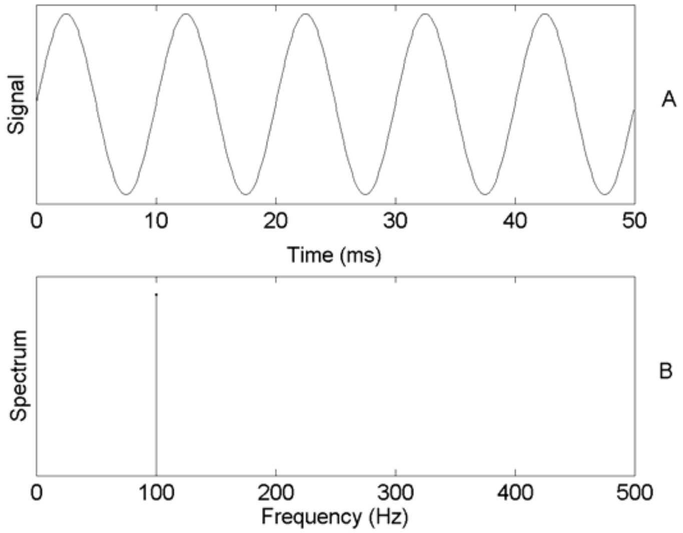

Figure 1-2. Pure tone. A) Signal. B) Spectrum.

Object 1-2. Pure tone (WAV file, 32 KB).

therefore frequency determines pitch in this case. This may not be true if the tones are presented at different levels. Intriguingly, the pitch of a pure tone may change with intensity level (Stevens, 1935): as intensity increases, the pitch of high frequency tones tends to increase, and the pitch of low frequency tones tends to decrease. However, this change is usually less than 1% or 2% (Verschuure and van Meeteren, 1975), occurs at very disparate intensity levels, and varies significantly from person to person.

Since the goal in this research is to predict pitch for sounds represented in a computer as a sequence of numbers without knowing the level at which the sound will be played, it will be assumed that the sound will be played at a "comfortable" level, and therefore the algorithm will be designed to predict the pitch at that level. Nevertheless, variations of pitch with level are small, and have little effect even for complex tones (Fastl, 2007), otherwise, music would become out of tune as we change the volume.

1.2.2 Sawtooth Waveform and the Largest Peak Hypothesis

The sawtooth waveform presented in Section 1.1.3 was shown to have a harmonic spectrum with components whose amplitude decays inversely proportional to frequency (see Figure 1-1). The computational determination of the pitch of a sawtooth waveform is not as easy as it is for a pure tone because its spectrum has more than one component. Since the pitch of a sawtooth waveform corresponds to its fundamental frequency, and the fundamental frequency in this case is the component with the highest energy, one possible hypothesis for the derivation of the pitch is that the pitch corresponds to the largest peak in the spectrum. However, as we will show in the next section, this hypothesis does not always hold.

1.2.3 Missing Fundamental and the Components Spacing Hypothesis

This section shows that it is possible to create a periodic sound with a pitch corresponding to a frequency at which there is no energy in the spectrum. A sound with such property is said to

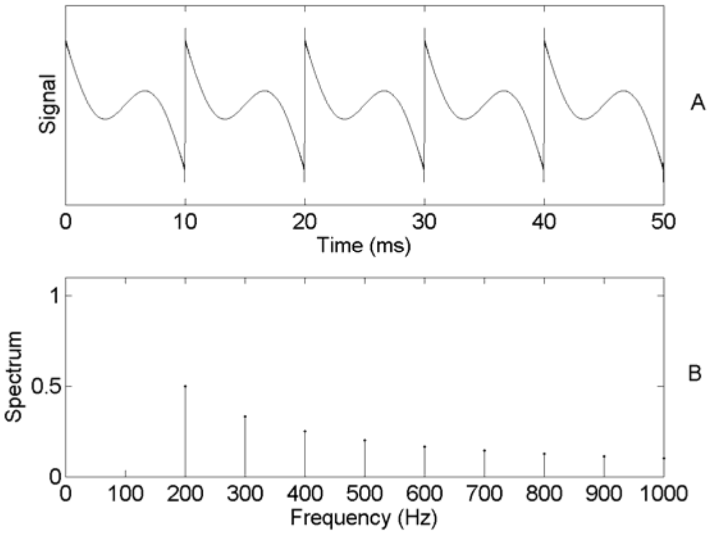

Figure 1-3. Missing fundamental. A) Signal. B) Spectrum.

Object 1-3. Missing fundamental (WAV file, 32 KB).

have a *missing fundamental*. It is easy to build such a signal: just take a sawtooth waveform and remove its fundamental, as shown in Figure 1-3. Certainly, the timbre of the sound will change, but not its pitch. This fact disproves the hypothesis that the pitch corresponds to the largest peak in the spectrum.

After it was realized that the pitch of a complex tone was unaffected by removing the fundamental frequency, it was hypothesized that the pitch corresponds to the spacing of the frequency components. However, this hypothesis is not always valid, as we will show in the next section.

1.2.4 Square Wave and the Maximum Common Divisor Hypothesis

The previous section hypothesized that the pitch corresponds to the spacing between the frequency components. However, it is easy to find an example for which this hypothesis fails: a

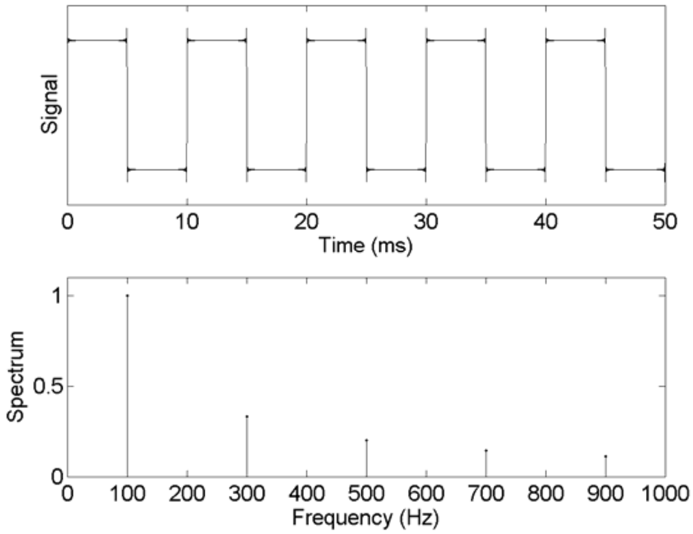

Figure 1-4. Square wave. A) Signal. B) Spectrum.

Object 1-4. Square wave (WAV file, 32 KB).

square wave. A square wave is similar to a sawtooth wave, but does not have even order harmonics:

$$x(t) = \sum_{k=1}^{\infty} \frac{1}{(2k-1)} \sin 2\pi (2k-1) f_0 t \ . \qquad (1\text{-}4)$$

A square wave with a fundamental frequency of 100 Hz and its spectrum is shown in Figure 1-4. The components are located at odd multiples of 100 Hz, producing a spacing of 200 Hz between them. However, the fundamental frequency, and indeed its pitch, is 100 Hz. Thus, the components spacing hypothesis is invalid.

A hypothesis that seems to work for this example, and all the previous ones, is that the pitch must correspond to the maximum common divisor of the frequency components. As shown in Equation 1-2, this is equivalent to saying that the pitch corresponds to the fundamental frequency. However, we will show in the next section that this hypothesis is also wrong.

1.2.5 Alternating Pulse Train

A pulse train is a sum of pulses separated by a constant time interval T_0:

$$x(t) = \sum_{k=1}^{\infty} \delta(t - kT_0), \qquad (1\text{-}5)$$

where δ is the delta or pulse function, a function whose value is one if its argument is zero, and zero otherwise. A pulse train with a fundamental frequency of 100 Hz (fundamental period of 10 ms) and its spectrum are shown in Figure 1-5. The spectrum of a pulse train is another pulse train with pulses at multiples of the fundamental frequency, which corresponds to the pitch. If the signal is modified by decreasing the height of every other pulse in the time domain to 0.7, as shown in Figure 1-6, the period of the signal will change to 20 ms. This will be reflected in the spectrum as a change in the fundamental frequency from 100 Hz to 50 Hz. However, although this change may cause an effect on the timbre (depending on the overall level of the signal), the

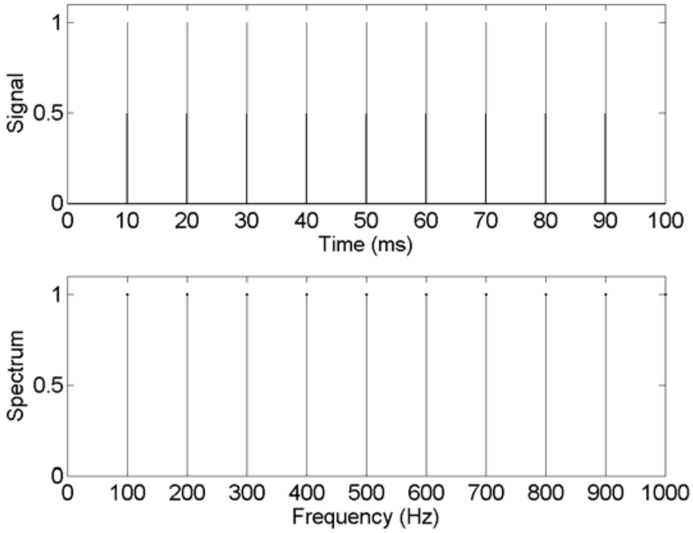

Figure 1-5. Pulse train. A) Signal. B) Spectrum.

Object 1-5. Pulse train (WAV file, 32 KB).

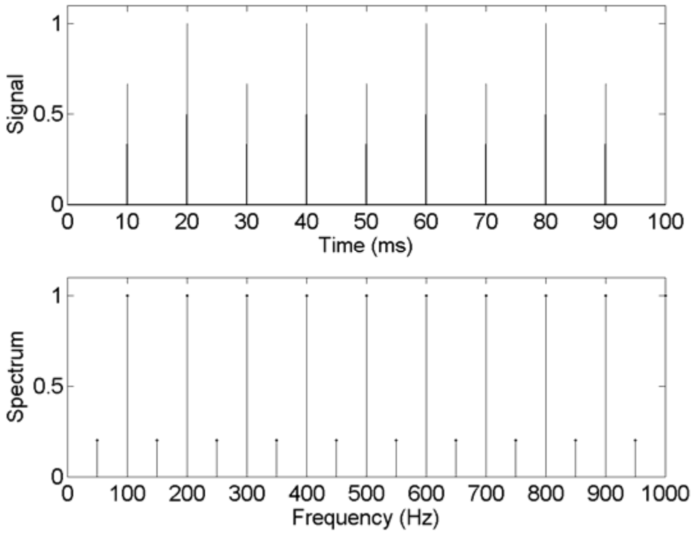

Figure 1-6. Alternating pulse train. A) Signal. B) Spectrum.

Object 1-6. Alternating pulse train (WAV file, 32 KB).

pitch will remain the same: 100 Hz, refuting the hypothesis that the pitch of a sound corresponds to its fundamental frequency (i.e., the maximum common divisor of the frequency components).

1.2.6 Inharmonic Signals

This section shows another example of a signal whose pitch does not correspond to its fundamental frequency (i.e., the maximum common divisor of its frequency components). Consider a signal built from the 13th, 19th, and 25th harmonics of 50 Hz (i.e., 650, 950, and 1250 Hz), as shown in Figure 1-7. Its fundamental frequency is 50 Hz, but its pitch is 334 Hz (Patel and Balaban, 2001). This is interesting since the ratios between the components and the pitch are far from being integer multiples: 1.95, 2.84, and 3.74. In any case, the pitch of the signal no longer corresponds to its fundamental frequency. Although the true period of the signal is $T_0 = 20$ ms, the signal peaks about every 3 ms, which corresponds to the pitch period of the

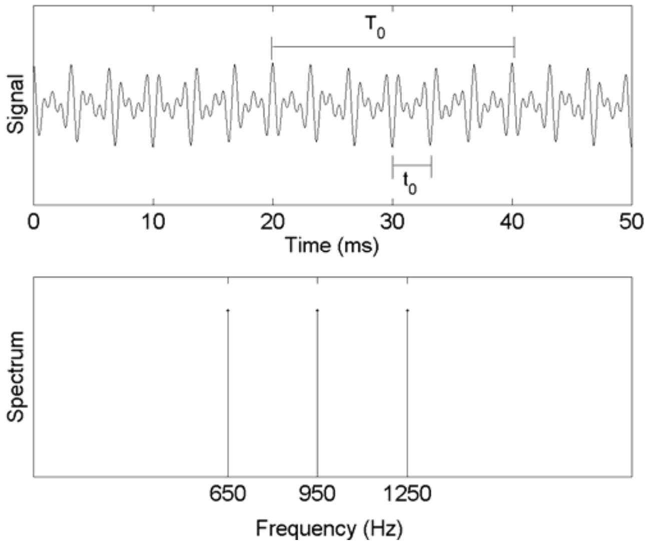

Figure 1-7. Inharmonic signal. A) Signal. T_0 corresponds to the fundamental period of the signal and t_0 corresponds to the pitch period. B) Spectrum.

Object 1-7. Inharmonic signal (WAV file, 32 KB).

signal t_0 (see Panel A). These type of signals for which the components are not integer multiples of the pitch are called *inharmonic signals*.

1.3 Loudness

Loudness is another perceptual quality of sound that provides us with information about its source. It is important for pitch because the unification of the components of a sound into a single entity, for which we identify a pitch, may be mediated by the relative loudness of the components of the sound.

A conceptual definition of loudness is (Moore, 1997)

> "…that attribute of auditory sensation in terms of which sounds can be ordered on a scale extending from quiet to loud."

The most common unit to measure loudness is the *sone*. A sone is defined as the loudness elicited by a 1 kHz tone presented at 40 dB sound pressure level. The loudness L of a pure tone is usually modeled as a power function of the sound pressure P of the tone, i.e.,

$$L = k\ P^{\alpha}, \tag{1-6}$$

where k is a constant that depends on the units and α is the exponent of the power law.

In a review of loudness studies, Takeshima *et. al* (2003) found that the value of α is usually reported to be within the range 0.4-0.6. They also reviewed more elaborate models with many more parameters, but for simplicity, in this work we will use the simpler power model, and for reasons we will explain later, we choose the value of α to be 0.5. In other words, we model the loudness of a tone as being proportional to the square-root of its amplitude.

1.4 Equivalent Rectangular Bandwidth

The bandwidth and the distribution of the filters used to extract the spectral components of a sound are important issues that may affect our perception of pitch. Since each point of the cochlea responds better to certain frequencies than others, the cochlea acts as a spectrum analyzer. The bandwidth of the frequency response of each point of the cochlea is not constant but varies with frequency, being almost proportional to the frequency of maximum response at each point (Glasberg and Moore, 1990).

The concept of Equivalent Rectangular Bandwidth (ERB) was introduced as a description of the spread of the frequency response of a filter. The ERB of a filter F is defined as the bandwidth (in Hertz) of a rectangular filter R centered at the frequency of maximum response of F, scaled to have the same output as F at that frequency, and passing the overall same amount of white noise energy as F. In other words, when the power responses of F and R are plotted as a

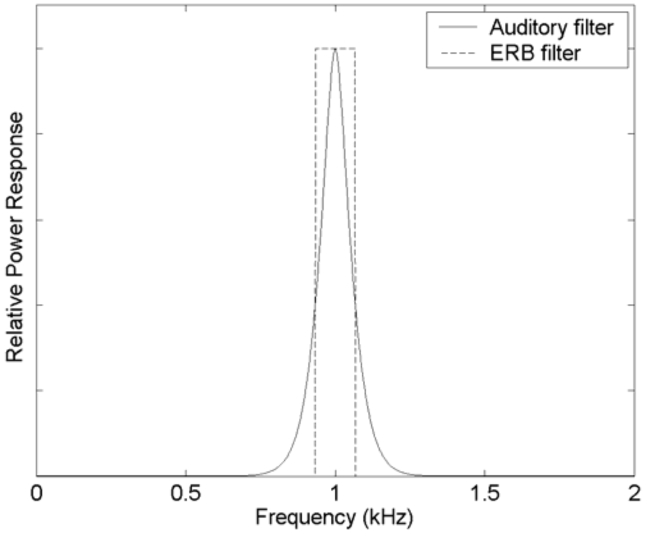

Figure 1-8. Equivalent rectangular bandwidth.

function of frequency, as in Figure 1-8, the central frequency of *R* corresponds to the mode of *F*, and both curves have the same height and area.

Glasberg and Moore (1990) studied the response of auditory filters at different frequencies, and proposed the following formula to approximate the ERB of the filters:

$$\mathrm{ERB}(f) = 24.7 + 0.108f \, . \tag{1-7}$$

Another property of the cochlea is that the relation between frequency and site of maximum excitation in the cochlea is not linear. If the distance between the apex of the cochlea and the site of maximum excitation of a pure tone is plotted as a function of frequency of the tone, it will be found that a displacement of 0.9 mm in the cochlea corresponds approximately to one ERB (Moore, 1986). Therefore, it is possible to build a scale to measure the position of maximum response in the cochlea for a certain frequency *f* by integrating Equation 1-7 to obtain the number of ERBs below *f*, and then multiplying it by 0.9 mm to obtain the position. However,

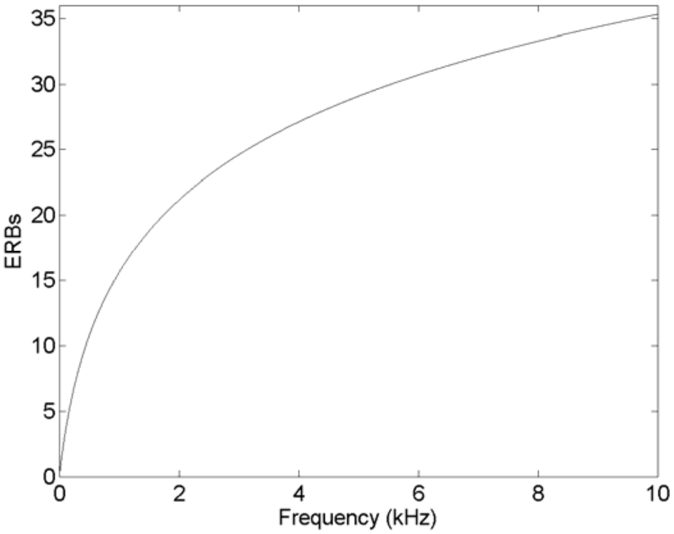

Figure 1-9. Equivalent-rectangular-bandwidth scale.

it is common practice in psychoacoustics to merely compute the number of ERBs below *f*, which can be computed as

$$\mathrm{ERBs}(f) = 21.4\log_{10}(1 + f/229)\,. \tag{1-8}$$

This scale is shown in Figure 1-9, and it will be the scale used by SWIPE to compute spectral similarity.

1.5 Dissertation Organization

The rest of this dissertation is organized as follows. Chapter 2 presents previous pitch estimation algorithms that are related to SWIPE, their problems and possible solutions to these problems. Chapter 3 will discuss how these problems, plus some ones and their solutions, lead to SWIPE. Chapter 4 evaluates SWIPE using publicly available speech/music databases and a disordered speech database. Publicly available implementations of other algorithms are also evaluated on the same databases, and their performance is compared against SWIPE's.

1.6 Summary

Here we have presented the motivations and applications for pitch estimation. Then, we presented conceptual and operational definitions of pitch, together with the related concept of pitch strength and the duration threshold to perceive pitch. Next, we presented examples of signals and their pitch, together with hypotheses about how pitch is determined. The sawtooth waveform was highlighted, since it plays a key role in the development of SWIPE. Psychoacoustic concepts such as inharmonic signals, loudness, and the ERB scale were also introduced since they are also relevant for the development of SWIPE.

CHAPTER 2
PITCH ESTIMATION ALGORITHMS: PROBLEMS AND SOLUTIONS

This chapter presents some well known pitch estimation algorithms that appear in the literature. These algorithms were chosen because of their influence upon the creation of SWIPE. We will present the algorithms in a very basic form with the intent to capture their essence in a simple expression, although their actual implementations may have extra details that we do not present here. The purpose of those details is usually to fine tune the algorithms, but the actual power of the algorithms is based on the essence we describe here.

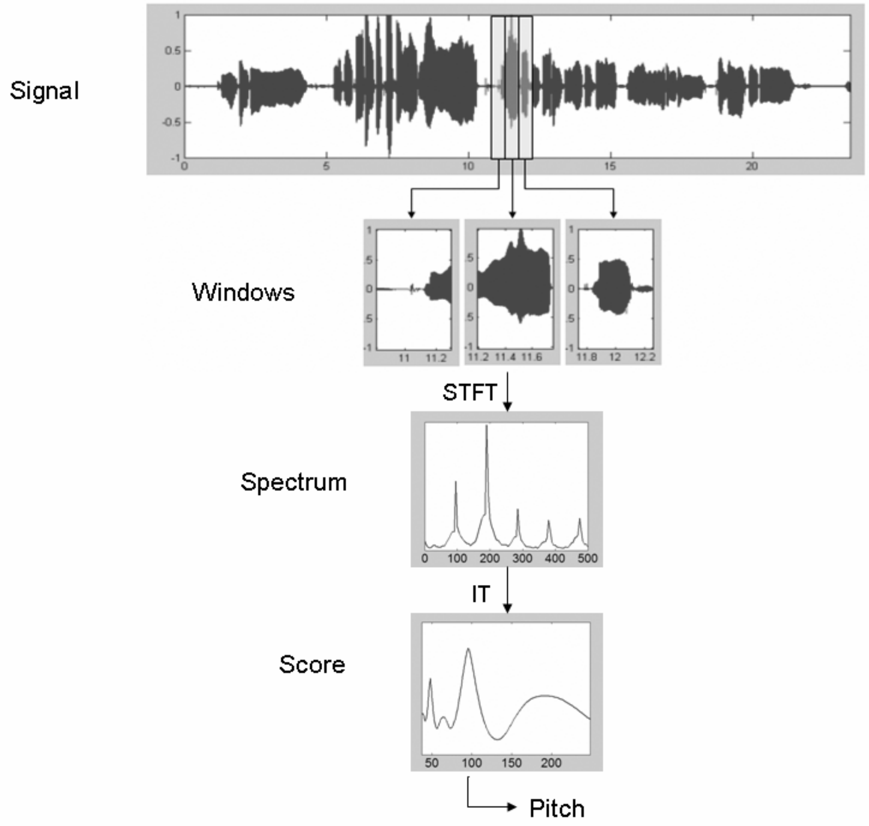

Figure 2-1. General block diagram of pitch estimators.

Traditionally, there have been two types of pitch estimation algorithms (PEAs): algorithms based on the spectrum[1] of the signal, and algorithms based on the time-domain representation of the signal. The time-domain based algorithms presented in this chapter can also be formulated based on the spectrum of the signal, which will be the approach followed here.

The basic steps that most PEAs perform to track the pitch of a signal are shown in the block diagram of Figure 2-1. First, the signal is split into windows. Then, for each window the following steps are performed: (i) the spectrum is estimated using a short-time Fourier transform (STFT), (ii) a score is computed for each pitch candidate within a predefined range by computing an integral transform (IT) over the spectrum, and (iii) the candidate with the highest score is selected as the estimated pitch. The algorithms will be presented in an order that is convenient for our purposes, but does not necessarily correspond to the chronological order in which they were developed.

2.1 Harmonic Product Spectrum (HPS)

The first algorithm to be presented is Harmonic Product Spectrum (HPS) (Schroeder, 1968). This algorithm estimates the pitch as the frequency that maximizes the product of the spectrum at harmonics of that frequency, i.e. as

$$p = \arg\max_{f} \prod_{k=1}^{n} | X(kf) |, \tag{2-1}$$

where X is the estimated spectrum of the signal, n is the number of harmonics to be used (typically between 5 and 11), and p is the estimated pitch. The purpose of limiting the number of harmonics to n is to reduce the computational cost, but there is no logical reason behind this limit; it is hard to believe that the n-th harmonic is useful for pitch estimation, but not the n+1-th.

[1] Since all the pitch estimators presented here use the magnitude of the spectrum but not its phase, the words "magnitude of" will be omitted, and the word *spectrum* should be interpreted as *magnitude of the spectrum* unless explicitly noted otherwise.

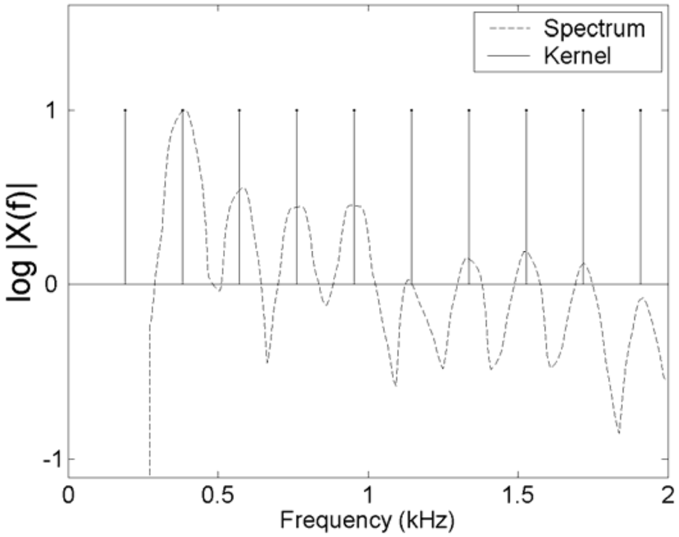

Figure 2-2. Harmonic product spectrum.

Object 2-1. Bandpass filtered /*u*/ (WAV file 6 KB)

Since the logarithm is an increasing function, an equivalent approach is to estimate the pitch as the frequency that maximizes the logarithm of the product of the spectrum at harmonics of that frequency. Since the logarithm of a product is equal to the sum of the logarithms of the terms, HPS can also be written as

$$p = \arg\max_{f} \sum_{k=1}^{n} \log |X(kf)|, \tag{2-2}$$

or using an integral transform, as

$$p = \arg\max_{f} \int_{0}^{\infty} \log |X(f')| \sum_{k=1}^{n} \delta(f'-kf)\ df'. \tag{2-3}$$

Figure 2-2 shows the kernel of this integral for a pitch candidate with frequency 190 Hz.

A pitfall of this algorithm is that if any of the harmonics is missing (i.e., its energy is zero), the product will be zero (equivalently, the sum of the logarithms will be minus infinity) for the candidate corresponding to the pitch, and therefore the pitch will not be recognized. Figure 2-2 also shows the spectrum of the vowel /*u*/ (as in *good*) with a pitch of 190 Hz (Object 2-1). This sample was passed through a filter with a bandpass range of 300–3400 Hz to simulate telephone-quality speech. Therefore, the fundamental is missing and HPS is not able to recognize the pitch of this signal. Another salient characteristic of this sample is its intense second harmonic at 380 Hz, caused probably by the first formant of the vowel, which is on average around 380 Hz as well (Huang, Acero, and Hon, 2001).

2.2 Sub-harmonic Summation (SHS)

An algorithm that has no problem with missing harmonics is Sub-Harmonic Summation (SHS) (Hermes, 1988), which solves the problem by using addition instead of multiplication. Therefore, if any harmonic is missing, it will not contribute to the total, but will not bring the sum to zero either. In mathematical terms, SHS estimates the pitch as

$$p = \arg\max_{f} \sum_{k=1}^{n} | X(kf) |, \tag{2-4}$$

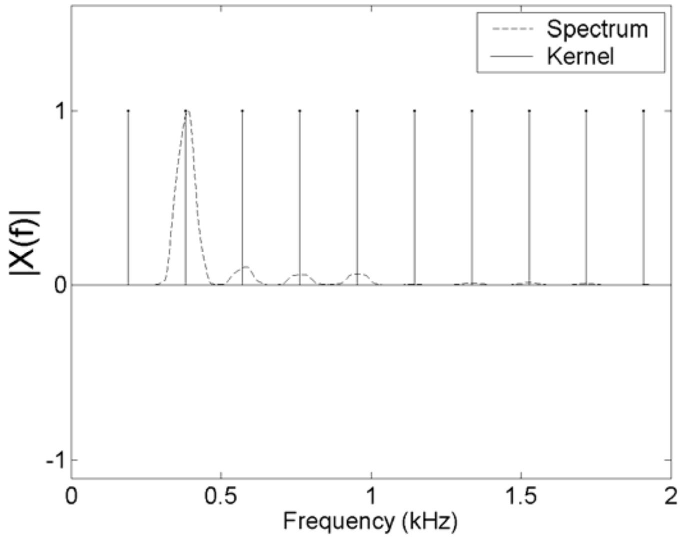

Figure 2-3. Subharmonic summation.

or using an integral transform as

$$p = \arg\max_{f} \int_{0}^{\infty} |X(f')| \sum_{k=1}^{n} \delta(f'-kf) \; df' . \tag{2-5}$$

An example of the kernel of this integral is shown in Figure 2-3.

A pitfall of this algorithm is that since it gives the same weight to all the harmonics, subharmonics of the pitch may have the same score as the pitch, and therefore they are valid candidates for being recognized as the pitch. For example, suppose that a signal has a spectrum consisting of only one component at *f* Hz. By definition, the pitch of the signal is *f* Hz as well. However, since the algorithm adds the spectrum at *n* multiples of the candidate, each of the subharmonics *f*/2, *f*/3,…, *f*/*n* will have the same score as *f*, and therefore they are equally valid to be recognized as the pitch.

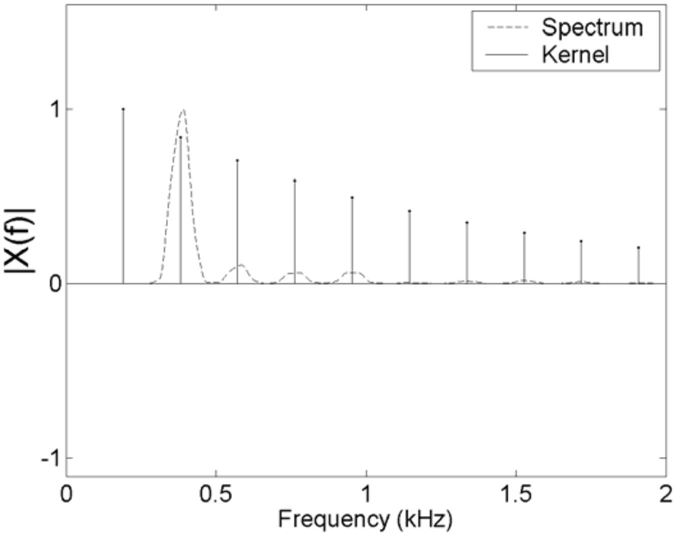

Figure 2-4. Subharmonic summation with decay.

This problem can be solved by introducing a monotonically decaying weighting factor for the harmonics. SHS implements this idea by weighting the harmonics with a geometric progression as

$$p = \arg\max_{f} \int_{0}^{\infty} |X(f')| \sum_{k=1}^{n} r^{k-1}\delta(f'-kf)\ df', \tag{2-6}$$

where the value of *r* was empirically set to 0.84 based on experiments using speech. The kernel of this integral is shown in Figure 2-4. SHS is the only algorithm in this chapter that solves the subharmonic problem by applying this decay factor. Later, another algorithm will be presented (Biased Autocorrelation) which solves this problem in a different way.

2.3 Subharmonic to Harmonic Ratio (SHR)

A drawback of the algorithms presented so far is that they examine the spectrum only at the harmonics of the fundamental, ignoring the contents of the spectrum everywhere else. An example will illustrate why this is a problem. Suppose that the input signal is white noise (i.e., a signal with a flat spectrum). This signal is perceived as having no pitch. However, the previous algorithms will produce the same score for each pitch candidate, making each of them a valid estimate for the pitch.

This problem is solved by the Subharmonic to Harmonic Ratio algorithm (SHR) (Sun, 2000), which not only adds the spectrum at harmonics of the pitch candidate, but also subtracts the spectrum at the middle points between harmonics. However, this algorithm uses the logarithm of the spectrum, and therefore has the problem previously discussed for HPS. Also, this algorithm gives the same weight to all the harmonics and therefore it suffers from the subharmonics problem. SHR can be written as

$$p = \arg\max_{f} \int_{0}^{\infty} \log|X(f')| \sum_{k=1}^{n} \delta(f'-kf) - \delta(f'-(k-1/2)f)\ df'. \tag{2-7}$$

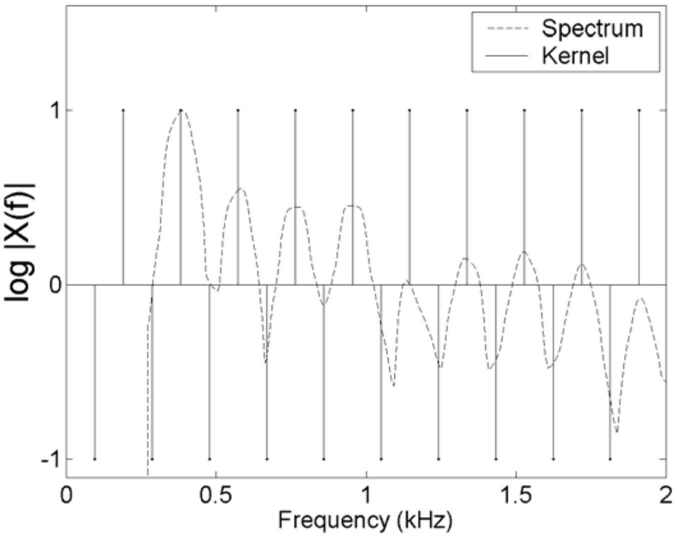

Figure 2-5. Subharmonic to harmonic ratio.

The kernel of the integral is shown in Figure 2-5. Notice that SHR will produce a positive score for a signal with a harmonic spectrum and a score of zero for white noise. However, this algorithm has a problem that is shared by all the algorithms presented so far: since they examine the spectrum only at harmonic locations, they cannot recognize the pitch of inharmonic signals.

Before we move on to the next algorithm, we wish to add some insight to SHR. If we further divide the sum in Equation 2-7 by *n*, the algorithm would compute the average peak-to-valley ratio, where the peaks are expected to be at the harmonics of the candidate, and the valleys are expected to be at the middle point between harmonics. This idea will be exploited later by SWIPE, albeit with some refinements: the average will be weighted, the ratio will be replaced with the distance, and the peaks and valleys will be examined over wider and blurred regions.

2.4 Harmonic Sieve (HS)

One algorithm that is able to recognize the pitch of some inharmonic signals is the Harmonic Sieve (HS) (Duifhuis and Willems, 1982). This algorithm is similar to SHS, but has two key differences: instead of using pulses it uses rectangles, and instead of computing the inner product between the spectrum and the rectangles, it counts the number of rectangles that contain at least one component (a rectangle is said to contain a component if the component fits within the rectangle and its amplitude exceeds a certain threshold *T*). The rectangles are centered at the harmonics of the pitch candidates, and their width is 8% of the frequency of the harmonics. This algorithm can be expressed mathematically as

$$p = \arg\max_{f} \sum_{k=1}^{n} \left[T < \max_{f' \in (0.96kf, 1.04kf)} |X(f')| \right], \tag{2-8}$$

where [·] is the Iverson bracket (i.e., produces a value of one if the bracketed proposition is true, and zero otherwise). Notice that the expression in the sum is a non-linear function of the spectrum, and therefore this algorithm cannot be written using an integral transform. Figure 2-6 shows the kernel used by this algorithm.

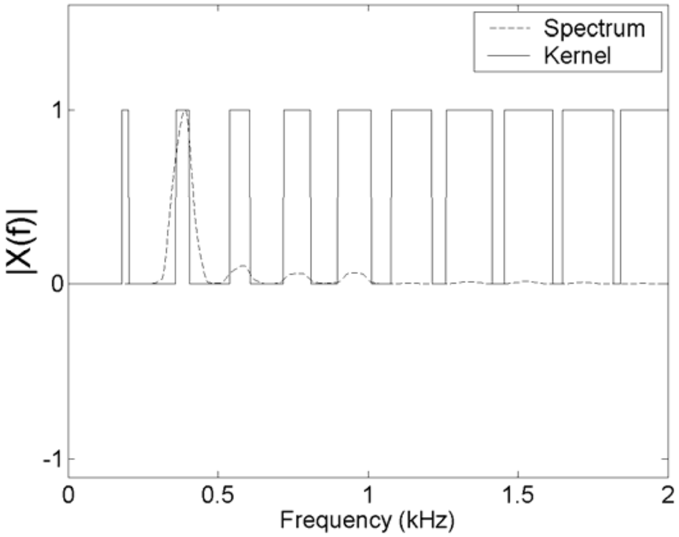

Figure 2-6. Harmonic sieve.

A pitfall of HS is that, when a component is close to an edge of a rectangle, a slight change in its frequency could put it in or out of the rectangle, possibly changing the estimated pitch drastically. Such radical changes do not typically occur in pitch perception, where small changes in the frequency of the components lead to small changes in the perceived pitch, as mentioned in Section 1.2.6. This problem can be solved by using smoother boundaries to decide whether a component should be considered as a harmonic or not, as done by the next algorithm.

2.5 Autocorrelation (AC)

One of the most popular methods for pitch estimation is autocorrelation. The autocorrelation function $r(t)$ of a signal $x(t)$ measures the correlation of the signal with itself after a lag of size t, i.e.,

$$r(t) = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} x(t')\ x(t'+t)\ dt' . \qquad (2\text{-}9)$$

The Wiener-Khinchin theorem shows that autocorrelation can also be computed as the inverse Fourier cosine transform of the squared spectrum of the signal, i.e., as

$$r(t) = \int_{0}^{\infty} |X(f)|^2 \cos(2\pi ft)\ df . \qquad (2\text{-}10)$$

The autocorrelation-based pitch estimation algorithm (AC) estimates the pitch as the frequency whose inverse maximizes the autocorrelation function of the signal, i.e., as

$$p = \arg\max_{f<f_{\max}} \int_{0}^{\infty} |X(f')|^2 \cos(2\pi f'/f)\ df' , \qquad (2\text{-}11)$$

where the parameter $f_{\max}$ is introduced to avoid the maximum that the integral has at infinity. The kernel for this integral is shown in Figure 2-7. It is easy to see that as f increases, the kernel stretches without limit, and since the cosine starts with a value of one and decays smoothly, eventually it will give a weight of one to the whole spectrum, producing a maximum at infinity.

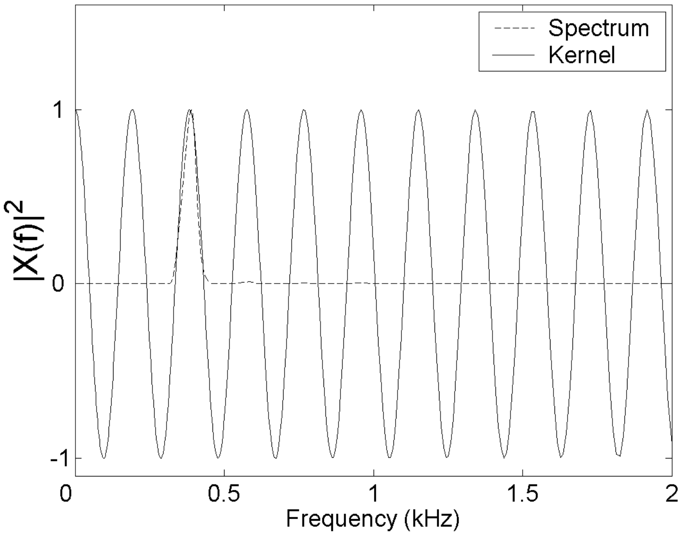

Figure 2-7. Autocorrelation.

Notice that this problem can be easily solved by removing the first quarter of the first cycle of the cosine (i.e., setting it to zero). Since the DC of a signal (i.e., it zero-frequency component) only adds a constant to the signal, ignoring the DC should not affect the pitch estimation of a periodic signal.

Because of the frequency domain representation of autocorrelation, we can see that there is a large resemblance between AC and SHR (compare the kernel of Figure 2.7 with the kernel of Figure 2.5), although with three main differences. First, instead of using an alternating sequence of pulses, AC uses a cosine, which adds a smooth interpolation between the pulses. Second, AC adds an extra lobe at DC, which was already shown to have a negative effect. Third, AC uses the power of the spectrum (i.e., the squared spectrum) instead of the logarithm of the spectrum. Therefore, both algorithms measure the average peak-to-valley distance, one in the power domain and the other in the logarithmic domain, although AC does it in a much smoother way.

There is also a similarity between AC and HS (compare the kernel of Figure 2.7 with the kernel of Figure 2.6). HS allows for inharmonicity of the components of the signal by considering as harmonic any component within a certain distance from a harmonic of the candidate pitch. AC does the same in a smoother way by assigning to a component a weight that is a function of its distance to the closer harmonic of the candidate pitch; the smaller the distance, the larger the weight, and the further the distance, the smaller the weight. In fact, if the component is too far from any harmonic, its weight can be negative.

Like all the algorithms presented so far, except SHS, AC exhibits the subharmonics problem caused by the equal weight given to all the harmonics (see Section 2.2). To solve this problem, it is common to take the local maximum of highest frequency rather than the global maximum. However, this technique sometimes fails. For example, consider a signal with fundamental frequency 200 Hz (i.e., period of 5 ms) and first four harmonics with amplitudes 1,6,1,1, as shown in Figure 2-8A (Object 2-2). Except at very low intensity levels, the four components are audible, and the pitch of the signal corresponds to its fundamental frequency. However, as shown in Figure 2-8C, AC has its first non-zero local maximum at 2.5 ms, which corresponds to a pitch of 400 Hz.

Another common solution is to use the biased autocorrelation (BAC) (Sondhi, 1968; Rabiner, 1977), which introduces a factor that penalizes the selection of low pitch values. This factor gives a weight of one to a pitch period of zero and decays linearly to zero for a pitch period corresponding to the window size *T*. This can be written as

$$p = \underset{f \in (1/T, f_{\max})}{\arg\max} \left(1 - \frac{1}{Tf}\right) \int_{0}^{\infty} |X(f')|^2 \cos(2\pi f'/f)\ df' . \qquad (2\text{-}12)$$

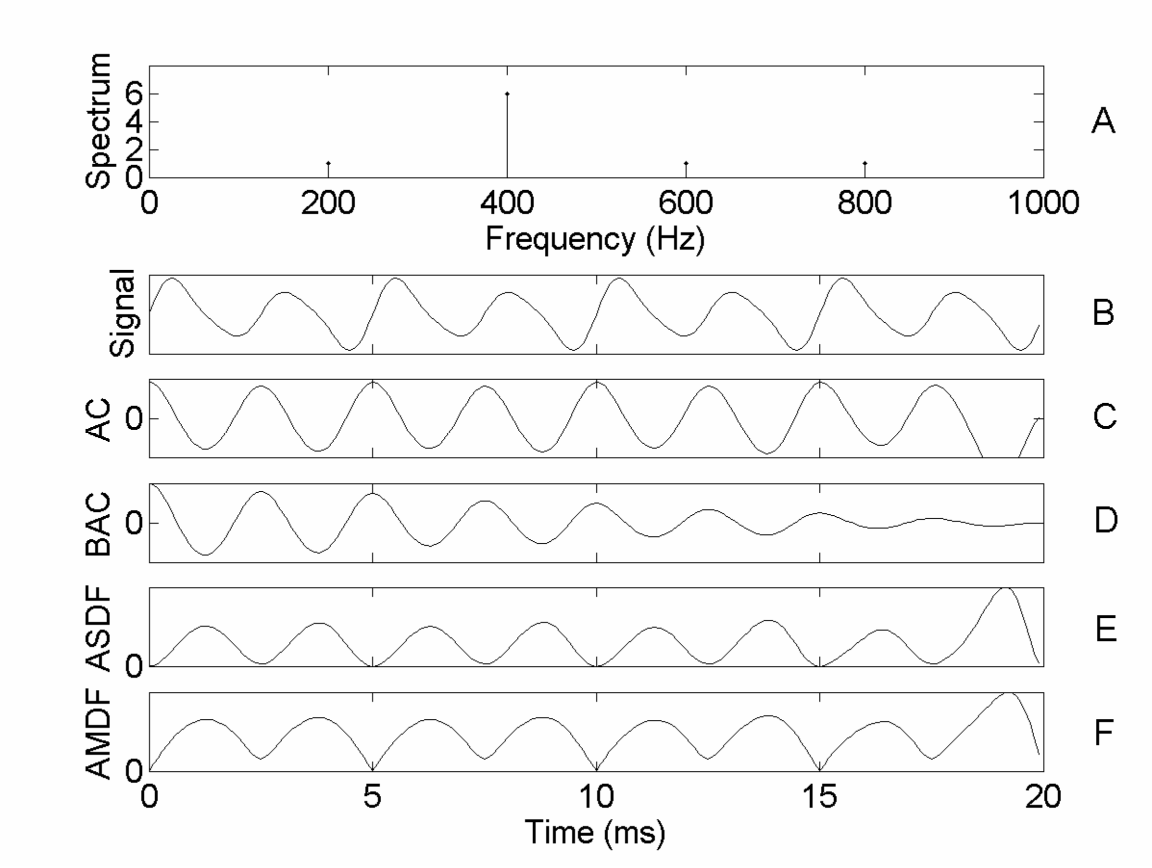

Figure 2-8. Comparison between AC, BAC, ASDF, and AMDF. A) Spectrum of a signal with pitch and fundamental frequency of 200 Hz. B) Waveform of the signal with a fundamental period of 5 msec. C) AC has a maximum at every multiple of 5 ms, making it hard to choose the best candidate. The first (non-zero) local maxima is at 2.5 ms, making the "first peak" criteria to fail. D) BAC has its first peak and its non-zero largest local maximum at 2.5 ms. E) ASDF is an inverted, shifted, ands scaled AC. F) AMDF is similar to ASDF.

Object 2-2. Signal with strong second harmonic (WAV file, 32 KB)

However, the combination of this bias and the squaring of the spectrum may introduce new problems. For example, if $T = 20$ ms as in the BAC function of Figure 2-8D, the bias will make the height of the peak at 2.5 ms larger than the height of the peak at 5 ms, consequently causing an incorrect pitch estimate.

2.6 Average Magnitude and Squared Difference Functions (AMDF, ASDF)

Two functions similar to autocorrelation (in the sense that they compare the signal with itself after a lag of size t) are the magnitude difference function (AMDF) and the average squared difference function (ASDF). The AMDF is defined as

$$d(t) = \frac{1}{T} \int_{T/2}^{T/2} |\, x(t') - x(t'+t)| \;\, dt', \tag{2-13}$$

and the ASDF as

$$s(t) = \frac{1}{T} \int_{T/2}^{T/2} [x(t') - x(t'+t)]^2 \;\, dt'. \tag{2-14}$$

It is easy to show that ASDF and autocorrelation are related through the equation (Ross, 1974)

$$s(t) = 2\big(r(0) - r(t)\big), \tag{2-15}$$

and therefore, $s(t)$ is just an inverted, shifted, and scaled version of autocorrelation. Therefore, as illustrated in the panels C (or D) and E of Figure 2-8, where (biased) autocorrelation has peaks, $s(t)$ has dips. Thus, an ASDF-based algorithm must look for minima instead of maxima to estimate pitch.

It has also been shown (Ross, 1974) that $d(t)$ can be approximated as

$$d(t) \cong \beta(t) \;\, [s(t)]^{1/2}. \tag{2-16}$$

Although the relation between $d(t)$ and $s(t)$ depends on t through $\beta(t)$, it is found in practice that this factor does not play a significant role, and a large similarity between $d(t)$ and $s(t)$ exists, as observed in panels E and F of Figure 2-8. Therefore, since the functions $r(t)$, $s(t)$, and $d(t)$ are so strongly related, none of them is expected to offer much more than the others for pitch estimation. However, modifications to these functions, which cannot be expressed in terms of the other functions, have been used successfully to improve their performance on pitch estimation.

An example is given by YIN (de Cheveigne, 2002), which uses a variation of $s(t)$ to avoid the dip at lag zero, improving its performance. Another variation is the one we proposed in the previous section (i.e., the removal of the first quarter of the cosine) to avoid the maximum at zero lag for autocorrelation.

2.7 Cepstrum (CEP)

An algorithm similar to AC is the cepstrum-based pitch estimation algorithm (CEP) (Noll, 1967). The *cepstrum* $c(t)$ of a signal $x(t)$ is very similar to its autocorrelation. The only difference is that it uses the logarithm of the spectrum instead of its square, i.e.,

$$c(t) = \int_0^\infty \log|X(f)|\cos(2\pi ft)\ df\ . \tag{2-17}$$

CEP estimates the pitch as the frequency whose inverse maximizes the cepstrum of the signal, i.e., as

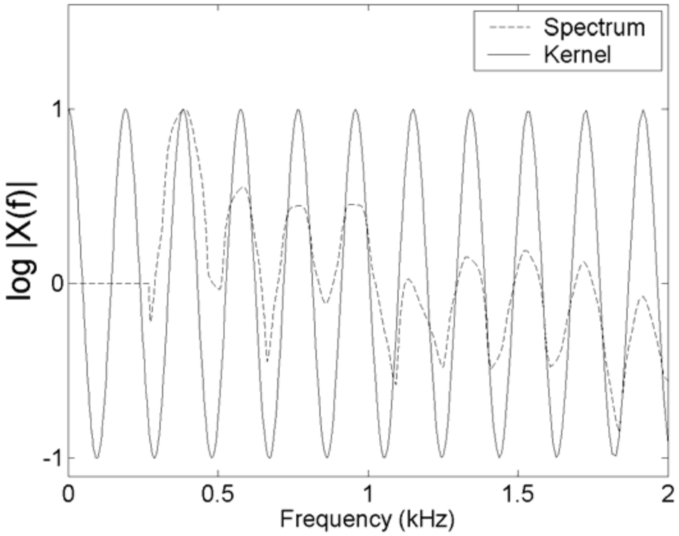

Figure 2-9. Cepstrum.

$$p = \arg\max_{f < f_{\max}} \int_0^\infty \log|X(f')| \cos(2\pi f'/f)\ df'. \qquad (2\text{-}18)$$

The kernel of this integral is shown in Figure 2-9. Like AC, CEP exhibits the subharmonics problem and the problem of having a maximum at a large value of *f*. The maximum is not necessarily at infinity because, depending on the scaling of the signal, the logarithm of the spectrum may be negative at large frequencies, and therefore assigning a positive weight to that region may in fact decrease the score. Figure 2-10 shows the spectrum of the speech signal that has been used in previous figures and the kernel that produces the highest score for that spectrum, which corresponds to a candidate pitch of about 10 kHz. Notice that the logarithm of the spectrum was arbitrarily set to zero for frequencies below 300 Hz because its original value (minus infinity) would make unfeasible the evaluation of the integral in Equation 2-18. This problem of the use of the logarithm when there are missing harmonics was already discussed in Section 2.1.

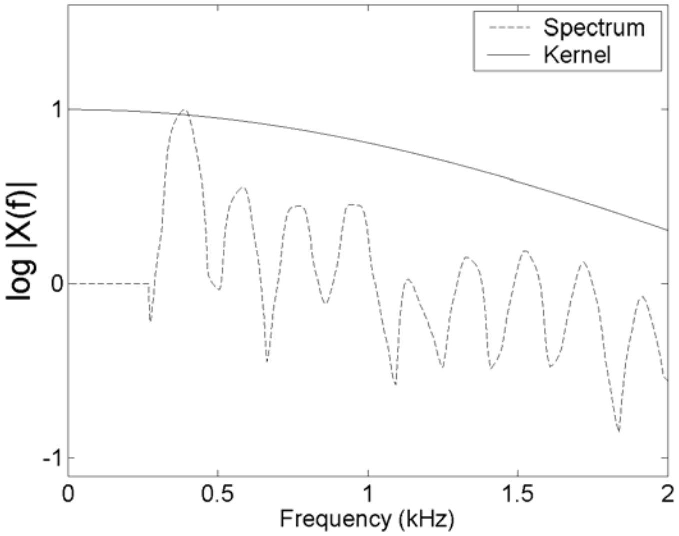

Figure 2-10. Problem caused to cepstrum by cosine lobe at DC.

2.8 Summary

In this chapter we presented pitch estimation algorithms that have influenced the creation of SWIPE. The most common problems found in these algorithms were the inability to deal with missing harmonics (HPS, SHR, and CEP) and inharmonic signals (HPS, SHS, and SHR), and the tendency to produce high scores for subharmonics of the pitch (all the algorithms, although to a lesser extent SHS and BAC). Solutions to these problems were either found in other algorithms or were proposed by us.

CHAPTER 3
THE SAWTOOTH WAVEFORM INSPIRED PITCH ESTIMATOR

Aiming to improve upon the algorithms presented in Chapter 2, we propose the Sawtooth Waveform Inspired Pitch Estimator (SWIPE)[2]. The seed of SWIPE is the implicit idea of the algorithms presented in Chapter 2: to find the frequency that maximizes the average peak-to-valley distance at harmonics of that frequency. However, this idea will be implemented trying to avoid the problem-causing features found in those algorithms. This will be achieved by avoiding the use of the logarithm of the spectrum, applying a monotonically decaying weight to the harmonics, observing the spectrum in the neighborhood of the harmonics and middle points between harmonics, and using smooth weighting functions.

3.1 Initial Approach: Average Peak-to-Valley Distance Measurement

If a signal is periodic with fundamental frequency *f*, its spectrum must contain peaks at multiples of *f* and valleys in between. Since each peak is surrounded by two valleys, the average peak-to-valley distance (APVD) for the *k*-th peak is defined as

$$d_k(f) = \frac{1}{2}\left[\ |X(kf)| - |X((k-1/2)f)|\ \right] + \frac{1}{2}\left[\ |X(kf)| - |X((k+1/2)f)|\ \right]$$

$$= |X(kf)| - \frac{1}{2}\left[\ |X((k-1/2)f)| + |X((k+1/2)f)|\ \right]. \qquad (3\text{-}1)$$

Averaging over the first *n* peaks, the global APVD is

$$D_n(f) = \frac{1}{n}\sum_{k=1}^{n} d_k(f)$$

$$= \frac{1}{n}\left[\frac{1}{2}|X(f/2)| - \frac{1}{2}|X((n+1/2)f)| + \sum_{k=1}^{n}|X(kf)| - |X((k-1/2)f)|\right]. \qquad (3\text{-}2)$$

[2] The name of the algorithm will become clear in a posterior section.

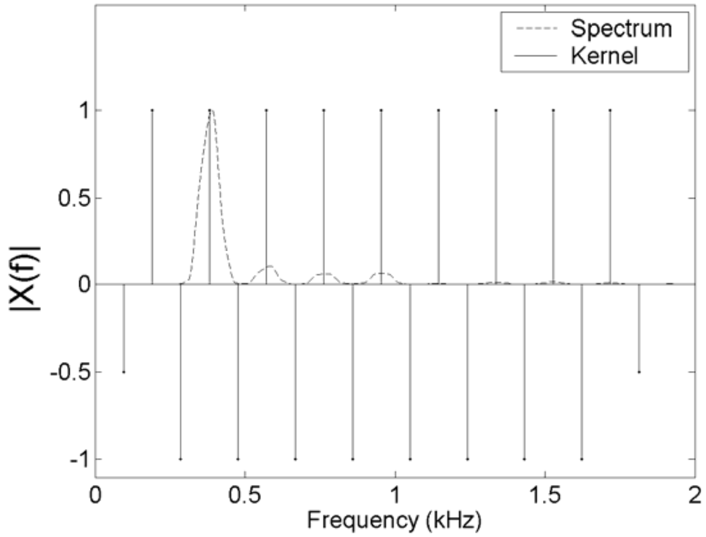

Figure 3-1. Average-peak-to-valley-distance kernel.

Our first approach to estimate pitch is to find the frequency that maximizes the APVD. Staying with the integral transform notation used in Chapter two, and dropping the unnecessary 1/n term, the algorithm can be expressed as

$$p = \underset{f<f_{\max}}{\arg\max} \int_0^\infty |X(f')| K_n(f,f')\, df', \tag{3-3}$$

where

$$K_n(f,f') = \frac{1}{2}\delta(f'/2f) - \frac{1}{2}\delta((n+1/2)f'/f) + \sum_{k=1}^{n} \delta(kf'/f) - \delta((k-1/2)f'/f)\,. \tag{3-4}$$

The kernel $K_n(f,f')$ for f = 190 Hz is shown in Figure 3-1 together with the spectrum of the sample vowel /*u*/ used in Chapter 2, which will be used extensively in this chapter as well. The kernel is a function not only of the frequencies but also of n, the number of harmonics to be used. Each positive pulse in the kernel has a weight of 1, each negative pulse between positive pulses has a weight of -1, and the first and last negative pulses have a weight of -1/2. This kernel is similar to the kernel used by SHR (see Chapter 2), with the only difference that in K_n the first negative pulse has a weight of -1/2 and K_n has an extra negative pulse at the end, also with a weight of -1/2.

3.2 Blurring of the Harmonics

The previous method of measuring the APVD works if the signal is harmonic, but not if it is inharmonic. To allow for inharmonicity, our first approach was to blur the location of the harmonics by replacing each pulse with a triangle function with base *f*/2,

$$\Lambda_f(f') = \begin{cases} f/4 - |f'| & \text{,if } |f'| < f/4 \\ 0 & \text{,otherwise.} \end{cases} \tag{3-5}$$

The base of the triangle was set to *f*/2 to produce a triangular wave as shown in Figure 3-2. To be consistent with the APVD measure, the first and last negative triangles were given a height of 1/2. One reason for using a base that is proportional to the candidate pitch is that it allows for a pitch-independent handling of inharmonicity, as seems to be done in the auditory system (see section 1.2.6).

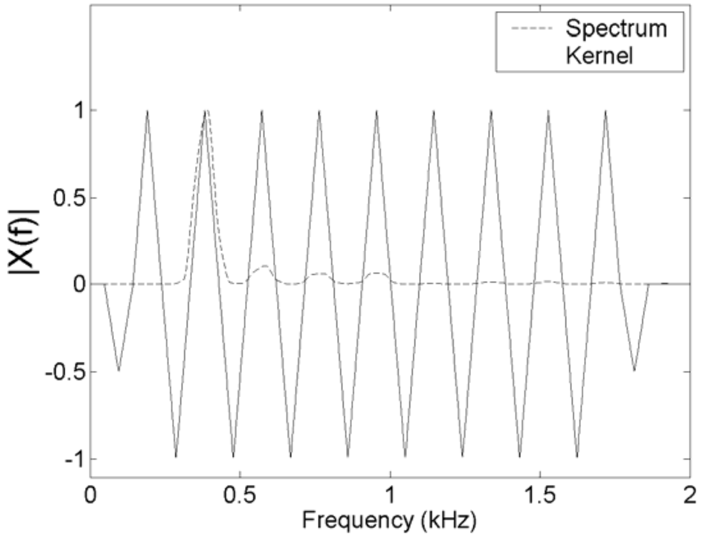

Figure 3-2. Triangular wave kernel.

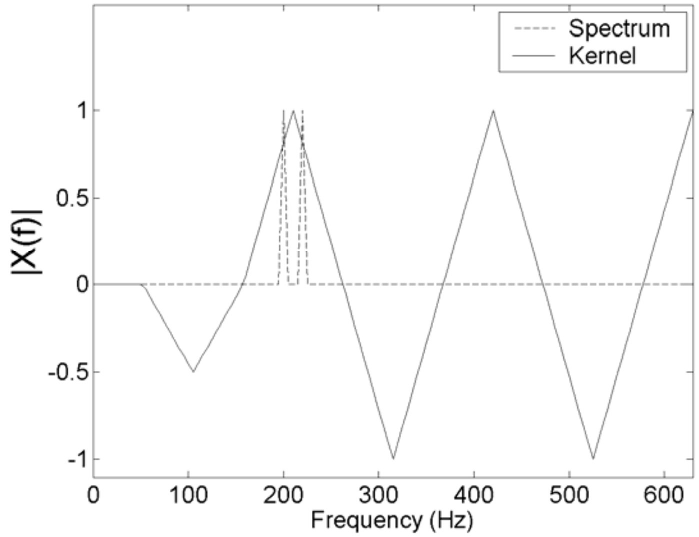

Figure 3-3. Necessity of strictly convex kernels.

Object 3-1. Beating tones (WAV file, 32 KB)

The triangular kernel approach was abandoned because it was found that the components of the kernel must be *strictly concave* (i.e., must have a continuous second derivative) at their maxima. The following example will illustrate why this is necessary. Suppose we have a signal with components at 200 and 220 Hz, as shown in Figure 3-3 (Object 3-1). This signal is perceived as a loudness-varying tone with a pitch of 210 Hz, phenomena known as *beating*. However, the triangular kernel produces the same score for each candidate between 200 and 220 Hz. This is easy to see by slightly stretching or compressing the kernel such that its first positive peak is within that range. Such stretching or compression would cause an increment on the weight of one of the components and a decrement of the same amount on the other, keeping the score constant.

Therefore, the triangle was discarded and concatenations of truncated squarings, Gaussians, and cosines were explored. The squaring function was truncated at its fixed point, and

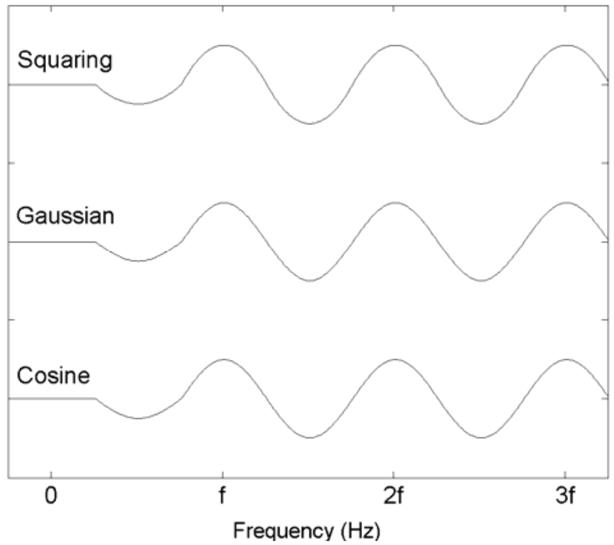

Figure 3-4. Kernels formed from concatenations of truncated squarings, Gaussians, and cosines.

the Gaussian and the cosine functions were truncated at their inflection points. The Gaussian was truncated at the inflection points to ensure that the concatenation of positive and negative Gaussians have a continuous second derivative. The same can be said about the cosine, but furthermore, the concatenation of positive and negative cosine lobes produces a cosine, which has all order derivatives.

Concatenations of these three functions, stretched or compressed to form the desired pattern of maxima at multiples of the candidate pitch, are illustrated in Figure 3-4. Although informal tests showed no significant differences in pitch estimation performance among the three, the cosine was preferred because of its simplicity. Notice also that this kernel is the one used by the AC and CEP pitch estimators (see Chapter 2).

3.3 Warping of the Spectrum

As mentioned in Chapter 2, the use of the logarithm of the spectrum in an integral transform is inconvenient because there may be regions of the spectrum with no energy, which

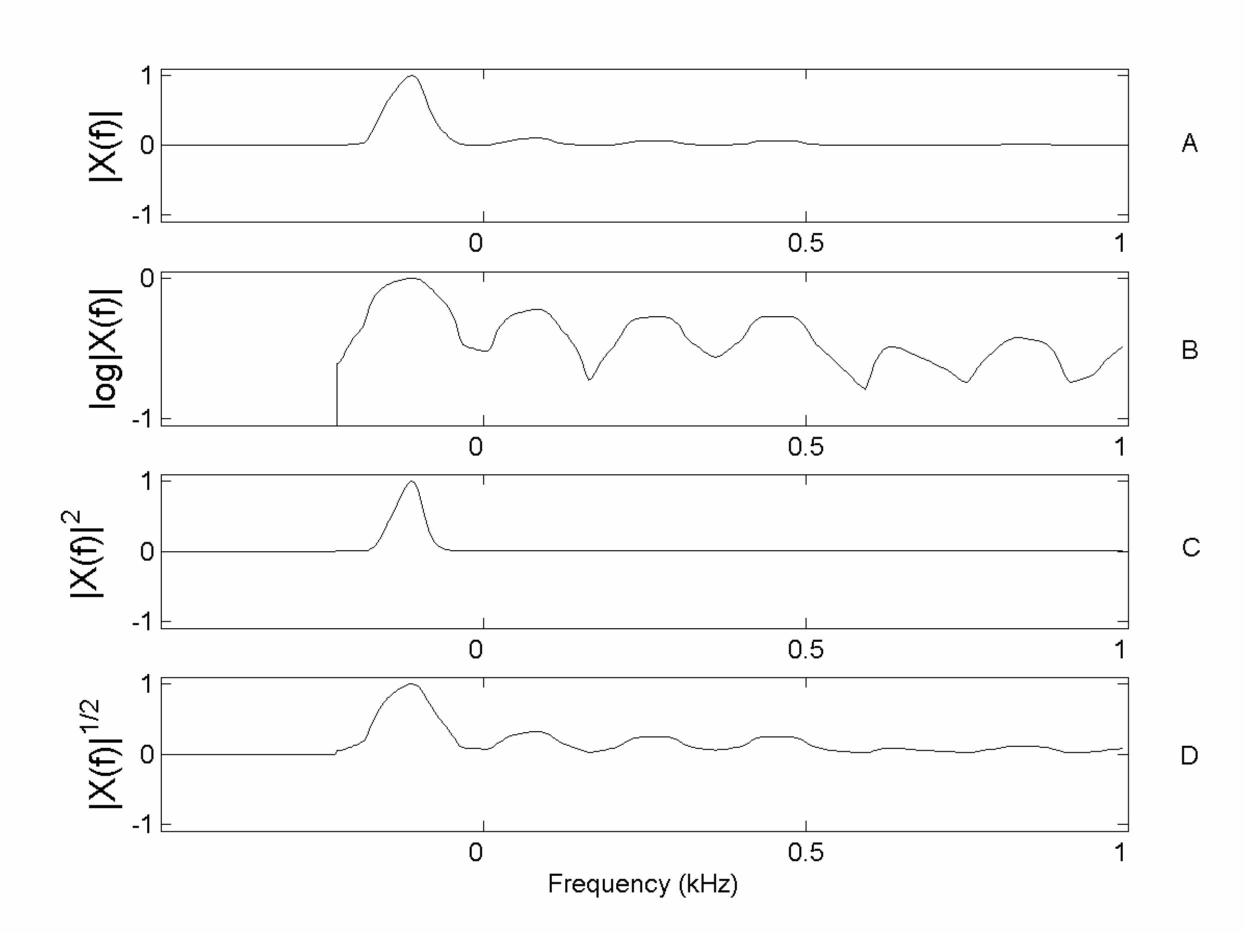

Figure 3-5. Warping of the spectrum.

would prevent the evaluation of the integral, since the logarithm of zero is minus infinity. But even if there is some small energy in those regions, the large absolute value of the logarithm could make the effect of these low energy regions on the integral larger than the effect of the regions with the most energy, which is certainly inconvenient.

To avoid this situation, the use of the logarithm of the spectrum was discarded and other commonly used functions were explored: square, identity, and square-root. Figure 3-5 shows how these functions warp the spectrum of the vowel /*u*/ used in Chapter 2. As mentioned earlier, this spectrum has two particularities: it has a missing fundamental, and it has a salient second harmonic. The missing fundamental is evident in panel B, which shows that the logarithm of the spectrum in the region of 190 Hz is minus infinity. The salient second harmonic at 380 Hz shows up clearly in the other three panels, but especially in panel C, where the spectrum has been squared. Panel D shows the square-root of the spectrum, which neither overemphasizes the missing fundamental (as the logarithm does) nor the salient second harmonic (as the square does).

We believe the square-root warping of the spectrum is more convenient for three reasons. First, it matches better the response of the auditory system to amplitude, which is close to a power function with an exponent in the range 0.4-0.6 (see Chapter 2); second, it allows for a weighting of the harmonics proportional to their amplitude, as we will show in the next section; and third, it produces better pitch estimates, as found tests presented later.

3.4 Weighting of the Harmonics

To avoid the subharmonics problem presented in Chapter 2, a decaying weighting factor was applied to the harmonics. The types of decays explored were exponential and harmonic. For exponential decays, a weight of r^{k-1} was applied to the k-th harmonic ($k = 1, 2, \ldots, n$, and $r = 0.9, 0.7, 0.5$) through the multiplication of the kernel by the envelope $r^{f'/f-1}$, as shown in Figure 3-6. For harmonic decays, a weight of $1/k^{p}$ was applied to the k-th harmonic ($k = 1, 2, \ldots, n$, and $p = 1/2, 1, 2$) through the multiplication of the kernel by the envelope $(f/f')^{p}$, as shown in Figure 3-6. In informal tests, the best results were obtained using harmonic decays with $p = 1/2$, which matches the decay of the square-root of the average spectrum of vowels (see Chapter 2). In other words, better pitch estimates were obtained when computing the inner product (IP) of the square-root of the input spectrum and the square-root of the expected spectrum, than when computing the IP's over the raw spectra.

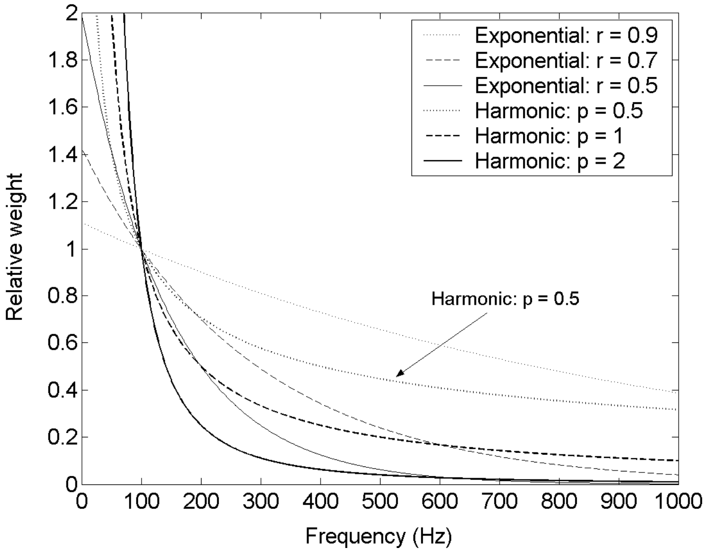

Figure 3-6. Weighting of the harmonics.

One explanation for this is that when the input spectrum matches its corresponding template (i.e., the expected spectrum for that pitch), the use of the square-root of the spectra in the IP gives to each harmonic a weight proportional to its amplitude. For example, if the input spectrum has the expected shape for a vowel, i.e., the amplitude of the harmonics decay as 1, 1/2, 1/3, etc., then their square root decays as 1, 1/√2, 1/√3, etc. Since the terms in the sum of the IP are the squares of these values (i.e., 1, 1/2, 1/3, etc.), then the relative contribution of each harmonic is proportional to its amplitude. Conversely, if we compute the IP over the raw spectra, the terms of the sum will be 1, 1/4, 1/9, etc., which are not proportional to the amplitude of the components, but to their square. This would make the contribution of the strongest harmonics too large and the contribution of the weakest too small. The situation would be even worse if we would compute the IP over the energy of the spectrum (i.e., its square). The expected energy of the harmonics for a vowel follows the pattern 1, 1/4, 1/16, etc., and computing the IP of the energy of the harmonics with itself produces the terms 1, 1/16, 1/256, etc, which gives too much weight to the first harmonic and almost no weight to the other harmonics.

In the ideal case in which there is a perfect match between the input and the template, any of the previous types of warping would produce the same result: a normalized inner product (NIP) equal to 1. However, the likelihood of a perfect match is low, and the warping may play a big role in the determination of the best match, as we found in informal tests, which show that the use of the square-root of the spectrum produces better pitch estimates.

3.5 Number of Harmonics

An important issue is the number of harmonics to be used to analyze the pitch. HPS, SHS, SHR, and HS use a fixed finite number of harmonics, and CEP and AC use all the available harmonics (i.e., as many as the sampling frequency allows). In informal tests the best results were obtained when using as many harmonics as available, although it was found that going beyond 3.3 kHz for speech and 5 kHz for musical instruments did not improve the results significantly. Thus, to reduce computational cost it is reasonable to set these limits.

3.6 Warping of the Frequency Scale

As mentioned in Section 3.4, if the input matches perfectly any of the templates, their NIP will be equal to 1, regardless of the type of warping used on the spectrum. The same applies to the frequency scale. However, since a perfect match will rarely occur, a warping of the frequency scale may play a role in determining the best match.

For the purposes of computing the integral of a function, we can think of a warping of the scale as the process of sampling the function more finely in some regions than others, effectively giving more emphasis to the more finely sampled regions. In our case, since we are computing an inner product to estimate pitch, it makes sense to sample the spectrum more finely in the region that contributes the most to the determination of pitch. It seems reasonable to assume that this region is the one with the most harmonic energy. In the case of speech, and assuming that the amplitude of the harmonics decays inversely proportional to frequency, it seems reasonable to sample the spectrum more finely in the neighborhood of the fundamental and decrease the granularity as we move up in frequency, following the expected $1/f$ pattern for the amplitude of the harmonics. A decrease in granularity should also be performed below the fundamental because no harmonic energy is expected below it. However, the determination of the frequency at which this decrease should begin is non-trivial, since we do not know a-priori the fundamental frequency of the incoming sound (that is precisely what we wish to determine).

As we did for the selection of the warping of the amplitude of the spectrum, we appeal to the auditory system and borrow the frequency scale it seems to use: the ERB scale (see Chapter 1). Therefore, to compute the similarity between the input spectrum and the template, we sample both of them uniformly in the ERB scale, whose formula is given in Equation 1-8. This scale has several of the characteristics we desire (see Figure 1-9): it has a logarithmic behavior as f increases, tends toward a constant as f decreases, and the frequency at which the transition occurs (229 Hz) is close to the mean fundamental frequency of speech, at least for females (Bagshaw, 1994; Wang and Lin, 2004; Schwartz and Purves, 2004). It does not produce a decrease of granularity as f approaches zero, but at least does not increase without bound either, as a pure logarithmic scale does.

The convenience of the use of the ERB scale for pitch estimation over the Hertz and logarithmic scales was confirmed in informal tests, since better results were obtained when using the ERB scale. Two other common psychoacoustic scales, the Mel and Bark scales, were also explored, but they produced worse results than the ERB scale.

3.7 Window Type and Size

Along this chapter we have been mentioning our wish to obtain a perfect match (i.e., NIP equal to 1) between the input spectrum and the template corresponding to the pitch of the input. This section deals with the feasibility of achieving such goal.

First of all, since the input is non-negative but the template has negative regions, a perfect match is impossible. One solution would be to set the negative part of the template to zero, but this would leave us without the useful property that the negative weights have: the production of low scores for noisy signals (see Section 2.3). Instead, the solution we adopt is to preserve the negative weights, but ignore them when computing the norm of the template. In other words, we normalize the kernel using only the norm of its positive part

$$K^+(f) = \max(0, K(f)) \,. \tag{3-6}$$

Hereafter, we will refer to this normalization as K^+-normalization.

To obtain a K^+-normalized inner product (K^+-NIP) close to 1, we must direct our efforts to make the shape of the spectral peaks match the shape of the positive cosine lobe used as base element of the template, and also to force the template have a value of zero in the negative part of the cosine. Since the shape of the spectral peaks is the same for all peaks, it is enough to concentrate our efforts on one of them, and for simplicity we will do it for the peak at zero frequency.

The shape of the spectral peaks is determined by the type of window used to examine the signal. The most straightforward window is the rectangular window, which literally acts like a window: it allows seeing the signal inside the window but not outside it. More formally, the rectangular window multiplies the signal by a rectangular function of the form

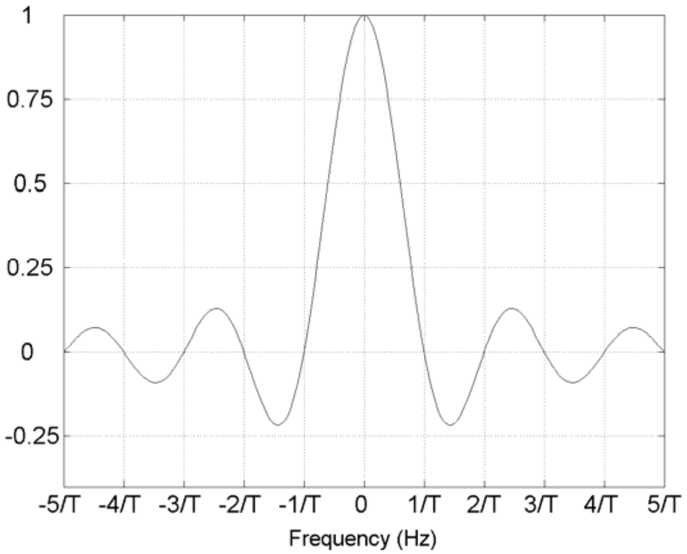

Figure 3-7. Fourier transform of rectangular window.

$$\Pi_T(t) = \begin{cases} 1/T & ,\text{if } |t| < T/2 \\ 0 & ,\text{otherwise,} \end{cases} \tag{3-7}$$

where T is the window size.

If a rectangular window is used to extract a segment of a sinusoid of frequency f Hz to compute its Fourier transform, the support of this transform will not be concentrated at a single point but will be smeared in the neighborhood of f. This effect is shown in Figure 3-7 for $f = 0$, in other words, the figure shows the Fourier transform of $\Pi_T(t)$. This transform can be written as sinc(Tf'), where the sinc function is defined as

$$\text{sinc}(\phi) = \frac{\sin(\pi\phi)}{\pi\phi}. \tag{3-8}$$

This function consists of a main lobe centered at zero and small side lobes that extend towards both sides of zero. For any other value of f, its Fourier transform is just a shifted version of this function, centered at f.

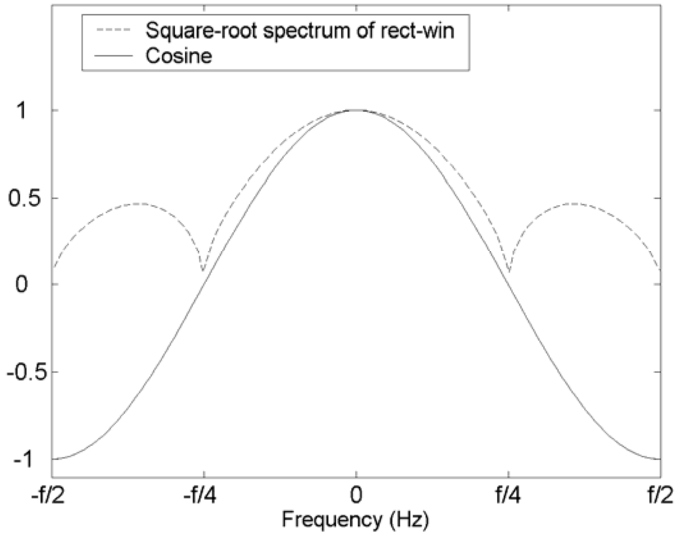

Figure 3-8. Cosine lobe and square-root of the spectrum of rectangular window.

Since the height of the side lobes is small compared to the height of the main lobe, the most obvious approach to try to maximize the match between the input and the template is to match the width of the main lobe, $2/T$, to the width of the cosine lobe, $f/2$, and solve for the free variable T. This produces an "optimal" window size, hereafter denoted T^*, equal to $T = 4/f$. Figure 3-8 shows the square-root of the spectrum of a rectangular window of size $T = T^* = 4/f$ and a cosine with period f (i.e., the template used to recognize a pitch of f Hz). The K^+-NIP of the main lobe of the spectrum and the cosine positive lobe (i.e., from $-f/4$ to $f/4$) sampled at 128 equidistant points is 0.9925, which seems satisfactorily high. However, the K^+-NIP computed over the whole period of the cosine (i.e., from $–f/2$ to $f/2$) sampled at 128 equidistant points is only 0.5236, which is not very high. This low K^+-NIP is caused by the relatively large side lobes, which reach a height of almost 0.5.

A window with much smaller side lobes is the Hann window. The shorter side lobes are achieved by attenuating the time-domain window down towards zero at the edges[3]. The formula for this window is

$$h_T(t) = \frac{1}{T}\left[1 + \cos\left(\frac{2\pi t}{T}\right)\right], \tag{3-9}$$

where *T* is the window size (i.e., the size of its support). This window is simply one period of a raised cosine centered at zero, as illustrated in Figure 3-9.

The Fourier transform of a Hann window of size *T* is

$$H_T(f) = \operatorname{sinc}(Tf) + \frac{1}{2}\operatorname{sinc}(Tf - 1) + \frac{1}{2}\operatorname{sinc}(Tf + 1), \tag{3-10}$$

a sum of three sinc functions, as illustrated in Figure 3-10. The width of the main lobe of this transform is 4/*T*, twice as large as the main lobe of the spectrum of the rectangular window.

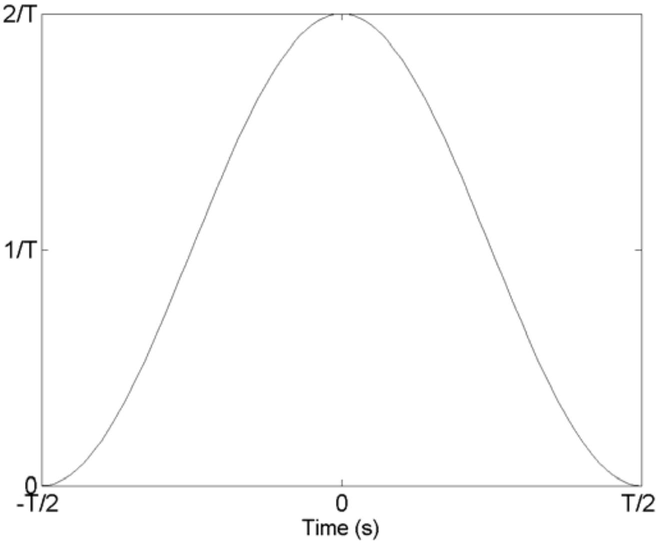

Figure 3-9. Hann window.

[3] This time-frequency relation may not be obvious at first sight, but it can be shown using Fourier analysis.

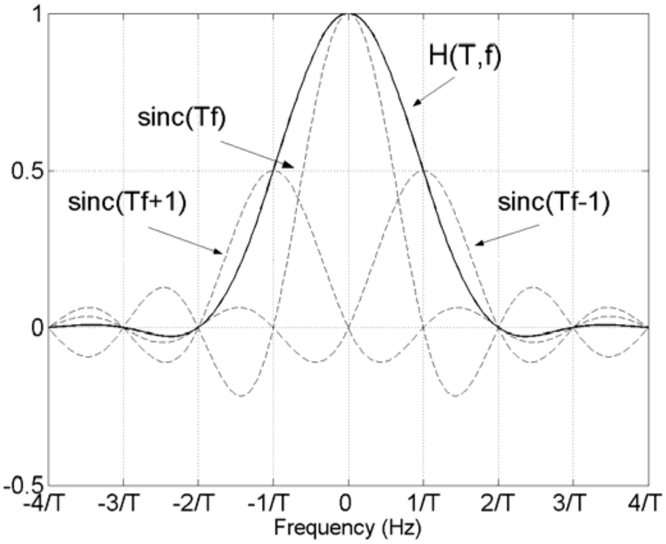

Figure 3-10. Fourier transform of the Hann window. The FT of the Hann window consists of a sum of three sinc functions.

Equalizing this width to the width of the cosine lobe, *f*/2, and solving for *T*, we obtain an optimal window size of $T^{*} = 8/f$.

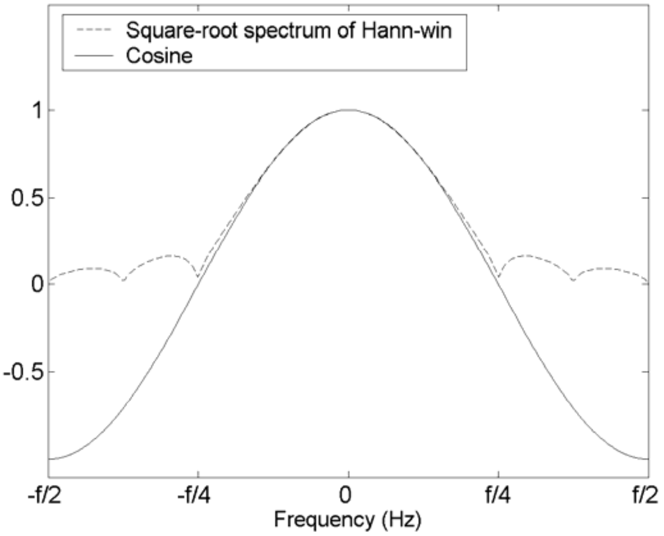

Figure 3-11. Cosine lobe and square-root of the spectrum of Hann window.

Figure 3-11 shows the square-root of the spectrum of a Hann window of size $T = T^* = 8/f$ and a cosine with period f. The similarity between the main lobe and the positive lobe of the cosine is remarkable. Using Equations 3-8 and 3-10 it can be shown that they match at 5 points: 0, +/- $f/8$, and +/- $f/4$, with values $\cos(0) = 1$, $\cos(\pi/4) = 1/\sqrt{2}$, and $\cos(\pi/2) = 0$, respectively. The K^+-NIP of the main lobe of the spectrum and the positive part of the cosine sampled at 128 equidistant points is 0.9996, and the K^+-NIP computed over the whole period of the cosine sampled at 256 equidistant points is 0.8896, much larger than the one obtained with the rectangular window.

The same approach can be used to obtain the optimal window size for other window types. For the most common window types used in signal processing, it can be shown that the width of the main lobe is $2k/T$, where the parameter k depends on the window type (see Oppenheim, Schafer, and Buck, 1999) and is tabulated in Table 3-1. For these windows, the optimal window

Table 3-1. Common windows used in signal processing*

		K^+-NIP	
Window type	k	Positive lobe	Whole period
Bartlett	2	0.9984	0.7959
Bartlett-Hann	2	0.9995	0.8820
Blackman	3	0.9899	0.9570
Blackman-Harris	4	0.9738	0.9689
Bohman	3	0.9926	0.9474
Flat top	5	0.9896	0.9726
Gauss	3.14	0.9633	0.8744
Hamming	2	0.9993	0.9265
Hann	2	0.9996	0.8896
Nuttall	4	0.9718	0.9682
Parzen	4	0.9627	0.9257
Rectangular	1	0.9925	0.5236
Triangular	2	0.9980	0.8820

* The K^+-NIP values were computed using 128 equidistant samples for the positive lobe and 256 equidistant samples for the whole period.

size to analyze a signal with pitch *f* Hz can be obtained by equalizing 2*k*/*T* to the width of the cosine lobe, *f*/2, to produce $T^* = T = 4k/f$.

Table 3-1 also shows the K^+-NIPs between the square-root of the spectrum and the cosine computed over the positive lobe of the cosine (from -*f*/4 to *f*/4) and over the whole period of the cosine (from -*f*/2 to *f*/2). The window that produces the largest K^+-NIP over the whole period is the flat-top window. However, its size is so large compared to other windows that the increase in K^+-NIP is probably not worth the increase in computational cost; similar results are obtained with the Blackman-Harris window, which is 4/5 its size. If computational cost is a serious issue, a good compromise is offered by the Hamming window, which requires half the size of the Blackman-Harris window, and produces a K^+-NIP of about 0.93. This K^+-NIP is larger than the one produced by the Hann window, with no increased computational cost (*k*=2 in both cases). However, since the difference in performance between them is not large, we prefer the analytically simpler Hann window.

3.8 SWIPE

Putting all the previous sections together, the SWIPE estimate of the pitch at time *t* can be formulated as

$$p(t) = \arg\max_f \frac{\displaystyle\int_0^{\mathrm{ERBs}(f_{\max})} \frac{1}{\eta(\varepsilon)^{1/2}}\ K(f,\eta(\varepsilon))\ |X(t,f,\eta(\varepsilon))|^{1/2}\ d\varepsilon}{\left(\displaystyle\int_0^{\mathrm{ERBs}(f_{\max})} \frac{1}{\eta(\varepsilon)}\ [K^+(f,\eta(\varepsilon))]^2\ d\varepsilon\right)^{1/2} \left(\displaystyle\int_0^{\mathrm{ERBs}(f_{\max})} |X(t,f,\eta(\varepsilon))|\ d\varepsilon\right)^{1/2}}, \qquad \text{(3-11)}$$

where

$$K(f,f') = \begin{cases} \cos(2\pi f'/f) & ,\text{if } 3/4 < f'/f < n(f)+1/4, \\ \frac{1}{2}\cos(2\pi f'/f) & ,\text{if } 1/4 < f'/f < 3/4 \text{ or } n(f)+1/4 < f'/f < n(f)+3/4, \\ 0 & ,\text{otherwise,} \end{cases} \qquad \text{(3-12)}$$

$$X(t,f,f') = \int_{-\infty}^{\infty} w_{4k/f}(t'-t)\; x(t')\; e^{-j2\pi f't'} dt' , \qquad (3\text{-}13)$$

ε is frequency in ERBs, $\eta(\cdot)$ converts frequency from ERBs into Hertz, ERBs($\cdot$) converts frequency from Hertz into ERBs, $K^{+}(\cdot)$ is the positive part of $K(\cdot)$ {i.e., max[0, $K(\cdot)$]}, $f_{\max}$ is the maximum frequency to be used (typically the Nyquist frequency, although 5 kHz is enough for most applications), $n(f) = \lfloor f_{\max} / f - 3/4 \rfloor$, and $w_{4k/f}(t)$ is one of the window functions in Table 3-1, with size $4k/f$. The kernel corresponding to a candidate with frequency 190 Hz is shown in Figure 3-12. Panel A shows the kernel in the Hertz scale and Panel B in the ERB scale, the scale used to compute the integral.

Although the initial approach of measuring a smooth average peak to valley distance has been used everywhere in this chapter, we can make a more precise description of the algorithm.

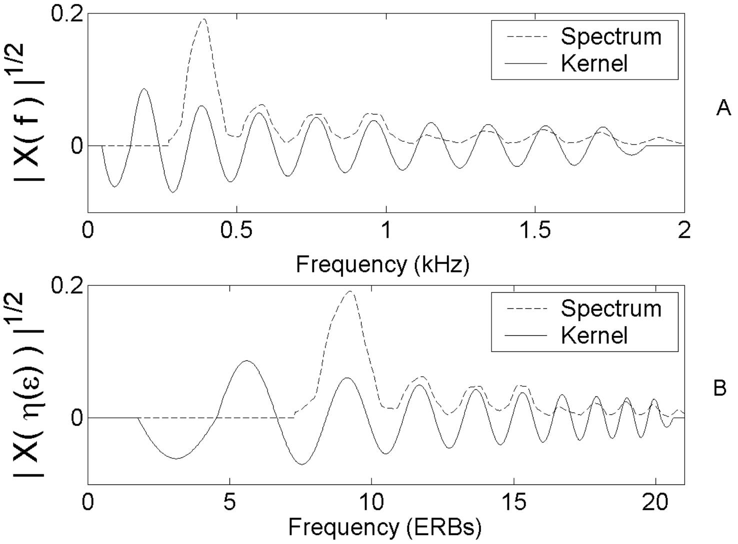

Figure 3-12. SWIPE kernel. A) The SWIPE kernel consists of a cosine that decays as $1/f$, with a truncated DC lobe and halved first and last negative lobes. B) SWIPE kernel in the ERB scale.

It can be described as the computation of the similarity between the square-root of the spectrum of the signal and the square-root of the spectrum of a sawtooth waveform, using a pitch-dependant optimal window size. This description gave rise to the name Sawtooth-Waveform Inspired Pitch Estimator (SWIPE).

3.9 SWIPE′

So far in this chapter we have concentrated our efforts on maximizing the similarity between the input and the desired template, but we have not done anything explicitly to reduce the similarity between the input and the other templates, which will be the goal of this section.

The first fact we want to mention is that most of the mistakes that pitch estimators make, including SWIPE, are not random: they consist of estimations of the pitch as multiples or submultiples of the pitch. Therefore, a good source of error to attack is the score (pitch strength) of these candidates.

A good feature to reduce supraharmonic errors is to use negative weights between harmonics. When analyzing a pitch candidate, if there is energy between any pair of consecutive harmonics of the candidate, this suggests that the pitch, if any, is a lower candidate. This idea is implemented by the negative weights, which reduce the score of the candidate if there is any energy between its harmonics. This feature is used by algorithms like SHR, AC, CEP, and SWIPE.

The effect of negative weights on supraharmonics of the pitch is illustrated in Figure 3-13A. It shows the spectrum of a signal with fundamental at 100 Hz and all its harmonics at the same amplitude (vertical lines). (Only harmonics up to 1 kHz are shown, but the signal contains harmonics up to 5 kHz.) The components are shown as lines to facilitate visualization, but in general they will be wider, with a width that depends on the window size.

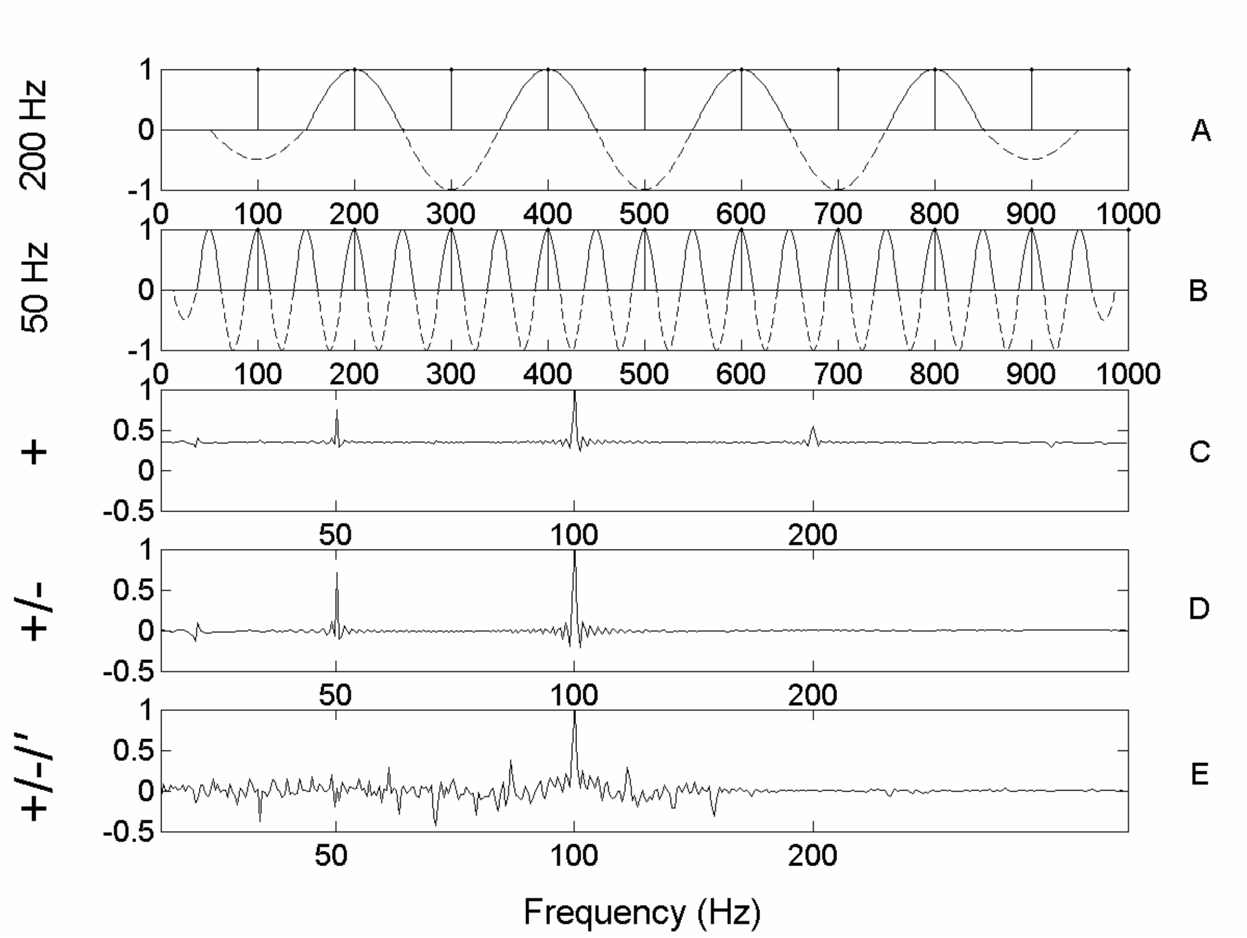

Figure 3-13. Most common pitch estimation errors. A) Harmonic signal with 100 Hz fundamental frequency and all the harmonics at the same amplitude, and 200 Hz kernel with positive (continuous lines) and negative (dashed lines) cosine lobes. B) Same signal and 50 Hz kernel. C) Scores using only positive cosine lobes (exhibits peaks at sub and supraharmonics). D) Scores using both positive and negative cosine lobes (exhibits peaks at subharmonics). E) Scores using both positive and negative cosine lobes at the first and prime harmonics (exhibits a major peaks only at the fundamental)

Panel A also shows the positive cosine lobes (continuous curves) used to recognize a pitch of 200 Hz and the negative cosine lobes that reside in between (dashed curves). The positive cosine lobes at the harmonics of 200 Hz produce a positive contribution towards the score of the 200 Hz candidate, but the negative cosine lobes at the odd multiples of 100 Hz cancel out this contribution. Panel C shows the score for each pitch candidate using as kernel only the positive cosine lobes, whereas Panel D shows the scores using both the positive and the negative cosine lobes. The effect on the 200 Hz peak is definite: it has disappeared. The same effect is obtained for higher order multiples of 100 Hz (not shown in the figure).

To reduce subharmonic errors, two techniques were presented in Chapter 2: the use of a decaying weighting factor for the harmonics, and the use of a bias to penalize the selection of low frequency candidates. The former is used by SHS and SWIPE, and the latter by AC. Although these techniques have an effect in reducing the score of subharmonics, significant peaks are nevertheless present at submultiples of the pitch, as shown in Figure 3-13D.

To further reduce the height of the peaks at subharmonics of the pitch we propose to remove from the kernel the lobes located at non-prime harmonics, except the lobe at the first harmonic. Figure 3-13B helps to show the intuition behind this idea. This figure shows the same spectrum as in Figure 3-13A and the kernel corresponding to the 50 Hz candidate. This kernel has positive lobes at each multiple of 50 Hz and therefore at each multiple of 100 Hz, producing a high score for the 50 Hz candidate, as shown in Panel D. Notice that this candidate gets all of its credit from its 2nd, 4th, 6th, etc., harmonics, i.e., 100 Hz, 200 Hz, 300 Hz, etc., frequencies that suggest a fundamental frequency (and pitch) of 100 Hz. The same situation occurs with the candidate at 33 Hz (kernel not shown), but in this case its credit comes from its 3rd, 6th, 9th, etc., harmonics.

If we use only the first and prime lobes of the kernel, the candidates at subharmonics of 100 Hz would get credit only from their harmonic at 100 Hz, but not from any other. In general, it can be shown that with this approach, no candidate below 100 Hz can get credit from more than one of the harmonics of 100 Hz. In other words, if there is a match between one of the prime harmonics of this candidate and a harmonic of 100 Hz, no other prime harmonic of the candidate can match another harmonic of 100 Hz, and therefore the score of all the candidates below 100 Hz has to be low compared to the score of the 100 Hz candidate. This effect is evident in Figure 3-13E, which shows the scores of the pitch candidates when using only their first and prime harmonics. Certainly, there are peaks below 100 Hz, but they are relatively small compared to the peak at 100 Hz. Contrast this with Panels C and D, where the score of 50 Hz is relatively high, and therefore the risk of selecting this candidate is high.

An extra step needs to be done to avoid bias in the scores. Remember from the beginning of this chapter that the central idea of SWIPE was to compute the average peak-to valley distance at harmonic locations in the spectrum. When computing this average for a single peak, the weight of the peak was twice as large as the weight of its valleys, as expressed in Equation 3-1. Since the global average is the average of this equation over all the peaks, and since each valley is associated to two peaks too, the weight of the valleys, except the first and the last ones, was the same as the weight of the peaks, as expressed in Equation 3-2. However, if we use only the first and prime harmonics, the weight of the valleys will not be necessarily -1, but will depend on whether the valleys are between the first or prime harmonics. The only valleys with a weight of -1 will be the valley between the first and second harmonics, and the valley between the second and third harmonics; all the other valleys will have a weight of -1/2, before applying the decaying weighting factor, of course.

This variation of SWIPE in which only the first and prime harmonics are used to estimate the pitch will be denominated SWIPE′ (read SWIPE prime). Its kernel is defined as

$$K(f,f') = \sum_{i\in\{1\}\cup P} K_i(f,f'), \tag{3-14}$$

where P is the set of prime numbers, and

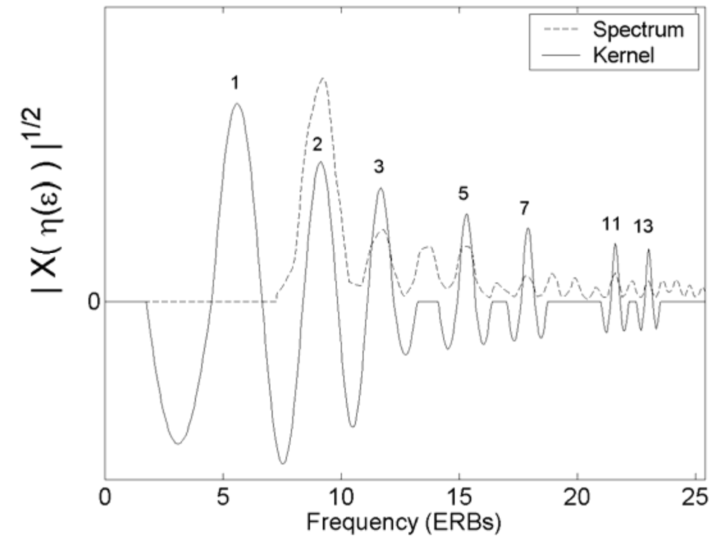

Figure 3-14. SWIPE′ kernel. Similar to the SWIPE kernel but includes only the first and prime harmonics.

$$K_i(f,f') = \begin{cases} \cos(2\pi f'/f) & ,\text{if } \ |f'/f - i| < 1/4, \\ \frac{1}{2}\cos(2\pi f'/f) & ,\text{if } \ 1/4 < |f'/f - i| < 3/4, \\ 0 & ,\text{otherwise.} \end{cases} \tag{3-15}$$

Notice that the SWIPE kernel can also be written as in Equation 3-14, by including all the harmonics in the sum. The SWIPE′ kernel corresponding to a pitch candidate of 190 Hz (5.6 ERBs) is shown in Figure 3-14. The numbers on top of the peaks show the harmonic number they correspond to.

3.9.1 Pitch Strength of a Sawtooth Waveform

Since the template used by SWIPE′ has peaks only at the first and prime harmonics, a perfect match between the template and the spectrum of a sawtooth waveform is impossible (unless $f_{\max}$ is so small relative to the pitch that the template contains no more than three

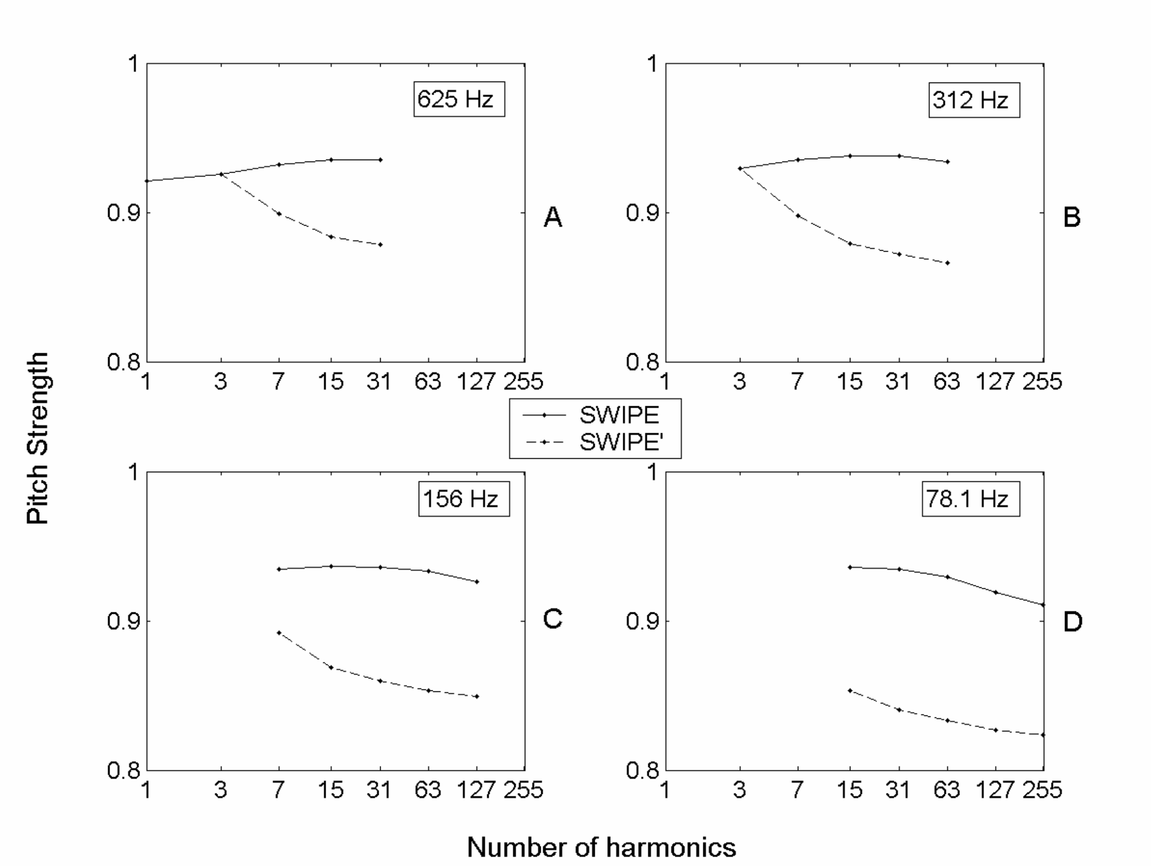

Figure 3-15. Pitch strength of sawtooth waveform. A) 625 Hz. B) 312 Hz. C) 156 Hz. D) 78.1 Hz.

harmonics). Therefore, it would be interesting to analyze the K^{+}-NIP between the spectrum and the template as a function of the number of harmonics. Figure 3-15 shows the pitch strength (K^{+}-NIP) obtained using SWIPE and SWIPE′ for different pitches and different number of harmonics. The pitches shown are 625, 312, 156, and 78.1 Hz. They were chosen because their optimal window sizes are powers of two for the sampling rates used: 2.5, 5, 10, 20, and 40 kHz. In each case, $f_{\max}$ was set to the Nyquist frequency.

The pitch strength estimates produced by SWIPE are larger than the ones produced by SWIPE′, except when the number of harmonics is less than four, in which case both algorithms use all the harmonics. The pitch strength estimates produced by SWIPE in Figure 3-15 have a mean of 0.93 and a variance of 5.1×10^{-5}. This mean is significantly larger than the K^+-NIP reported in Table 3-1 for the Hann window. The reason of the mismatch is that the granularity used to produce the data in Table 3-1 and the data in Figure 3-15 is different. The K^+-NIP values in Table 3-1 are based on a sampling of 128 points per spectral lobe, while the data in Figure 3-15 is based on a sampling of 10 points per ERB, which depending on the pitch and the harmonic being sampled, may correspond to a range of about 0 to 40 points per spectral lobe.

On the other hand, the mean of the pitch strength estimates produced by SWIPE′ is 0.87 and the variance is 1.0×10^{-3}. The smaller mean is expected since the template of SWIPE′ includes only the first and prime harmonics, while a sawtooth waveform has energy at each of its harmonics. The larger variance is also expected since the prime numbers become sparser as they become larger, causing a reduction in the similarity of the template and the spectrum of the sawtooth waveform as the number of harmonics increases.

It would be useful to have a lower bound for the pitch strength estimates produced by SWIPE′, but an analytical formulation for it is intractable. However, the data in Figure 3-15, which is representative of a wide range of pitches and number of harmonics, suggests that the pitch strength produced by SWIPE′ for a sawtooth waveform does not go below 0.8.

3.10 Reducing Computational Cost

3.10.1 Reducing the Number of Fourier Transforms

The computation of Fourier transforms is one of the most computationally expensive operations of SWIPE and SWIPE′. Therefore, to reduce computational cost it is important to reduce the number of Fourier transforms. There are two strategies to achieve this: to reduce the window overlap and to share Fourier transforms among several candidates.

3.10.1.1 Reducing window overlap

The most common windows used in signal processing are the ones that are attenuated towards zero at their edges (e.g., Hann and Hamming windows). A disadvantage of this attenuation is that it is possible to overlook short events if these events are located at the edges of the windows. To avoid this situation, it is common to use overlapping windows, which increases the coverage of the signal, at the cost of an increase in computation. However, after a certain point, overlapping windows start to produce redundancy in the analysis, without adding any significant benefit. The goal of this section is to propose a schema obtain a good balance between signal coverage and computational cost.

As mentioned in Section 1.1.4, depending on frequency, a minimum of two to four cycles are necessary to perceive the pitch of a pure tone. Based on the similarity of the data used to arrive to this conclusion and data obtained using musical instruments, it is reasonable to assume that these results are applicable to more general waveforms, in particular, to sawtooth waveforms. To avoid the interaction between the number of cycles and pitch, for purposes of the algorithm, we set the minimum number of cycles necessary to determine pitch to four, the maximum among the minimum number of cycles required over all frequencies.

Since SWIPE and SWIPE′ are designed to produce maximum pitch strength for a sawtooth waveform[4] and zero pitch strength for a flat spectrum[5], a natural choice to decide whether a sound has pitch is to use as threshold half the pitch strength of a sawtooth waveform. (In Section 3.9.1 it was found that the pitch strength of a sawtooth waveform is about 0.93 for SWIPE and between 0.83 and 0.93 for SWIPE′.) To make these algorithms produce maximum pitch strength,

[4] In fact, SWIPE′ produces maximum pitch strength for sawtooth waveforms with the non-prime harmonics removed (except the first one), but we believe this type of signal is unlikely to occur in nature.

[5] The pitch strength of a flat spectrum is in fact negative because of the decaying kernel envelope.

a perfect match between the kernel and the spectrum of the signal is necessary, which requires that the window contains eight cycles of the sawtooth waveform, when using a Hann window. If the signal contains exactly eight cycles (i.e., if it is zero outside the window) and is shifted slightly with respect to the window, the pitch strength decreases, and it reaches a limit of zero when the signal gets completely out of the window. Although hard to show analytically, it is easy to show numerically that that the relation between the shift and pitch strength is linear. Therefore, if the window contains four or more cycles of the sawtooth waveform, the pitch strength is at least half the maximum attainable pitch strength (i.e., the one achieved when the window is full of the sawtooth waveform), and if the window contains less than four cycles of the sawtooth waveform, the pitch strength is less than half the maximum attainable pitch strength.

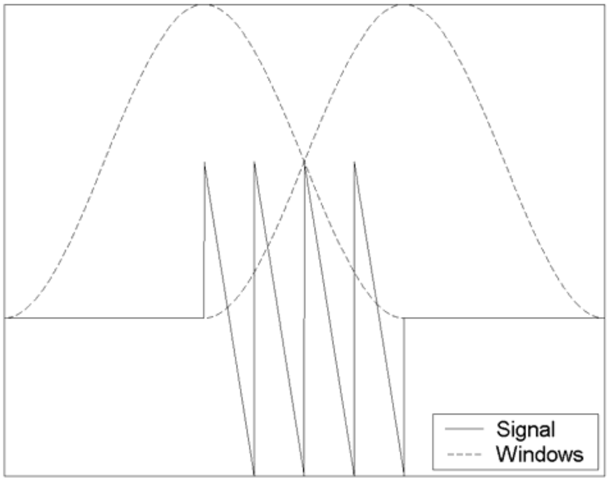

Figure 3-16. Windows overlapping.

Object 3-2. Four cycles of a 100 Hz sawtooth waveform (WAV file, 2 KB)

Therefore, if we determine the existence of pitch based on a pitch strength threshold equal to half the maximum attainable pitch strength, to determine as pitched a signal consisting of four cycles of a sawtooth waveform, we need to ensure that there exists at least one window whose coverage includes the whole signal. It is straightforward to show that to achieve this goal, we need to distribute the windows such that their separation in no larger than four cycles of the pitch period of the signal. In other words, the windows must overlap by at least 50%.

This situation is illustrated in Figure 3-16, which shows a signal consisting of four cycles of a sawtooth waveform (listen to Object 3-2) and two Hann windows centered at the beginning and the end of the signal. The windows are separated at a distance of four cycles, and the support of each of them overlaps with the whole signal, making it possible for each window to reach the pitch strength threshold. If the signal is slightly shifted in any direction, one of the windows will cover less than four periods, but the other will cover the four periods.

This would not be true if the separation of the windows is larger than four cycles. If the support of one of the windows overlaps completely with the signal but the separation of the windows is larger than four cycles, the other window will not cover the signal completely, and therefore a small shift of the signal towards the latter window would not necessarily put the whole signal inside the window, making it impossible for any of the windows to produce a pitch strength larger than the threshold.

3.12.1.2 Using only power-of-two window sizes

There is a problem with the optimal window size (O-WS) proposed in Section 3.7: each pitch candidate has its own, which means that a different STFT must be computed for each candidate. If we separate the candidates at a distance of 1/8 semitone over a range of 5 octaves (appropriate for music, for example), we will need to compute 8*12*5 = 480 STFTs for each pitch estimate. Not only that, for some WSs it may be inefficient to use an FFT (recall that the FFT is more efficient for windows sizes that are powers of two).

To alleviate this problem, we propose to substitute the O-WS with the power-of-two (P2) WS that produces the maximum K^{+}-NIP between the square-root of the main lobe of the spectrum and the cosine kernel. To find such a WS, it is convenient to have a closed-form formula for the K^{+}-NIP of these functions, but this involves integrating the product of a cosine and the square-root of the sum of three sinc functions, which is analytically intractable.

As an alternative, we approximate the square-root of the spectral lobe with an idealized spectral lobe (ISL) consisting of the function it approximates: a positive cosine lobe. Figure 3-17 shows a K^{+}-normalized cosine whose positive part has a width of $f/2$ (i.e., the cosine template used by an f Hz pitch candidate), and two normalized ISLs whose widths are half and twice the width of the positive part of the cosine. Since the cosine and the ISLs are symmetric around zero, the K^{+}-NIP can be computed using only the positive frequencies. Hence, the K^{+}-NIP

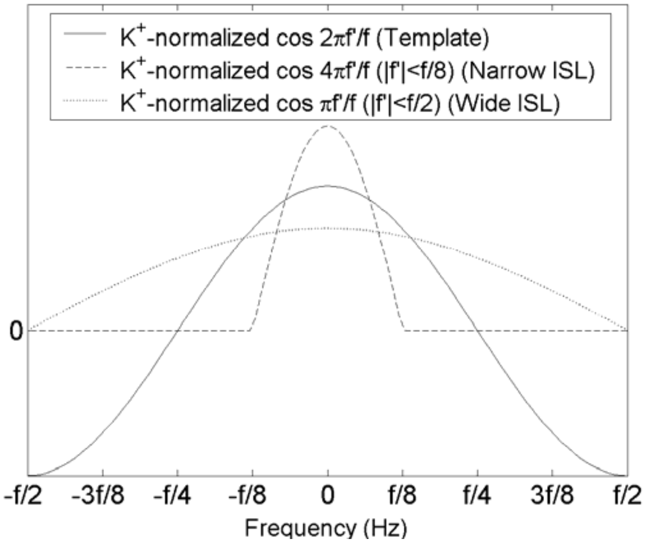

Figure 3-17. Idealized spectral lobes.

of the central positive lobe of a cosine with period *rf* (the ISL) and a cosine with period *f* (the template) can be computed as

$$P(r)=\frac{\int_0^{f/4r}\cos(2\pi rf'/f)\ \cos(2\pi f'/f)\ df'}{\left[\int_0^{f/4r}\cos^2(2\pi rf'/f)\ df'\right]^{1/2}\left[\int_0^{f/4}\cos^2(2\pi f'/f)\ df'\right]^{1/2}}$$

$$=\frac{\frac{1}{2}\int_0^{f/4r}\left\{\ \cos\left[2\pi(1+r)f'/f\right]+\cos\left[2\pi(1-r)f'/f\right]\ \right\}df'}{\left[f/8r\right]^{1/2}\ \left[f/8\right]^{1/2}}.$$

$$=\frac{2\sqrt{r}}{\pi}\left\{\frac{\sin\left[2\pi(1+r)f'/f\right]}{1+r}+\frac{\sin\left[2\pi(1-r)f'/f\right]}{1-r}\right\}\Bigg|_{f'=0}^{f'=f/4r}$$

$$=\frac{2\sqrt{r}}{\pi}\left\{\frac{\sin\left[\pi(1+r)/2r\right]}{1+r}+\frac{\sin\left[\pi(1-r)/2r\right]}{1-r}\right\}. \qquad (3\text{-}16)$$

It is convenient to transform the input of this function to a base-2 logarithmic scale, $\lambda = \log_2(r)$, and then redefine the function as

$$\Pi(\lambda)=\frac{2^{1+\lambda/2}}{\pi}\left\{\frac{\sin\left[(2^{-\lambda}+1)\pi/2\right]}{1+2^{\lambda}}+\frac{\sin\left[(2^{-\lambda}-1)\pi/2\right]}{1-2^{\lambda}}\right\}. \qquad (3\text{-}17)$$

Figure 3-18A shows $\Pi(\lambda)$ for λ between -1 and 1 (i.e., $r = 2^{\lambda}$ between 1/2 and 2). As λ departs from zero, $\Pi(\lambda)$ departs from 1, as expected. However, the distribution is not symmetric: a decrease in λ has a larger effect on $\Pi(\lambda)$ than an increase in λ. This make sense since a decrease in λ corresponds to a widening of the ISL, which puts part of it in the region where the cosine template is negative (see wider ISL in Figure 3-1), producing a large decrease in $\Pi(\lambda)$. On the other hand, narrowing the ISL keeps it in the positive region of the cosine template, producing a smaller decrease in $\Pi(\lambda)$.

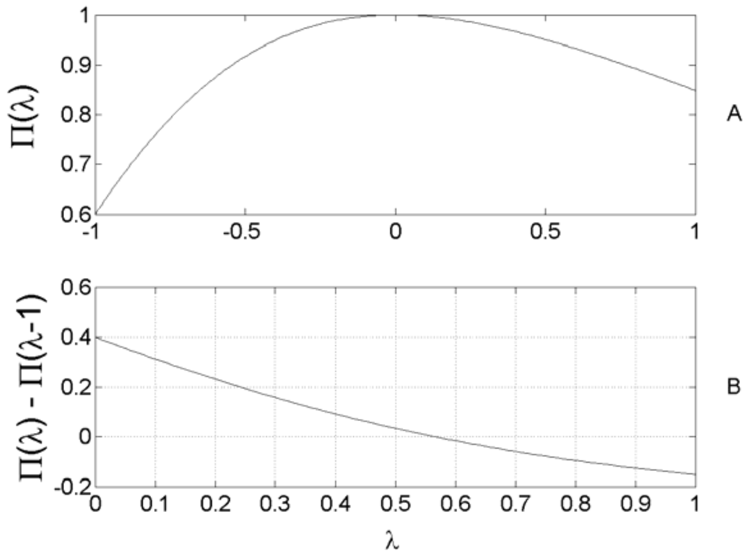

Figure 3-18. K^{+}-normalized inner product between template and idealized spectral lobes.

Figure 3-18A can be helpful in finding the P2-WS that produces the largest K^{+}-NIP between the ISL and the template. If the O-WS for the template is T^{*} seconds and the sampling rate is f_s, then the O-WS in samples is $N^{*} = T f_s$, which correspond to $\lambda = 0$ in the figure. Smaller λ's correspond to smaller WSs, and larger λ's correspond to larger WSs. In general, the WS in number of samples, denoted N, and λ, are related through the equation $N = 2^{\lambda} N^{*}$.

It is straightforward to show that the two λ's that correspond to the two closest P2-WSs to the optimal must be between -1 and 1, and not only that, their difference must be 1. Figure 3-18B shows the difference between $\Pi(\lambda)$ and $\Pi(\lambda - 1)$ as a function of λ, for λ between 0 and 1. From the figure we can infer that, for λ's between 0 and 0.56, we should use the larger P2-WS, and for λ between 0.56 and 1, we should use the smaller P2-WS. However, Figure 3-18B shows also that there is not much loss in the K^{+}-NIP by choosing 0.5 as threshold rather than 0.56. Therefore, to simplify the algorithm, we decided to set the threshold at 0.5. In other words, to determine the

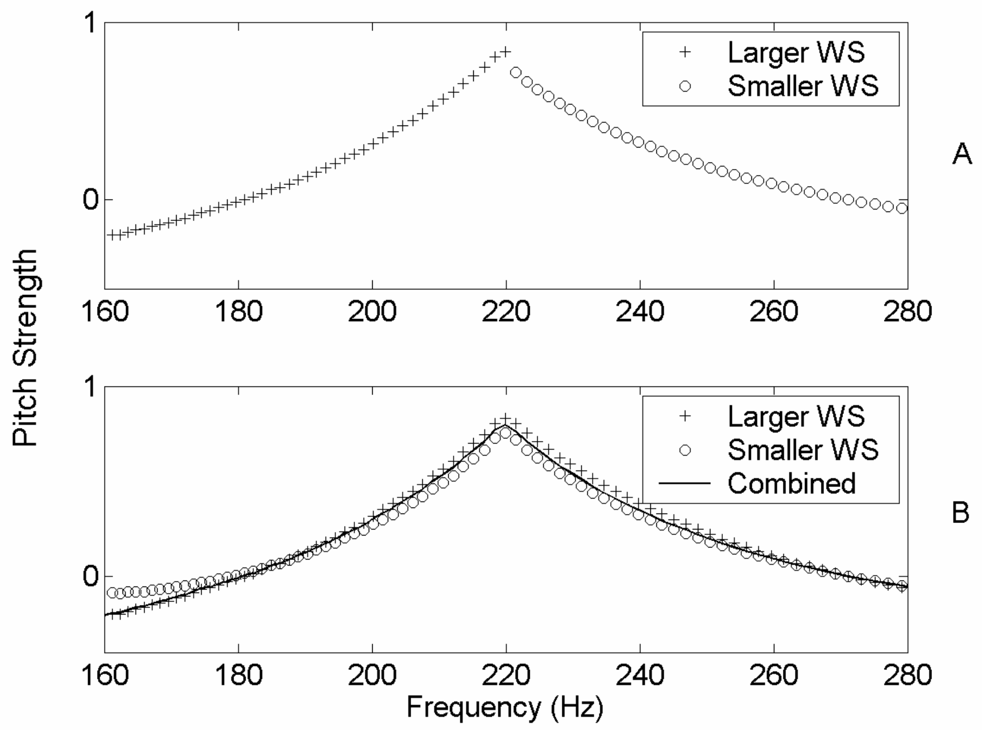

Figure 3-19. Individual and combined pitch strength curves.

P2-WS to use for a pitch candidate, we transform the O-WS and the P2-WSs to a logarithmic scale, and choose the P2-WS closest to the optimal.

Unfortunately, this approach produces discontinuities in the pitch strength (PS) curves, as illustrated in Figure 3-19A. The PS values marked with a plus sign were produced using a WS larger than the WS than the ones marked with a circle. To emphasize the effect, the pitch of the signal (220 Hz) was chosen to match the point at which the change of WS occurs. Since the PS values produced by the larger window in the neighborhood of the pitch are larger than the ones produced by the smaller window, the pitch could be biased toward a lower value.

Although an effort was made to find an appropriate value for the threshold, it was based on an idealized spectrum, which does not have the side lobes found in real spectra. This problem can be alleviated by using a threshold larger than 0.56, determined through trial and error, but we found a better solution: to compute the PS as a linear combination of the PS values produced by the two closest P2-WSs, where the coefficients of the combination are proportional to the log-distance between the P2-WSs and the O-WS.

Concretely, to determine the P2-WSs used to compute the PS of a candidate with frequency f Hz, the O-WS is written as a power of two, $N^* = 2^{L+\lambda}$, where L is an integer and $0 \le \lambda < 1$. Then, the PS values $S_0(f)$ and $S_1(f)$ are computed using windows of size 2^L and 2^{L+1}, respectively. Finally, these PSs are combined into a single one to produce the final PS

$$S(f) = (1-\lambda)\ S_0(f)\ +\ \lambda\ S_1(f)\,. \tag{3-18}$$

Figure 3-19B shows how this combination of PS curves smoothes the discontinuity found in Figure 3-19A.

It would be interesting to know how much is lost in PS by using the formula proposed in Equation 3-18, when the O-WS is not a power of two. This lost can be approximated by finding

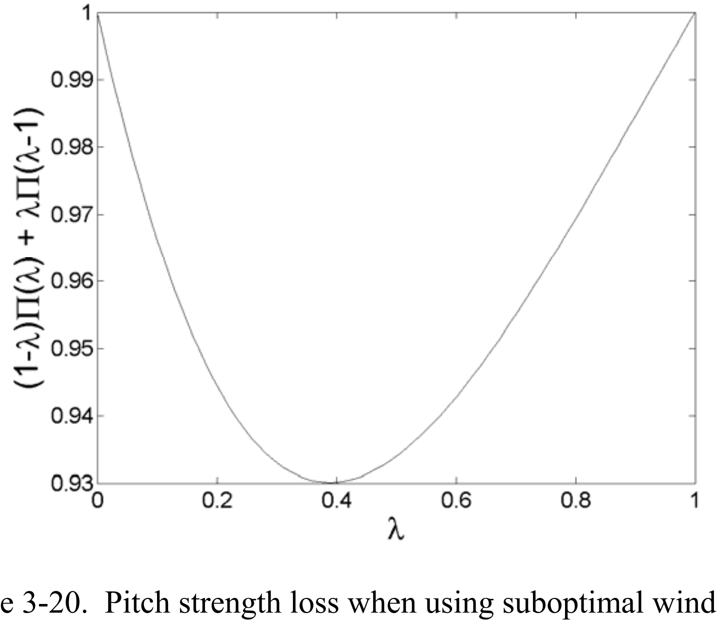

Figure 3-20. Pitch strength loss when using suboptimal window sizes.

the minimum of the linear combination (1-λ) $\Pi(\lambda)$ + λ $\Pi(\lambda$-1) for $0 < \lambda < 1$, which is plotted in Figure 3-20. It can be seen that it has a minimum of 0.93 at around $\lambda = 0.4$. Therefore, the maximum loss when computing PS using the two closest P2-WSs is 7%. Since the minimum PS of a sawtooth waveform when using an O-WS is about 0.92 for SWIPE and 0.83 for SWIPE′ (see Figure 3-15), the minimum pitch strength of a sawtooth waveform when using the two closest P2-WSs is about 0.86 for SWIPE and 0.77 for SWIPE′.

Besides using an optimal window size for the FFT computation, the approximation of O-WSs using P2-WSs has another advantage that is probably more important: the same FFT can be shared by several pitch candidates, more precisely, by all the candidates within an octave of the optimal pitch for that FFT. Going back to the example that started this section, the replacement of the O-WS with the closest P2-WSs reduces the number of FFTs required to estimate the pitch from 480 to just 5: a huge save in computation.

Using this approach, and translating the algorithm to a discrete-time domain (necessary to compute an FFT), we can write the SWIPE′ estimate of the pitch at the discrete-time index τ as

$$p[\tau] = \underset{f}{\arg\max}\ (1-\lambda(f))\ S_{L(f)}(\tau, f)\ +\ \lambda(f)\ S_{L(f)+1}(\tau, f), \tag{3-19}$$

where

$$\lambda(f) = L^*(f) - L(f), \tag{3-20}$$

$$L(f) = \lfloor L^*(f) \rfloor, \tag{3-21}$$

$$L^*(f) = \log_2(4kf_s / f), \tag{3-22}$$

$$S_L(\tau, f) = \frac{\displaystyle\sum_{m=0}^{\left\lfloor \frac{\mathrm{ERBs}(f_{\max})}{\Delta\varepsilon} \right\rfloor} \frac{1}{\eta(m\Delta\varepsilon)^{1/2}} \; K(f, \eta(m\Delta\varepsilon)) \; |\hat{X}_{2^L}[\tau, \eta(m\Delta\varepsilon)]|^{\;1/2}}{\left(\displaystyle\sum_{m=0}^{\left\lfloor \frac{\mathrm{ERBs}(f_{\max})}{\Delta\varepsilon} \right\rfloor} \frac{1}{\eta(m\Delta\varepsilon)} \; K^{+}(f, \eta(m\Delta\varepsilon)) \right)^{1/2} \left(\displaystyle\sum_{m=0}^{\left\lfloor \frac{\mathrm{ERBs}(f_{\max})}{\Delta\varepsilon} \right\rfloor} |\hat{X}_{2^L}[\tau, \eta(m\Delta\varepsilon)]| \right)^{1/2}} \text{,(3-23)}$$

$$\hat{X}_N[\tau, f'] = I \left(\; \{0,\ldots,N-1\}, \;\; X_N[\tau, \{0,\ldots,N-1\}], \;\; f'N / f_s \; \right), \qquad (3\text{-}24)$$

$$X_N[\tau, \varphi] = \sum_{\tau'=-\infty}^{\infty} w_N[\tau' - \tau] \; x[\tau'] \; e^{-j2\pi\varphi\tau'/N} \; , \qquad (3\text{-}25)$$

$\Delta\varepsilon$ is the ERB scale step size (0.1 gives good enough resolution), $I(\Phi,\Xi,\phi)$ is an interpolating function that uses the functional relations $\Xi_k = F(\Phi_k)$ to predict the value of $F(\phi)$, and $X_N[\tau,\varphi]$ ($\varphi = 0, 1,\ldots, N-1$) is the discrete Fourier transform (computed via FFT) of the discrete signal $x[\tau']$, multiplied by the size-N windowing function $w_N[\tau']$, centered at τ. The other variables, constants, and functions are defined as before (see Section 3.8). A Matlab implementation of this algorithm is given in Appendix A.

3.10.2 Reducing the Number of Spectral Integral Transforms

The pitch resolution of SWIPE and SWIPE′ depends on the granularity of the pitch candidates. Therefore, to achieve high pitch resolution, a large number of pitch candidates must be used, and since the pitch strength of each candidate is determined by computing a K^+-NIP between its kernel and the spectrum, the computational cost of the algorithm would increase enormously. To avoid this situation, we propose to compute K^+-NIPs only for certain candidates, and then use interpolation to estimate the pitch strength of the other candidates.

As noted by de Cheveigné (2002), the AC of a signal is the Fourier transform of its power spectrum, and therefore the AC is a sum of cosines that can be approximated around zero by using a Taylor series expansion with even powers. If the signal is periodic, its AC is also periodic, and therefore the shape of the AC around the pitch period is the same as the shape around zero, and therefore it can also be approximated by the same Taylor series, centered at the pitch period. If the width of the spectral lobes is narrow and the energy of the high frequency components is small, the terms of order 4 in the series vanish as the independent variable approaches the pitch period, and therefore the series can be approximated using a parabola.

Since SWIPE perform an inner product between the spectrum and a kernel consisting of cosine lobes, a similar argument can be applied to the pitch strength curves produce by SWIPE. However, the quality of the fit of a parabola is not guaranteed for two reasons: first, the width of the spectral lobes produced by SWIPE are not narrow, in fact, they are as wide as the positive lobes of the cosine; and second, the use of the square-root of the spectrum rather than its energy makes the contribution of the high frequency components large, violating the requirement of low contribution of high frequency components. Nevertheless, parabolic interpolation produces a good fit to the pitch strength curve in the neighborhood of the SWIPE peaks, as we will proceed to show.

Let's derive an approximation to the pitch strength curve $\sigma(t)$ produced by SWIPE for a sawtooth waveform with fundamental frequency $f_0 = 1/T_0$ Hz in the neighborhood of the pitch period T_0. To simplify the equations, let's define the scaling transformations $\omega = 2\pi f$ and $\tau = 2\pi t/T_0$. To make the calculations tractable, let's use idealized spectral lobes (i.e. cosine lobes) and let's ignore the normalization factors and the change of width of the spectral lobe with change of window size caused by a change of pitch candidate. Let's also replace the continuous decaying envelope of the kernel with a decaying step function that gives a weight of $1/\sqrt{k}$ to the k-th harmonic. With all this simplifications, the pitch strength of a candidate with scaled pitch period τ in the neighborhood of 2π (i.e., when the non-scaled pitch period t is in the neighborhood of T_0) can be approximated as

$$\sigma(\tau) = \sum_{k=1}^{n} \sigma_k(\tau)\,, \tag{3-26}$$

where

$$\begin{aligned}
\sigma_k(t) &= \frac{1}{k} \int_{k-1/4}^{k+1/4} \cos(\tau w)\ \cos(2\pi\omega)\ d\omega \\
&= \frac{1}{2k} \int_{k-1/4}^{k-1/4} \left\{ \cos[(t-2\pi)\omega] + \cos[(t+2\pi)\omega] \right\} d\omega \\
&= \frac{1}{2k} \left\{ \frac{\sin[(t-2\pi)\omega]}{t-2\pi} + \frac{\sin[(t+2\pi)\omega]}{t+2\pi} \right\} \Bigg|_{\omega=k-1/4}^{\omega=k+1/4} \\
&= \frac{1}{2k} \left\{ \frac{\sin[(k+1/4)(t-2\pi)] - \sin[(k+1/4)(t-2\pi)]}{t-2\pi} \right. \\
&\qquad \left. + \frac{\sin[(k+1/4)(t+2\pi)] - \sin[(k+1/4)(t+2\pi)]}{t+2\pi} \right\}.
\end{aligned} \tag{3-27}$$

Since we are interested in approximating this function in the neighborhood of 2π, we can equivalently shift the function 2π units to the left by defining $\sigma'_k(\tau) = \sigma_k(\tau + 2\pi)$, and then approximate $\sigma'_k(\tau)$ in the neighborhood of zero. Since $\sin(x)/x = 1 - x^2/3! + x^4/5! - O(x^6)$ in the neighborhood of zero, it is useful to express $\sigma'_k(\tau)$ as

$$\begin{aligned}
\sigma'_k(\tau) &= \frac{k+1/4}{2k}\ \frac{\sin[(k+1/4)\tau]}{(k+1/4)\tau} + \frac{k-1/4}{2k}\ \frac{\sin[(k-1/4)\tau]}{(k-1/4)\tau} \\
&\qquad + \frac{1}{2k}\ \frac{\sin[(k-1/4)\tau] - \sin[(k+1/4)\tau]}{\tau + 4\pi},
\end{aligned} \tag{3-28}$$

which has the Taylor series expansion

$$
\begin{aligned}
\sigma'_k(\tau) = {} & \frac{k+1/4}{2k}\left[1-\frac{(k+1/4)^2}{3!}\tau^2+\frac{(k+1/4)^4}{5!}\tau^4-O\left(\tau^6\right)\right] \\
& - \frac{k-1/4}{2k}\left[1-\frac{(k-1/4)^2}{3!}\tau^2+\frac{(k-1/4)^4}{5!}\tau^4-O\left(\tau^6\right)\right] \\
& + \frac{1}{2k(\tau+4\pi)}\left\{\left[(k-1/4)\tau-\frac{(k-1/4)^3}{3!}\tau^3+O\left(\tau^5\right)\right]\right. \\
& \qquad \left. -\left[(k+1/4)\tau-\frac{(k+1/4)^3}{3!}\tau^3+O\left(\tau^5\right)\right]\right\}
\end{aligned}
\qquad (3\text{-}29)
$$

in the neighborhood of zero. Finally, the approximation of the pitch strength curve in the shifted-time domain is

$$\sigma'(\tau)=a_0+a_1\tau+a_2\tau^2+a_3\tau^3+a_4\tau^4+O\left(\tau^5\right)=\sum_{k=1}^{n}\sigma'_k(\tau)\,. \qquad (3\text{-}30)$$

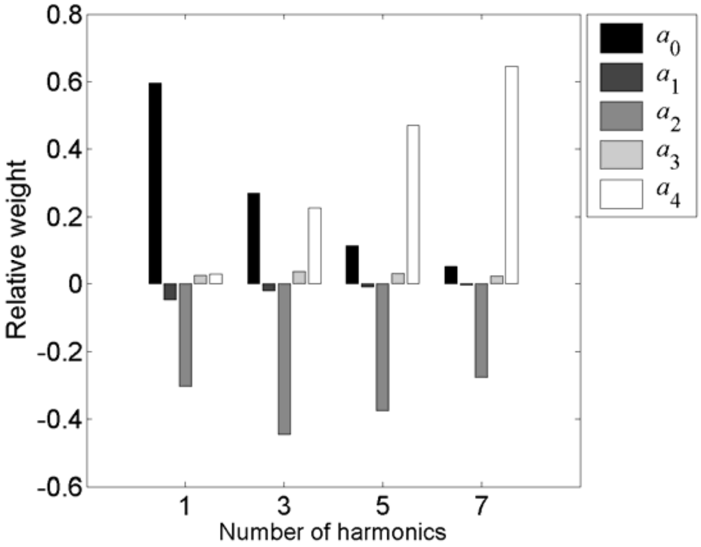

Figure 3-21. Coefficients of the pitch strength interpolation polynomial.

Figure 3-21 shows the relative value of the coefficients of the expansion as a function of the number of harmonics in the signal. As the number of harmonics increases, the relative weight of the order-4 coefficient increases. However, as τ approaches zero, its fourth power becomes so small that its overall contribution to the sum is small compared to the contribution of the order-2 term.

This effect is clear in Figure 3-22, which shows $\sigma'(\tau)$ for a sawtooth waveform with 15 harmonics using polynomials of order 2 and order 4 in the range $+/- 0.045$, which corresponds to $+/- 1/8$ semitones. The curve has been scaled to have a maximum of 1. The large circles correspond to candidates separated by 1/8 semitones, which is the interval used in our implementation of SWIPE and SWIPE′ for the distance between pitch candidates for which the pitch strength is computed directly. The other markers correspond to candidates separated by 1/64 semitones, which is the resolution used to fine tune the pitch strength curve based on the pitch strength of the candidates for which the pitch strength is computed directly. As observed in

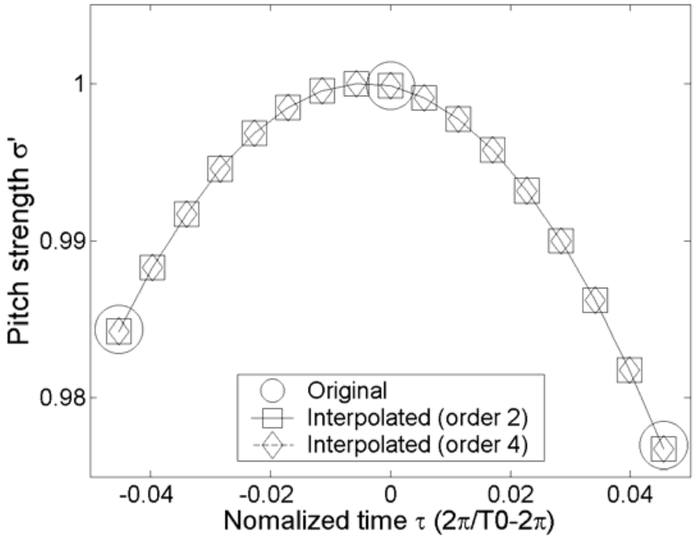

Figure 3-22. Interpolated pitch strength.

the figure, for such small values of τ, the pitch strength values obtained with an order 2 polynomial (squares) are indistinguishable from the ones obtained with an order 4 polynomial (diamonds). Hence, a parabola is good enough to estimate the pitch strength between candidates separated at distances as small as 1/8 semitones.

3.11 Summary

This chapter described the SWIPE algorithm and its variation SWIPE′. The initial approach of the algorithm was the search for the frequency that maximizes the average peak-to-valley distance at harmonic locations. Several modifications to this idea were applied to improve its performance: the locations of the harmonics were blurred, the spectral amplitude and the frequency scale were warped, an appropriate window type and size were chosen, and simplifications to reduce computational cost were introduced. After these modifications, SWIPE estimates the pitch as the fundamental frequency of the sawtooth waveform whose spectrum best matches the spectrum of the input signal. Its variation, SWIPE′, uses only the first and prime harmonics of the signal.

CHAPTER 4
EVALUATION

To asses the relevance of SWIPE and SWIPE′, they were compared against other algorithms using three speech databases and a musical instruments database. This chapter presents a brief description of these algorithms, databases, and the evaluation process. A more detailed description is given in Appendix B.

4.1 Algorithms

The algorithms with which SWIPE and SWIPE′ were compared were the following:

- **AC-P**: This algorithm (Boersma, 1993) computes the autocorrelation of the signal and divides it by the autocorrelation of the window used to analyze the signal. It uses post-processing to reduce discontinuities in the pitch trace. It is available with the Praat System at ⟨http://www.fon.hum.uva.nl/praat⟩. The name of the function is *ac*.

- **AC-S**: This algorithm uses the autocorrelation of the cubed signal. It is available with the Speech Filing System at ⟨http://www.phon.ucl.ac.uk/resource/sfs⟩. The name of the function is *fxac*.

- **ANAL**: This algorithm (Secrest and Doddington, 1983) uses autocorrelation to estimate the pitch, and dynamic programming to remove discontinuities in the pitch trace. It is available with the Speech Filing System at ⟨http://www.phon.ucl.ac.uk/resource/sfs⟩. The name of the function is *fxanal*.

- **CATE**: This algorithm uses a quasi autocorrelation function of the speech excitation signal to estimate the pitch. We implemented it based on its original description (Di Martino, 1999). The dynamic programming component used to remove discontinuities in the pitch trace was not implemented.

- **CC**: This algorithm uses cross-correlation to estimate the pitch and post-processing to remove discontinuities in the pitch trace. It is available with the Praat System at ⟨http://www.fon.hum.uva.nl/praat⟩. The name of the function is *cc*.

- **CEP**: This algorithm (Noll, 1967) uses the cepstrum of the signal and is available with the Speech Filing System at ⟨http://www.phon.ucl.ac.uk/resource/sfs⟩. The name of the function is *fxcep*.

- **ESRPD**: This algorithm (Bagshaw, 1993; Medan, 1991) uses a normalized cross-correlation to estimate the pitch, and post-processing to remove discontinuities in the pitch trace. It is available with the Festival Speech Filing System at ⟨http://www.cstr.ed.ac.uk/projects/festival⟩. The name of the function is *pda*.

- **RAPT**: This algorithm (Secrest and Doddington, 1983) uses a normalized cross-correlation to estimate the pitch, and dynamic programming to remove discontinuities in the pitch trace. It is available with the Speech Filing System at ⟨http://www.phon.ucl.ac.uk/resource/sfs⟩. The name of the function is *fxrapt*.

- **SHS**: This algorithm (Hermes, 1988) uses subharmonic summation. It is available with the Praat System at ⟨http://www.fon.hum.uva.nl/praat⟩. The name of the function is *shs*.

- **SHR**: This algorithm (Sun, 2000) uses the subharmonic-to-harmonic ratio. It is available at Matlab Central ⟨http://www.mathworks.com/matlabcentral⟩ under the title "Pitch Determination Algorithm". The name of the function is *shrp*.

- **TEMPO**: This algorithm (Kawahara et al., 1999) uses the instantaneous frequency of the outputs of a filterbank. It is available with the STRAIGHT System at its author web page ⟨http://www.wakayama-u.ac.jp/~kawahara⟩. The name of the function is *exstraightsource*.

- **YIN**: This algorithm (de Cheveigné and Kawahara, 2002) uses a modified version of the average squared difference function. It is available from its author web page at ⟨http://www.ircam.fr/pcm/cheveign/sw/yin.zip⟩. The name of the function is *yin*.

4.2 Databases

The databases used to test the algorithms were the following:

- **DVD**: *Disordered Voice Database*. This database contains 657 samples of sustained vowels produced by persons with disordered voice. It can be bought from Kay Pentax ⟨http://www.kayelemetrics.com⟩.

- **KPD**: *Keele Pitch Database*. This speech database was collected by Plante et. al (1995) at Keele University with the purpose of evaluating pitch estimation algorithms. It contains about 8 minutes of speech spoken by five males and five females. Laryngograph data was recorded simultaneously with speech, and was used to produce estimates of the fundamental frequency. It is publicly available at ⟨ftp://ftp.cs.keele.ac.uk/pub/pitch⟩.

- **MIS**: *Musical Instruments Samples*. This database contains more than 150 minutes of sound produced by 20 different musical instruments. It was collected at the University of Iowa Electronic Music Studios, directed by Lawrence Fritts, and is publicly available at ⟨http://theremin.music.uiowa.edu⟩.

- **PBD**: *Paul Bagshaw's Database for evaluating pitch determination algorithms*. This database contains about 8 minutes of speech spoken by one male and one female. Laryngograph data was recorded simultaneously with speech, and was used to produce estimates of the fundamental frequency. It was collected by Paul Bagshaw at the University of Edinburg (Bagshaw *et. al* 1993; Bagshaw 1994), and is publicly available at ⟨http://www.cstr.ed.ac.uk/research/projects/fda⟩.

4.3 Methodology

The algorithms were asked to produce a pitch estimate every millisecond. The search range was set to 40-800 Hz for speech and 30-1666 Hz for musical instruments. The algorithms were given the freedom to decide if the sound was pitched or not. However, to compute our statistics, we considered only the time instants at which *all* the algorithms agreed that the sound was pitched.

Special care was taken to account for time misalignments. Specifically, the pitch estimates were associated to the time corresponding to the center of their respective analysis windows, and when the ground truth pitch varied over time (i.e., for PBD and KPD), the estimated pitch time series were shifted within a range of +/-100 ms to find the best alignment with the ground truth.

The performance measure used to compare the algorithms was the gross error rate (GER). A gross error occurs when the estimated pitch is off from the reference pitch by more than 20%. At first glance this margin of error seems too large, but considering that most of the errors pitch estimation algorithms produce are octave errors (i.e., halving or doubling the pitch), this is a reasonable metric. On the other hand, this tolerance gives room for dealing with misalignments. The GER measure has been used previously to test PEAs by other researchers (Bagshaw, 1993; Di Martino, 1999; de Cheveigne and Kawahara, 2002).

4.4 Results

Table 4-1 shows the GERs for each of the algorithms over each of the speech databases. Both the rows and the columns are sorted by average GER: the best algorithms are at the top, and the more difficult databases are at the right. The best algorithm overall is SWIPE′, followed by SHS and SWIPE. Although on average SHS performs better than SWIPE, the only database in which SHS beats SWIPE is in the disordered voice database, which indicates that SWIPE performs better than SHS on normal speech.

Table 4-1. Gross error rates for speech*

	Gross error (%)			
Algorithm	PBD	KPD	DVD	Average
SWIPE′	0.13	0.83	0.63	0.53
SHS	0.15	1.00	1.10	0.75
SWIPE	0.15	0.87	1.70	0.91
RAPT	0.75	1.00	2.40	1.40
TEMPO	0.32	1.90	2.00	1.40
YIN	0.33	1.40	4.50	2.10
SHR	0.69	1.50	5.10	3.50
ESRPD	1.40	3.90	4.60	5.00
CEP	6.10	4.20	14.00	5.90
AC-P	0.73	2.90	16.00	6.70
CATE	2.60	10.00	7.20	6.60
CC	0.48	3.60	5.00	2.40
ANAL	0.83	2.00	35.00	13.00
AC-S	8.80	7.00	40.00	19.00
Average	1.70	3.00	9.90	4.90

* Values computed using two significant digits.

Table 4-2. Proportion of overestimation errors relative to total gross errors*

	Proportion of overestimations			
Algorithm	DVD	PBD	KPD	Average
CC	0.0	0.0	0.1	0.0
SHS	0.0	0.0	0.3	0.1
RAPT	0.0	0.1	0.5	0.2
SHR	0.0	0.4	0.3	0.2
AC	0.0	0.4	0.2	0.2
AC	0.0	0.2	0.3	0.2
ANAL	0.0	0.5	0.4	0.3
CEP	0.4	0.5	0.4	0.4
SWIPE′	0.0	0.6	0.7	0.4
SWIPE	0.1	0.6	0.7	0.4
YIN	0.1	0.9	0.5	0.5
TEMPO	0.1	0.8	0.9	0.6
CATE	0.5	0.5	0.8	0.6
ESRPD	0.5	0.7	0.9	0.7
Average	0.1	0.4	0.5	0.3

* Values computed using one significant digit.

Table 4-2 shows the proportion of GEs caused by overestimations of the pitch with respect to the total number of GEs. The proportion of GEs caused by underestimation of the pitch is just

Table 4-3. Gross error rates by gender*

	Gross error (%)		
Algorithm	Male	Female	Average
SWIPE′	0.36	2.40	1.4
SHS	0.55	2.50	1.5
SWIPE	0.49	2.70	1.6
RAPT	0.42	2.90	1.7
TEMPO	0.67	3.10	1.9
SHR	0.61	3.60	2.1
YIN	1.10	3.20	2.2
AC-P	2.10	3.60	2.9
CEP	1.80	4.20	3.0
CC	2.40	4.50	3.5
ESRPD	3.10	3.90	3.5
ANAL	1.30	5.90	3.6
AC-S	3.20	10.00	6.6
CATE	11.00	4.20	7.6
Average	2.10	4.00	3.1

* Values computed using two significant digits.

one minus the values shown in the table. Algorithms at the top have a tendency to underestimate the pitch while algorithms at the bottom have a tendency to overestimate it. Most algorithms tend to underestimate the pitch in the disordered voice database while the errors are more balanced in the normal speech databases.

Table 4-3 shows the pitch estimation performance as a function of gender for the two databases for which we had access to this information: PVD and KPD. The error rates are on average larger for female speech than for male speech.

Table 4-4 shows the GERs for the musical instruments database. Some of the algorithms were not evaluated on this database because they did not provide a mechanism to set the search range, and the range they covered was smaller that the pitch range spanned by the database. The two algorithms that performed the best were SWIPE′ and SWIPE.

Table 4-4. Gross error rates for musical instruments*

	Gross error (%)		
Algorithm	Underestimates	Overestimates	Total
SWIPE′	1.00	0.10	1.10
SWIPE	1.30	0.02	1.30
SHS	0.88	1.00	1.90
TEMPO	0.29	1.70	2.00
YIN	1.60	0.83	2.40
AC-P	3.20	0.00	3.20
CC	3.60	0.00	3.60
ESRPD	5.30	1.50	6.80
SHR	15.00	5.30	20.00
Average	3.60	1.20	4.70

* Values computed using two significant digits.

Table 4-5. Gross error rates by instrument family*

	Gross error (%)					
		Bowed			Plucked	
Algorithm	Brass	Strings	Woodwinds	Piano	Strings	Average
SWIPE'	0.01	0.19	0.14	2.20	8.80	2.30
SWIPE	0.00	0.22	0.23	0.02	11.00	2.30
TEMPO	0.00	2.60	1.40	7.30	4.00	3.10
YIN	0.03	1.10	1.50	0.36	14.00	3.40
SHS	0.02	1.50	0.72	12.00	8.10	4.50
AC-P	0.03	0.56	0.80	0.36	26.00	5.60
CC	0.07	0.83	1.00	0.36	28.00	6.00
ESRPD	4.00	6.90	7.10	6.00	11.00	7.00
SHR	22.00	25.00	38.00	26.00	15.00	25.00
Average	2.90	4.30	5.60	6.10	14.00	6.60

* Values computed using two significant digits. Brass: French horn, bass/tenor trombones, trumpet, and tuba. Bowed strings: double bass, cello, viola, and violin. Woodwinds: flute, bass/alto flutes, bass/Bb/Eb clarinets, alto/soprano saxophones. Plucked strings: double bass and violin.

Table 4-5 shows the GERs by instrument family. The two best algorithms are SWIPE′ and SWIPE. SWIPE′ tends to perform better than SWIPE except for the piano, for which SWIPE produces almost no error. On the other hand, SWIPE′ performance on piano is relatively bad compared to correlation based algorithms. The family for which fewer errors were obtained was the brass family; many algorithms achieved almost perfect performance for this family. The

Table 4-6. Gross error rates for musical instruments by octave*

	Gross error (%)						
Algorithm	46.2 Hz +/- 1/2 oct.	92.5 Hz +/- 1/2 oct.	185 Hz +/- 1/2 oct.	370 Hz +/- 1/2 oct.	740 Hz +/- 1/2 oct.	1480 Hz +/- 1/2 oct.	Average
SWIPE'	1.20	1.00	2.30	0.89	0.13	0.29	0.97
SWIPE	0.08	1.20	3.00	1.00	0.25	0.38	0.99
YIN	3.20	0.95	5.30	1.80	0.69	0.96	2.20
AC-P	0.24	2.00	7.80	2.50	0.71	0.30	2.30
SHS	7.80	2.60	3.20	1.20	0.23	0.14	2.50
CC	0.26	2.60	8.20	2.70	0.93	0.40	2.50
TEMPO	15.00	2.80	2.00	1.10	0.52	0.31	3.60
ESRPD	7.90	2.60	4.80	4.20	12.00	32.00	11.00
SHR	37.00	0.60	1.80	27.00	70.00	81.00	36.00
Average	8.10	1.80	4.30	4.70	9.50	13.00	6.90

* Values computed using two significant digits.

family for which more errors were produced was the strings family playing *pizzicato*, i.e., by plucking the strings. Indeed, pizzicato sounds were the ones for which the performers produced more errors and the ones that were hardest for us to label (see Appendix B).

Table 4-6 shows the GERs as a function of octave. The best performance on average was achieved by SWIPE′ and SWIPE. The results of the algorithms with an average GER less than

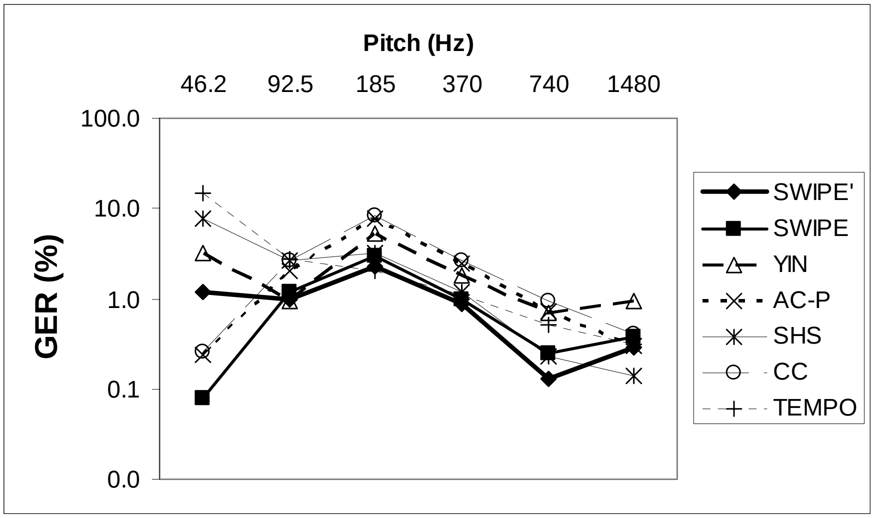

Figure 4-1. Gross error rates for musical instruments as a function of pitch.

Table 4-7. Gross error rates for musical instruments by dynamic*

	Gross error (%)			
Algorithm	*pp*	*mf*	*ff*	Average
SWIPE'	1.30	1.20	0.92	1.10
SWIPE	1.40	1.40	1.20	1.30
SHS	1.50	2.30	2.00	1.90
TEMPO	2.00	1.90	2.00	2.00
YIN	2.20	2.50	2.40	2.40
AC-P	3.30	3.20	3.30	3.30
CC	3.60	3.30	3.80	3.60
ESRPD	5.70	7.10	7.60	6.80
SHR	27.00	29.00	29.00	28.00
Average	5.30	5.80	5.80	5.60

* Values computed using two significant digits.

10% is reproduced in Figure 4-1. All algorithms have approximately the same tendency, except at the lowest octave, where a larger variance in the GERs can be observed.

Table 4-7 shows the GERs as a function of dynamic (i.e., loudness). In general, there is no significant variation of GERs with changes in loudness, although SWIPE′ has a tendency to reduce the GER as loudness increases [i.e., as the dynamic moves from pianissimo (*pp*) to fortissimo (*ff*)].

As a final test, we wanted to validate the choices we made in Chapter 3, i.e., shape of the kernel, warping of the spectrum, weighting of the harmonics, warping of the frequency scale, and selection of window type and size. For this purpose, we evaluated SWIPE′ replacing every time one of its features with a more standard feature, i.e., smooth vs. pulsed kernels, square-root vs. raw spectrum, decaying vs. flat kernel envelope, ERB vs. Hertz frequency scale, and pitch-optimized vs. fixed window size. We varied each of these variables independently and obtained the results shown in Table 4-8. Although some of the variations made SWIPE′ improve in some of the databases, overall SWIPE′ worked better with the features we proposed in Chapter 3.

Table 4-8. Gross error rates for variations of SWIPE′*

	Gross error (%)				
Variation	PBD	KPD	DVD	MIS	Average
Original	0.13	0.83	0.63	1.10	0.67
Flat envelope	0.16	1.00	1.40	0.60	0.79
Hertz scale[1]	0.23	1.70	1.40	0.37	0.93
Pulsed kernel	0.21	0.84	3.00	2.60	1.70
Raw spectrum[2]	0.25	2.10	1.60	4.90	2.20
Fixed WS[3]	0.15	0.77	1.70	9.10	2.90

* Values computed using two significant digits. [1] FFTs were computed using optimal window sizes and the spectrum was inter/extrapolated to frequency bins separated at 5 Hz.[2] The use of the raw spectrum rather than the square root of the spectrum implies the use of a kernel whose envelope decays as $1/f$ rather than $1/\sqrt{f}$, to match the spectral envelope of a sawtooth waveform.[3] The power-of-two window size whose optimal pitch was closest to the geometric mean pitch of the database was used in each case. A window of size 1024 samples was used for the speech databases and a window of size of 256 samples was used for the musical instruments database.

4.5 Discussion

SWIPE′ showed the best performance in all categories. SWIPE was the second best ranked for musical instruments and normal speech but not for disordered speech, for which SHS performed better (see Table 4-1). One possible reason is that it is common for disordered voices to have energy at multiples of subharmonics of the pitch, and therefore algorithms that apply negative weights to the spectral regions between harmonics (e.g., SWIPE, SWIPE′, and all autocorrelation based algorithms) are prone to produce low scores for the pitch. Although SWIPE′ is among this group, its use of only the first and prime harmonics, reduces substantially the score subharmonics of the pitch, producing most of the time a larger score for the pitch than for its subharmonics.

The rankings of the algorithms are relatively stable in all the tables except for SHR, which showed a good performance for speech but not for musical instruments. We believe this is caused by the wide pitch range spanned by the musical instruments. This is suggested by the results in Table 4-6, which show that SHR performs well in the octaves around 92.5 Hz and 185

Hz, which corresponds to the pitch region of speech, but performs very bad as the pitch moves from this region.

Figure 4-1 shows that the relative trend on performance with pitch for musical instruments is about the same for all the algorithms except in the lowest region, where a large variance in performance was observed. However, this variance may be caused by a significant reduction in the numbers of samples in this region (about 4% of the data). The figure also shows an overall increase in GER in the octave around 185 Hz. We believe this is caused by the presence of a set of difficult sounds in the database with pitches in that region, since it is hard to believe that there is an inherent difficulty of the algorithms to recognize pitch in that region.

CHAPTER 5
CONCLUSION

The SWIPE pitch estimator has been developed. SWIPE estimates the pitch as the fundamental frequency of the sawtooth waveform whose spectrum best matches the spectrum of the input signal. The schematic description of the algorithm is the following:

1. For each pitch candidate f within a pitch range $f_{\min}$-$f_{\max}$, compute its pitch strength as follows:
    a. Compute the square-root of the spectrum of the signal.
    b. Normalize the square-root of the spectrum and apply an integral transform using a normalized cosine kernel whose envelope decays as $1/\sqrt{f}$.
2. Estimate the pitch as the candidate with highest strength.

An implicit objective of the algorithm was to find the frequency for which the average peak-to-valley distance at its harmonics is maximized. To achieve this, the kernel was set to zero below the first negative lobe and above the last negative lobe, and to avoid bias, the magnitude of these two lobes was halved.

To make the contribution of each harmonic of the sawtooth waveform proportional to its amplitude and not to the square of its amplitude, the square-root of the spectrum was taken before applying the integral transform.

To make the kernel match the normalized square-root spectrum of the sawtooth waveform, a $1/\sqrt{f}$ envelope was applied to the kernel. The kernel was normalized using only its positive part.

To maximize the similarity between the kernel and the square-root of the input spectrum, each pitch candidate required its own window size, which in general is not a power of two, and therefore not ideal to compute an FFT. To reduce computational cost, the two closest power-of-two window sizes were used, and their results are combined to produce a single pitch strength value. This had the extra advantage of allowing an FFT to be shared by many pitch candidates.

Another technique used to reduce computational cost was to compute a coarse pitch strength curve and then fine tune it by using parabolic interpolation. A last technique used to reduce computational cost was to reduce the amount of window overlap while allowing the pitch of a signal as short as four cycles to be recognized.

The ERB frequency scale was used to compute the spectral integral transform since the density of this scale decreases almost proportionally to frequency, which avoids wasting computation in regions where there little spectral energy is expected.

SWIPE′, a variation of SWIPE, uses only the first and prime harmonics of the signal, producing a large reduction in subharmonic errors by reducing significantly the scores of subharmonics of the pitch.

Except for the obvious architectural decisions that must be taken when creating an algorithm (e.g., selection of the kernel), there are no free parameters in SWIPE and SWIPE′, at least in terms of "magic numbers".

SWIPE and SWIPE′ were tested using speech and musical instruments databases and their performance was compared against twelve other algorithms which have been cited in the literature and for which free implementations exist. SWIPE′ was shown to outperform all the algorithms on all the databases. SWIPE was ranked second in the normal speech and musical instruments databases, and was ranked third in the disordered speech database.

APPENDIX A
MATLAB IMPLEMENTATION OF SWIPE′

This is a Matlab implementation of SWIPE′. To convert it into SWIPE just replace [1 primes(n)] in the *for* loop of the function *pitchStrengthOneCandidate* with [1:n].

```
function [p,t,s] = swipep(x,fs,plim,dt,dlog2p,dERBs,sTHR)
% SWIPEP Pitch estimation using SWIPE'.
% P = SWIPEP(X,Fs,[PMIN PMAX],DT,DLOG2P,DERBS,STHR) estimates the pitch of
% the vector signal X with sampling frequency Fs (in Hertz) every DT
% seconds. The pitch is estimated by sampling the spectrum in the ERB scale
% using a step of size DERBS ERBs. The pitch is searched within the range
% [PMIN PMAX] (in Hertz) sampled every DLOG2P units in a base-2 logarithmic
% scale of Hertz. The pitch is fine tuned by using parabolic interpolation
% with a resolution of 1/64 of semitone (approx. 1.6 cents). Pitches with a
% strength lower than STHR are treated as undefined.
%
% [P,T,S] = SWIPEP(X,Fs,[PMIN PMAX],DT,DLOG2P,DERBS,STHR) returns the times
% T at which the pitch was estimated and their corresponding pitch strength.
%
% P = SWIPEP(X,Fs) estimates the pitch using the default settings PMIN =
% 30 Hz, PMAX = 5000 Hz, DT = 0.01 s, DLOG2P = 1/96 (96 steps per octave),
% DERBS = 0.1 ERBs, and STHR = -Inf.
%
% P = SWIPEP(X,Fs,...[],...) uses the default setting for the parameter
% replaced with the placeholder [].
%
%    EXAMPLE: Estimate the pitch of the signal X every 10 ms within the
%    range 75-500 Hz using the default resolution (i.e., 96 steps per
%    octave), sampling the spectrum every 1/20th of ERB, and discarding
%    samples with pitch strength lower than 0.4. Plot the pitch trace.
%     [x,Fs] = wavread(filename);
%     [p,t,s] = swipep(x,Fs,[75 500],0.01,[],1/20,0.4);
%     plot(1000*t,p)
%     xlabel('Time (ms)')
%     ylabel('Pitch (Hz)')
if ~ exist( 'plim', 'var' ) || isempty(plim), plim = [30 5000]; end
if ~ exist( 'dt', 'var' ) || isempty(dt), dt = 0.01; end
if ~ exist( 'dlog2p', 'var' ) || isempty(dlog2p), dlog2p = 1/96; end
if ~ exist( 'dERBs', 'var' ) || isempty(dERBs), dERBs = 0.1; end
if ~ exist( 'sTHR', 'var' ) || isempty(sTHR), sTHR = -Inf; end
t = [ 0: dt: length(x)/fs ]'; % Times
dc = 4; % Hop size (in cycles)
K = 2; % Parameter k for Hann window
% Define pitch candidates
log2pc = [ log2(plim(1)): dlog2p: log2(plim(end)) ]';
pc = 2 .^ log2pc;
S = zeros( length(pc), length(t) ); % Pitch strength matrix
% Determine P2-WSs
logWs = round( log2( 4*K * fs ./ plim ) );
ws = 2.^[ logWs(1): -1: logWs(2) ]; % P2-WSs
pO = 4*K * fs ./ ws; % Optimal pitches for P2-WSs
% Determine window sizes used by each pitch candidate
d = 1 + log2pc - log2( 4*K*fs./ws(1) );
```

```
% Create ERBs spaced frequencies (in Hertz)
fERBs = erbs2hz([ hz2erbs(pc(1)/4): dERBs: hz2erbs(fs/2) ]');
for i = 1 : length(ws)
    dn = round( dc * fs / pO(i) ); % Hop size (in samples)
    % Zero pad signal
    xzp = [ zeros( ws(i)/2, 1 ); x(:); zeros( dn + ws(i)/2, 1 ) ];
    % Compute spectrum
    w = hanning( ws(i) ); % Hann window
    o = max( 0, round( ws(i) - dn ) ); % Window overlap
    [ X, f, ti ] = specgram( xzp, ws(i), fs, w, o );
    % Interpolate at equidistant ERBs steps
    M = max( 0, interp1( f, abs(X), fERBs, 'spline', 0) ); % Magnitude
    L = sqrt( M ); % Loudness
    % Select candidates that use this window size
    if i==length(ws); j=find(d-i>-1); k=find(d(j)-i<0);
    elseif i==1; j=find(d-i<1); k=find(d(j)-i>0);
    else j=find(abs(d-i)<1); k=1:length(j);
    end
      Si = pitchStrengthAllCandidates( fERBs, L, pc(j) );
    % Interpolate at desired times
    if size(Si,2) > 1
        Si = interp1( ti, Si', t, 'linear', NaN )';
    else
        Si = repmat( NaN, length(Si), length(t) );
    end
    lambda = d( j(k) ) - i;
    mu = ones( size(j) );
    mu(k) = 1 - abs( lambda );
    S(j,:) = S(j,:) + repmat(mu,1,size(Si,2)) .* Si;
end
% Fine-tune the pitch using parabolic interpolation
p = repmat( NaN, size(S,2), 1 );
s = repmat( NaN, size(S,2), 1 );
for j = 1 : size(S,2)
      [ s(j), i ] = max( S(:,j) );
    if s(j) < sTHR, continue, end
    if i==1, p(j)=pc(1); elseif i==length(pc), p(j)=pc(1); else
        I = i-1 : i+1;
        tc = 1 ./ pc(I);
        ntc = ( tc/tc(2) - 1 ) * 2*pi;
            c = polyfit( ntc, S(I,j), 2 );
        ftc = 1 ./ 2.^[ log2(pc(I(1))): 1/12/64: log2(pc(I(3))) ];
        nftc = ( ftc/tc(2) - 1 ) * 2*pi;
            [s(j) k] = max( polyval( c, nftc ) );
        p(j) = 2 ^ ( log2(pc(I(1))) + (k-1)/12/64 );
    end
end

function S = pitchStrengthAllCandidates( f, L, pc )
% Normalize loudness
warning off MATLAB:divideByZero
L = L ./ repmat( sqrt( sum(L.*L) ), size(L,1), 1 );
warning on MATLAB:divideByZero
% Create pitch salience matrix
S = zeros( length(pc), size(L,2) );
for j = 1 : length(pc)
    S(j,:) = pitchStrengthOneCandidate( f, L, pc(j) );
```

```
end

function S = pitchStrengthOneCandidate( f, L, pc )
n = fix( f(end)/pc - 0.75 ); % Number of harmonics
k = zeros( size(f) ); % Kernel
q = f / pc; % Normalize frequency w.r.t. candidate
for i = [ 1 primes(n) ]
    a = abs( q - i );
    % Peak's weigth
    p = a < .25;
    k(p) = cos( 2*pi * q(p) );
    % Valleys' weights
    v = .25 < a & a < .75;
    k(v) = k(v) + cos( 2*pi * q(v) ) / 2;
end
% Apply envelope
k = k .* sqrt( 1./f  );
% K+-normalize kernel
k = k / norm( k(k>0) );
% Compute pitch strength
S = k' * L;

function erbs = hz2erbs(hz)
erbs = 21.4 * log10( 1 + hz/229 );

function hz = erbs2hz(erbs)
hz = ( 10 .^ (erbs./21.4) - 1 ) * 229;
```

APPENDIX B
DETAILS OF THE EVALUATION

B.1 Databases

All the databases used in this work are free and publicly available on the Internet, except the disordered voice database. Besides speech recordings, the speech databases contain simultaneous recordings of laryngograph data, which facilitates the computation of the fundamental frequency. The authors of these databases used them to produce ground truth pitch values, which are also included in the databases. The disordered voice database includes fundamental frequency estimates, but as it will be explained later, a different ground truth data set was used. The musical instruments database contains the names of the notes in the names of the files.

B.1.1 Paul Bagshaw's Database

Paul Bagshaw's database (PBD) for evaluation of pitch determination algorithms (Bagshaw *et. al* 1993; Bagshaw 1994) was collected at the University of Edinburgh, and is available at ⟨http://www.cstr.ed.ac.uk/research/projects/fda⟩. The speech and laryngograph signals of this database were sampled at 20 kHz. The ground truth fundamental frequency was computed by estimating the location of the glottal pulses in the laryngograph data and taking the inverse of the distance between each pair of consecutive pulses. Each fundamental frequency estimate is associated to the time instant in the middle between the pair of pulses used to derive the estimate.

B.1.2 Keele Pitch Database

The Keele Pitch Database (KPD) (Plante *et. al*, 1995) was created at Keele University and is available at ⟨ftp://ftp.cs.keele.ac.uk/pub/pitch⟩. The speech and laryngograph signals were sampled at 20 kHz. The fundamental frequency was estimated by using autocorrelation over a

26.5 ms window shifted at intervals of 10 ms. Windows where the pitch is unclear are marked with special codes.

Both of these speech databases PBD and KPD have been reported to contain errors (de Cheveigne, 2002), especially at the end of sentences, where the energy of speech decays and malformed pulses may occur. We will explain later how we deal with this problem.

B.1.3 Disordered Voice Database

The disordered voice database (DVD) was collected by Kay Pentax ⟨http://www.kayelemetrics.com⟩. It includes 657 disordered voice samples of the sustained vowel "ah" sampled at 25 kH, and some few at 50 kHz. The database includes samples from patients with a wide variety of organic, neurological, traumatic, psychogenic, and other voice disorders.

The database includes fundamental frequency estimates, but by definition, they do not necessarily match their pitch. Therefore we estimated the pitch by ourselves by listening to the samples through earphones, and matching the pitch to the closest note, using as reference a synthesizer playing sawtooth waveforms. Assuming that we chose one of the two closest notes every time, this procedure should introduce an error no larger than 6%, which is smaller than the 20% necessary to produce a GE (see Chapter 4).

There were some samples for which the pitch ranged over a perfect fourth or more (i.e., the higher pitch was more than 33% higher than the lower pitch). Since this range is large compared to the permissible 20%, these samples were excluded. Samples for which the range did not span more than a major third (i.e., the higher pitch was no more than 26% higher than the lower pitch) were preserved, and they were assigned the note corresponding to the median of the range. If the median was between two notes, it was assigned to any of them. This should introduce an error no larger than two semitones (12%), which is about half the maximum permissible error of 20%.

There were 30 samples for which we could not perceive with confidence a pitch, so they were excluded as well.

Since the ground truth data was based on the perception of only one listener (the author), it could be argued that this data has low validity. To alleviate this, we excluded the samples for which the minimum error produced by any algorithm was larger than 50%.

After excluding the non-pitch, variable pitch, and samples at which the algorithms disagreed with the ground truth, we ended up with 612 samples out of the original 657. Appendix C shows the ground truth used for each of these 612 samples.

B.1.4 Musical Instruments Database

The musical instruments samples database was collected at the University of Iowa, and is available at ⟨http://theremin.music.uiowa.edu⟩. The recordings were made using CD quality sampling at a rate of 44,100 kHz, but we downsampled them to 10 kHz in order to reduce computational cost. No noticeable change of perceptual pitch was perceived by doing this, even for the highest pitch sounds. This database contains recordings of 20 instruments, for a total of more than 150 minutes and 4,000 notes. The notes are played in sequence using a chromatic scale with silences in between. Each file usually spans one octave and is labeled with the name of the initial and final notes, plus the name of the instrument, and other details (e.g., Violin.pizz.mf.sulG.C4B4.aiff).

In order to test the algorithms, the files were split into separate files containing each of them a single note with no leading or trailing silence. This process was done in a semi-automatic way by using a power-based segmentation method, and then checking visually and auditively the quality of the segmentation.

While doing this task it was discovered that some of the note labels were wrong. The intervals produced by the performers were sometimes larger than a semitone, and therefore the names of the files did not correspond to the notes that were in fact played. This situation was common with string instruments, especially when playing in *pizzicato*.

Therefore, after splitting the files, we listened to each of them, and manually corrected the wrong names by using as reference an electronic keyboard. This procedure sometimes introduced name conflicts (i.e., there were repeated notes played by the same instrument, same dynamic, etc.), and when this occurred, we removed the repeated notes trying to keep the closest note to the target. When the conflicting notes were equally close to the target, the "best quality" sound was preserved. This removal of files was done to avoid the overhead of having to add extra symbols to the file names to allow for repetitions, which would have complicated the generation of scripts to test the algorithms.

Since this process of manually correcting the names of the notes was very tedious, especially for the pizzicato sounds, after fixing all the pizzicato bass and violin notes, the process was abandoned and the cello and viola pizzicato sounds were excluded from our evaluation. Arguably, except for the bass, pizzicato sounds are not very common in music, and therefore leaving the cello and viola pizzicato sounds out did not affect the representativeness of the sample significantly.

B.2 Evaluation Using Speech

Whenever possible, each of the algorithms was asked to give a pitch estimate every millisecond within the range 40-800 Hz, using the default settings of the algorithm (an exception was made for ESRPD: instead of using the default settings in the Festival implementation, the recommendations suggested by the author of the algorithm were followed). The range 40-800 was used to make the results comparable to the results published by de Cheveigne (2002). However, a full comparison is not possible since some other variables were treated differently in that study.

The commands issued for each of the algorithms were the following[6]:

- **AC-P**: To Pitch (ac)... 0.001 40 15 no 0.03 0.45 0.01 0.35 0.14 800
- **AC-S**: fxac input_file
- **ANAL**: fxanal input_file
- **CC**: To Pitch (cc)... 0.001 40 15 no 0.03 0.45 0.01 0.35 0.14 800
- **CEP**: fxcep input_file
- **ESRPD**: pda input_file -o output_file -L -d 1 -shift 0.001 -length 0.0384 -fmax 800 -fmin 40 -lpfilter 600
- **RAPT**: fxrapt input_file
- **SHS**: To Pitch (shs)... 0.001 40 15 1250 15 0.84 800 48
- **SHR**: [t, p] =shrp(x, fs, [40 800], 40, 1, 0.4, 1250, 0, 0);
- **SWIPE**: [p, t] = swipe(x, fs, [40 800], 0.001, 1/96, 0.1, -Inf);
- **SWIPE′**: [p, t] = swipep(x, fs, [40 800], 0.001, 1/96, 0.1, -Inf);
- **TEMPO**: f0raw = exstraightsource(x, fs);
- **YIN**: p.minf0 = 40; p.maxf0 = 800; p.hop = 20; p.sr = fs; r = yin(x, p);

where x is the input signal and f_s is the sampling rate in Hertz.

An important issue that had to be considered was the time associated to each pitch estimate. Since all algorithms use symmetric windows, a reasonable choice was to associate each estimate to the time at the center of the window. For CATE, ESRPD, and SHR, the user is allowed to determine the size of the window, so we followed the recommendation of their authors and we set the window sizes to 51.2, 38.4, and 40 ms, respectively. YIN uses a different window size for each pitch candidate, but the windows are always centered at the same time instant, and the largest window size is two periods of the largest expected pitch period. For the Praat's algorithms AC-P, CC, and SHS, through trial and error we found that they use windows of size 3, 1, and 2 times the largest expected pitch period, respectively. For AC-S, ANAL, CEP, RAPT, and TEMPO, the user is not allowed to set up the window size, but the algorithms output the time instants associated to each pitch estimate, so we used these times hoping that they correspond to the centers of the analysis windows used to determine the pitch.

[6] The command for CATE is not reported because we used our own implementation.

The times associated to the pitch ground truth series are explicitly given in the PBD database, but not in the KPD database. For KPD, each pitch value was associated to the center of the window. Therefore, since the ground truth pitch values were computed using 26.5 msec windows separated at a distance of 10 msec, the first pitch estimate was assigned a time of 13.25 msec, and the time associated to each successive pitch estimate added 10 msec to the time of the previous estimate. For the DVD databases, each vowel was assumed to have a constant pitch, so the ground truth pitch time series was assumed to be constant.

The purpose of the evaluation was to compare the pitch estimates of the algorithms, but not their ability to distinguish the existence of pitch. Therefore, we included in the evaluation only the regions of the signal at which all algorithms and the ground truth data agreed that pitch existed. To achieve this, we took the time instants of the ground truth values and the time instants produced by all the algorithms that estimated the pitch every millisecond (9 out of 13 algorithms), rounded them to the closest multiple of 1 millisecond, and took the intersection. This intersection would form the set of times at which all the algorithms would be evaluated. The algorithms that produced pitch estimates at a rate lower than 1,000 per second were not considered for finding the intersection because that would reduce the time granularity of our evaluation, which was desired to be one millisecond.

As suggested in the previous paragraph, some algorithms do not necessarily produce pitch estimates at times that are multiples of one millisecond, i.e., they may produce the estimates at the times $t + \Delta t$ ms, where t is an integer and $|\Delta t| < 1$. Thus, to evaluate them at multiples of one millisecond, the pitch values at the desired times were inter/extrapolated in a logarithmic scale. In other words, we took the logarithm of the estimated pitches, inter/extrapolated them to the desired times, and took the exponential of the inter/extrapolated pitches. Inter/extrapolation in the logarithmic domain was preferred because we believe this is the natural scale for pitch. This is what allows us to recognize a song even if it is sung by a male or a female.

An important issue that must be considered when using simultaneous recordings of the laryngograph and speech signals is that the latter are typically delayed with respect to the former. An attempt to correct this misalignment was reported by the authors of KPD, but the success was not warranted. No attempt of correction was reported for PBD. Since pitch in speech is time-varying, such misalignment could increase the estimation error significantly. To alleviate this problem, each pitch time series produced by each algorithm was delayed or advanced, in steps of 1 msec, and up to 100 msec, in order to find the best match with the ground truth data.

B.3 Evaluation Using Musical Instruments

Considering that many algorithms were designed for speech, the pitch range of the MIS database is probably too large for them to handle. To alleviate this, we excluded the samples that were outside the range 30-1666 Hz, which is nevertheless large, compared to the pitch range of speech. Since the range 30-1666 Hz was found to be too large for the Speech Filing System algorithms (AC-S, ANAL, CEP, and RAPT) these algorithms were not evaluated on the MIS database. The commands issued for each of the algorithms were the following:

- **AC-P**: To Pitch (ac)... 0.001 30 15 no 0.03 0.45 0.01 0.35 0.14 1666
- **CC**: To Pitch (cc)... 0.001 30 15 no 0.03 0.45 0.01 0.35 0.14 1666
- **ESRPD**: pda input_file -o output_file -P -d 1 -shift 0.001 -length 0.0384 -fmax 1666 -fmin 30 -n 0 -m 0
- **SHS**: To Pitch (shs)... 0.001 30 15 5000 15 0.84 1666 48
- **SHR**: [t, p] = shrp(x, fs, [30 1666], 40, 1, 0.4, 5000, 0, 0);
- **SWIPE**: [p, t] = swipe(x, fs, [30 1666], 0.001, 1/96, 0.1, -Inf);
- **SWIPE′**: [p, t] = swipep(x, fs, [30 1666], 0.001, 1/96, 0.1, -Inf);
- **YIN**: p.minf0 = 30; p.maxf0 = 1666; p.hop = 10; p.sr = 10000; r = yin(x,p);

Besides the widening of the pitch range, the only difference with respect to the commands used for the speech databases were for ESRPD and SHS. For both of them, the low-pass filtering was removed in order to use as much information from the spectrum as possible. This was convenient because the sounds were already low-pass filtered at 5 kHz, and therefore the highest pitch sounds (around 1666 Hz) had no more than three harmonics in the spectrum. The second change was the use of the ESRPD peak-tracker (option -P) as an attempt to make the algorithm improve upon its results with speech.

The evaluation process was very similar to the one followed for speech: the time instants of the ground truth and the pitch estimates were rounded to the closest millisecond, the intersection of all the times was taken, and the statistics were computed only at the times of this intersection. However, there was an issue that was necessary consider in this database. Some instruments played much longer notes than others. The range of durations goes from tenths of second for strings playing in *pizzicato*, to several seconds for some notes of the piano. If the overall error is computed without taking this into account, the results will be highly biased toward the performance produced with the instruments that play the largest notes.

To account for this, the GER was computed independently for each sample, and then averaged over all the samples. However, this introduced an undesired effect: some samples had very few pitch estimates (only one estimate in some cases), and therefore this procedure would give them too much weight, which potentially would introduce noise in our results. Therefore, we discarded the samples for which the time instants at which the algorithms were evaluated were less than half the duration of the sample (in milliseconds). This discarded 164 samples, resulting in a total of 3459 samples, which was nevertheless a significant amount of data to quantify the performance of the algorithms.

APPENDIX C
GROUND TRUTH PITCH FOR THE DISORDERED VOICE DATABASE

Table C-1. Ground truth pitch values for the disordered voice database

AAK02	220.0	AAS16	123.5	ABB09	246.9	ABG04	116.5	ACG13	207.7	ACG20	164.8
ACH16	185.0	ADM14	138.6	ADP02	155.6	ADP11	116.5	AEA03	220.0	AFR17	246.9
AHK02	110.0	AHS20	196.0	AJF12	110.0	AJM05	138.6	AJM29	123.5	AJP25	233.1
ALB18	123.5	ALW27	174.6	ALW28	220.0	AMB22	146.8	AMC14	92.5	AMC16	146.8
AMC23	196.0	AMD07	130.8	AMJ23	123.5	AMK25	77.8	AMP12	220.0	AMT11	246.9
AMV23	185.0	ANA15	155.6	ANA20	155.6	ANB28	196.0	AOS21	110.0	ASK21	116.5
ASR20	92.5	ASR23	130.8	AWE04	155.6	AXD11	174.6	AXD19	196.0	AXL04	196.0
AXL22	196.0	AXS08	155.6	AXT11	185.0	AXT13	196.0	BAH13	98.0	BAS19	293.7
BAT19	185.0	BBR24	164.8	BCM08	233.1	BEF05	185.0	BGS05	246.9	BJH05	174.6
BJK16	174.6	BJK29	103.8	BKB13	87.3	BLB03	110.0	BMK05	246.9	BMM09	233.1
BPF03	116.5	BRT18	311.1	BSD30	130.8	BSG13	174.6	BXD17	138.6	CAC10	185.0
CAH02	196.0	CAK25	196.0	CAL12	92.5	CAL28	261.6	CAR10	196.0	CBD17	164.8
CBD19	174.6	CBD21	207.7	CBR29	174.6	CCM15	110.0	CDW03	146.8	CEN21	92.5
CER16	185.0	CER30	174.6	CFW04	155.6	CJB27	116.5	CJP10	98.0	CLE29	116.5
CLS31	185.0	CMA06	123.5	CMA22	103.8	CMR01	185.0	CMR06	110.0	CMR26	174.6
CMS10	196.0	CMS25	185.0	CNP07	196.0	CNR01	185.0	CPK19	155.6	CPK21	174.6
CPW28	220.0	CRM12	185.0	CSJ16	233.1	CSY01	110.0	CTB30	146.8	CTY03	130.8
CXL08	174.6	CXM07	130.8	CXM14	220.0	CXM18	146.8	CXP02	207.7	CXR13	146.8
CXT08	155.6	DAC26	155.6	DAG01	185.0	DAM08	174.6	DAP17	130.8	DAS10	146.8
DAS24	146.8	DAS30	87.3	DAS40	77.8	DBA02	220.0	DBF18	155.6	DBG14	103.8
DFB09	233.1	DFS23	293.7	DFS24	293.7	DGL30	207.7	DGO03	110.0	DHD08	123.5
DJF23	146.8	DJM14	130.8	DJM28	185.0	DJP04	110.0	DLB25	261.6	DLL25	174.6
DLT09	207.7	DLW04	130.8	DMC03	185.0	DMF11	293.7	DMG07	146.8	DMG24	196.0
DMG27	155.6	DMP04	123.5	DMR27	233.1	DMS01	146.8	DOA27	92.5	DRC15	196.0
DRG19	116.5	DSC25	277.2	DSW14	138.6	DVD19	164.8	DWK04	130.8	DXS20	123.5
EAB27	164.8	EAL06	207.7	EAS11	110.0	EAS15	138.6	EAW21	207.7	EBJ03	146.8
EDG19	196.0	EEB24	164.8	EEC04	196.0	EED07	554.4	EFC08	130.8	EGK30	196.0
EGT03	138.6	EGW23	220.0	EJB01	92.5	EJM04	123.5	ELL04	116.5	EMD08	82.4
EML18	370.0	EMP27	174.6	EOW04	164.8	EPW04	164.8	EPW07	123.5	ERS07	185.0
ESL28	207.7	ESM05	138.6	ESP04	138.6	ESS05	174.6	ESS24	220.0	EWW05	174.6
EXE06	146.8	EXH21	185.0	EXI04	110.0	EXI05	116.5	EXS07	207.7	EXW12	164.8
FAH01	164.8	FGR15	130.8	FJL23	116.5	FLL27	207.7	FLW13	207.7	FMC08	196.0
FMM21	207.7	FMM29	207.7	FMQ20	155.6	FMR17	116.5	FRH18	146.8	FSP13	155.6
FXC12	110.0	FXE24	196.0	FXI23	103.8	GCU31	123.5	GEA24	130.8	GEK02	138.6
GJW09	174.6	GLB01	77.8	GLB22	98.0	GMM06	196.0	GMM07	207.7	GMS03	110.0
GMS05	261.6	GMW18	146.8	GRS20	110.0	GSB11	164.8	GSL04	116.5	GTN21	130.8
GXL21	196.0	GXT10	155.6	GXX13	164.8	HBS12	196.0	HED26	123.5	HJH07	130.8
HLC16	110.0	HLK01	116.5	HLK15	130.8	HLM24	138.6	HMG03	185.0	HML26	207.7
HWR04	164.8	HXB20	196.0	HXI29	82.4	HXL58	116.5	HXR23	116.5	IGD08	196.0
IGD16	174.6	JAB08	130.8	JAB30	164.8	JAF15	146.8	JAJ10	207.7	JAJ22	155.6
JAJ31	155.6	JAL05	174.6	JAM01	207.7	9-Jan	130.8	JAP02	138.6	JAP17	174.6
JAP25	174.6	JBP14	98.0	JBR26	110.0	JBS17	82.4	JBW14	130.8	JCC08	164.8
JCC10	207.7	JCH13	110.0	JCH21	116.5	JCL12	174.6	JCL20	146.8	JCR01	233.1
JDM04	110.0	JEG29	246.9	JES29	123.5	JFC28	82.4	JFG08	138.6	JFG26	138.6
JFM24	174.6	JFN11	110.0	JFN21	116.5	JHW29	146.8	JIJ30	146.8	JJD06	174.6
JJD11	185.0	JJD29	138.6	JJI03	110.0	JJM28	220.0	JLC08	185.0	JLD24	233.1
JLH03	174.6	JLM18	207.7	JLM27	123.5	JLS11	130.8	JLS18	138.6	JMC18	138.6
JME23	164.8	JMH22	155.6	JMJ04	207.7	JMZ16	196.0	JOP07	130.8	JPB07	98.0
JPB17	164.8	JPB30	98.0	JPM25	110.0	JPP27	207.7	JRF30	123.5	JRP20	110.0
JSG18	207.7	JTM05	87.3	JTS02	103.8	JWE23	185.0	JWK27	98.0	JWM15	116.5
JXB16	110.0	JXB26	116.5	JXC21	220.0	JXD01	138.6	JXD08	138.6	JXD30	123.5
JXF11	246.9	JXF29	103.8	JXG05	138.6	JXM30	146.8	JXS09	110.0	JXS14	146.8
JXS23	98.0	JXS39	146.8	JXZ11	123.5	KAB03	185.0	KAC07	246.9	KAO09	261.6
KAS09	233.1	KAS14	220.0	KCG23	246.9	KCG25	220.0	KDB23	220.0	KEP27	87.3

Table C-1. Continued

KEW22	220.0	KGM22	220.0	KJB19	164.8	KJI23	138.6	KJI24	130.8	KJL11	116.5
KJM08	130.8	KJS28	207.7	KJW07	103.8	KLC06	207.7	KLD26	164.8	KMC19	207.7
KMC22	207.7	KMC27	207.7	KMS29	155.6	KMW05	311.1	KPS25	103.8	KTJ26	220.0
KWD22	185.0	KXA21	164.8	KXB17	246.9	KXH19	246.9	LAC02	164.8	LAD13	130.8
LAI04	174.6	LAP05	116.5	LAR05	116.5	LBA24	220.0	LCW30	196.0	LDJ11	82.4
LGK25	110.0	LGM01	185.0	LHL08	207.7	LJH06	207.7	LJM24	196.0	LJS31	220.0
LLM22	277.2	LMB18	116.5	LMM04	185.0	LMM17	196.0	LMP12	196.0	LNC11	98.0
LPN14	146.8	LRD21	116.5	LRM03	293.7	LSB18	174.6	LVD28	261.6	LWR18	220.0
LXC01	207.7	LXC11	207.7	LXC28	207.7	LXD22	207.7	LXG17	116.5	LXR15	103.8
LXS05	196.0	MAB06	196.0	MAB11	146.8	MAC03	185.0	MAM08	207.7	MAM21	220.0
MAT26	261.6	MAT28	233.1	MBM05	155.6	MBM21	196.0	MBM25	185.0	MCA07	164.8
MCB20	174.6	MCW14	277.2	MCW21	196.0	MEC06	196.0	MEC28	174.6	MEH26	196.0
MEW15	246.9	MFC20	123.5	MGM28	220.0	MGV01	103.8	MHL19	138.6	MID08	174.6
MJL02	130.8	MJM04	207.7	MJZ18	196.0	MKL31	123.5	MLB16	196.0	MLC08	233.1
MLC23	174.6	MLF13	196.0	MLG10	233.1	MMD01	233.1	MMD15	233.1	MMG27	246.9
MMM12	246.9	MMR01	138.6	MMS29	130.8	MNH04	207.7	MNH14	261.6	MPB23	103.8
MPC21	207.7	MPF25	110.0	MPH12	220.0	MPS09	246.9	MPS21	233.1	MPS23	311.1
MPS26	220.0	MRB11	98.0	MRB25	98.0	MRB30	92.5	MRC20	174.6	MRM16	155.6
MRR22	174.6	MSM20	77.8	MWD28	110.0	MXC10	233.1	MXN24	233.1	MXS06	246.9
MXS10	233.1	MYW04	220.0	MYW14	207.7	NAC21	98.0	NAP26	92.5	NFG08	207.7
NGA16	116.5	NJS06	207.7	NLC08	185.0	NMB28	185.0	NMC22	233.1	NMF04	164.8
NML15	196.0	NMR29	123.5	NMV07	207.7	NXM18	185.0	NXR08	185.0	OAB28	69.3
ORS18	98.0	OWH04	233.1	OWP02	246.9	PAM01	92.5	PAT10	110.0	PCL24	110.0
PDO11	110.0	PEE09	185.0	PFM03	103.8	PGB16	110.0	PJM12	98.0	PLW14	207.7
PMC26	92.5	PMD25	130.8	PMF03	233.1	PSA21	155.6	PTO18	98.0	PTO22	98.0
PTS01	130.8	RAB08	185.0	RAB22	196.0	RAE12	110.0	RAM30	261.6	RAN30	261.6
RBC09	155.6	RBD03	155.6	RCC11	233.1	REC19	233.1	REW16	110.0	RFC19	233.1
RFC28	116.5	RFH18	155.6	RFH19	130.8	RGE19	82.4	RHG07	220.0	RHP12	196.0
RJC24	98.0	RJF14	164.8	RJF22	174.6	RJL28	92.5	RJR15	110.0	RJR29	116.5
RJZ16	185.0	RLM21	123.5	RMB07	98.0	RMC07	155.6	RMC18	196.0	RMF14	196.0
RML13	233.1	RMM13	246.9	RPC14	174.6	RPJ15	116.5	RPQ20	103.8	RSM20	130.8
RTH15	87.3	RTL17	87.3	RWC23	98.0	RWF06	146.8	RWR14	110.0	RWR16	116.5
RXG29	98.0	RXM15	110.0	RXP02	138.6	RXS13	130.8	SAC10	103.8	SAE01	164.8
SAM25	138.6	SAR14	207.7	SAV18	277.2	SBF11	207.7	SBF24	207.7	SCC15	138.6
SCH15	207.7	SEC02	196.0	SEF10	98.0	SEG18	130.8	SEH26	174.6	SEH28	246.9
SEK06	164.8	SEM27	116.5	SFD17	116.5	SFD23	87.3	SFM22	92.5	SGN18	138.6
SHC07	164.8	SHD17	220.0	SHT20	138.6	SJD28	123.5	SLC23	220.0	SLG05	196.0
SLM27	87.3	SMD22	207.7	SMK04	370.0	SMK23	146.8	SMW17	77.8	SPM26	92.5
SRB31	174.6	SRR24	130.8	SWB14	123.5	SWS04	155.6	SXC02	146.8	SXG23	174.6
SXH10	185.0	SXM27	196.0	SXS16	220.0	SXZ01	87.3	TAB21	174.6	TAC22	207.7
TAR18	155.6	TCD26	138.6	TES03	220.0	TLP13	233.1	TLS08	185.0	TMK04	261.6
TNC14	207.7	TPM04	155.6	TPP11	220.0	TPP24	185.0	TPS16	116.5	TRF06	116.5
TRF21	98.0	TRS28	185.0	VFM11	220.0	VJV02	130.8	VJV09	110.0	VMB18	174.6
VMS04	277.2	VMS05	246.9	VRS01	164.8	WBR12	277.2	WCB24	174.6	WDK04	110.0
WDK13	220.0	WDK17	130.8	WDK47	146.8	WFC07	116.5	WJB06	233.1	WJB12	110.0
WJF15	174.6	WJP20	123.5	WPB30	123.5	WPK11	110.0	WSB06	110.0	WST20	87.3
WTG07	130.8	WXE04	123.5	WXH02	103.8	WXS21	110.0	LME07	659.3	EAM05	146.8
JEC18	196.0	TMD12	349.2	SMA08	220.0	SHD04	349.2	KXH30	174.6	VAW07	174.6

REFERENCES

American Standards Association (**1960**). “Acoustical Terminology SI 1-1960” (American Standards Association, New York).

American National Standards Institute (**1994**). ANSI S1.1-1994, ‘‘American National Standard Acoustical Terminology’’ (Acoustical Society of America, New York).

Askenfelt, A. (**1979**). “Automatic notation of played music: the VISA project," Fontes Artis Musicae **26**, 109-118.

Bagshaw, P. C., Hiller, S. M., and Jack, M. A. (**1993**). “Enhanced pitch tracking and the processing of *F*0 contours for computer and intonation teaching,” Proc. European Conf. on Speech Comm. (Eurospeech), pp. 1003–1006.

Bagshaw, P. C., (**1994**), “Automatic prosodic analysis for computer aided pronunciation teaching”, doctoral dissertation.

Boersma, P. (**1993**). “Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound,” Proc. Institute of Phonetic Sciences **17**, 97-110.

Dannenberg, R. B., Birmingham, W. P., Tzanetakis, G. P., Meek, C. P., Hu, N. P., and Pardo, B., P. (2004). “The MUSART Testbed for Query-by-Humming Evaluation,” Comp. Mus. J. 28, 34-48.

de Boer, E. (**1976**). “On the ‘residue’ and auditory pitch perception," in Handbook of Sensory Physiology, edited by W. D. Keidel and W. D. Neff (Springer-Verlag, New York), Vol. V/3, 479–583.

De Bot, K. (**1983**). “Visual feedback of intonation I: Effectiveness and induced practice behavior,” Language and Speech **26**, 331-350.

De Cheveigné, A., Kawahara, H. (**2002**). “YIN, a fundamental frequency estimator for speech and music,” J. Acoust. Soc. Am. **111**, 1917-1930.

Di Martino, J., Laprie, Y. (**1999**): “An efficient *F*0 determination algorithm based on the implicit calculation of the autocorrelation of the temporal excitation signal,” Proc. EUROSPEECH, 2773-2776.

Doughty, J., and Garner, W. (**1947**). “Pitch characteristics of short tones. I. Two kinds of pitch threshold,” J. Exp. Psychol. **37**, 351–365

Duifhuis, H., Willems, L. F., and Sluyter, R. J. (**1982**). “Measurement of pitch in speech: an implementation of Goldstein's theory of pitch perception,” J. Acoust. Soc. Am. **71**, 1568-1580.

Fant, G. (**1960**). *Acoustic theory of speech production, with calculations based on X-ray studies of Russian articulations*. (Mouton De Gruyter).

Fastl, H., and Stoll G. (**1979**). “Scaling of pitch strength,” Hear. Res. **1**, 293-301.

Fastl, H., and Zwicker, E. (**2007**). *Psychoacoustics: Facts and Models* (Springer, Berlin).

Galembo, A., Askenfelt, A., Cuddy, L. L., and Russo, F. A. (**2001**). “Effects of relative phases on pitch and timbre in the piano bass range,” J. Acoust. Soc. Am. **110**, 1649–1666.

Glasberg, B. R., and Moore, B. C. J. (**1990**). “Derivation of auditory filter shapes from notched-noise data,” Hear. Res. **47**, 103-138.

Hermes, D. J. (**1988**). “Measurement of pitch by subharmonic summation,” J. Acoust. Soc. Am. **83**, 257-264.

Houtgast, T. (**1976**). "Subharmonic pitches of a pure tone at low S/N ratio," J. Acoust. Soc. Am. **60**, 405–409.

Houtsma, A. J. M., and Smurzynski, J. (**1990**). "Pitch identification and discrimination for complex tones with many harmonics," J. Acoust. Soc. Am. **87**, 304–310.

Kawahara, H., Katayose, H., de Cheveigné, A., and Patterson, R. D. (**1999**). ‘‘Fixed Point Analysis of Frequency to Instantaneous Frequency Mapping for Accurate Estimation of F0 and Periodicity,’’ Proc. EUROSPEECH **6**, 2781–2784.

Medan, Y., Yair, E., and Chazan, D. (**1991**). ‘‘Super resolution pitch determination of speech signals,’’ IEEE Trans. Acoust., Speech, Signal Process. **39**, 40–48.

Moore, B. C. J. (**1977**). ‘‘Effects of relative phase of the components on the pitch of three-component complex tones,’’ in *Psychophysics and Physiology of Hearing*, edited by E. F. Evans and J. P. Wilson (Academic, London).

Moore, B. C. J. (**1986**). “Parallels between frequency selectivity measured psychophysically and in cochlear mechanics,” Scand. Audiol. Supplement **25**, 139-152.

Moore, B. C. J. (**1997**). *An Introduction to the Psychology of Hearing* (Academic, London).

Murray, I. R., and Arnott, J., L. (**1993**). “Toward the simulation of emotion in synthetic speech: A review of the literature of human vocal emotion,” J. Acoust. Soc. Am. **93**, 1097-1108.

Noll, A. M. (**1967**). “Cepstrum pitch determination,” J. Acoust. Soc. Am. **41**, 293-309.

Oppenheim, A. V., Schafer, R. W. and Buck, J. R. (**1999**). *Discrete-Time Signal Processing* (Prentice Hall, New Jersey).

Patel, A. D., and Balaban, E. (**2001**). “Human pitch perception is reflected in the timing of stimulus-related cortical activity,” Nature Neuroscience **4**, 839-844.

Plante, F., Meyer, G., and Ainsworth, W. A. (**1995**). “A pitch extraction reference database," EUROSPEECH-1995, 837--840.

Rabiner, L.R. (**1977**), “On the Use of Autocorrelation Analysis for Pitch Detection,” IEEE Trans. Acoust., Speech, Signal Process. **25**, 24-33.

Robinson, K.L. and Patterson, R.D. (**1995a**) “The duration required to identify the instrument, the octave, or the pitch-chroma of a musical note,” Music Perception **13**, 1-15.

Robinson, K.L. and Patterson, R.D. (**1995b**) "The stimulus duration required to identify vowels, their octave, and their pitch-chroma," J. Acoust. Soc. Am. **98**, 1858-1865.

Schroeder, M. R. (**1968**). “Period histogram and product spectrum: new methods for fundamental frequency measurement,” J. Acoust. Soc. Am. **43**, 829-834.

Schwartz, D. A., and Purves, D. (**2004**). “Pitch is determined by naturally occurring periodic sources,” Hear. Res. **194**, 31-46.

Secrest, B., and Doddington, G. (**1983**) “An integrated pitch tracking algorithm for speech systems,” Proc. ICASSP-83, pp. 1352-1355.

Sethares, W. A. (**1998**). *Tuning, Timbre, Spectrum, Scale* (Springer, London).

Shackleton, T. M. and Carlyon, R. P. (**1994**). "The role of resolved and unresolved harmonics in pitch perception and frequency modulation discrimination," J. Acoust. Soc. Am. **95**, 3529-3540.

Shofner, W. P., and Selas, G. (**2002**), “Pitch strength and Stevens’s power law,” Percept Psychophys. **64**, 437-450.

Sondhi, M. M. (**1968**), “New Methods of Pitch Extraction,” IEEE Trans. Audio and Electroacoustics **AU-16**, 262-266.

Spanias, A. S. (**1994**). “Speech coding: a tutorial review,” Proc. IEEE, **82**, 1541-1582.

Stevens, S.S. (**1935**), “The relation of pitch to intensity,” J. Acoust. Soc. Am. **6**, 150–154.

Sun, X. (**2000**). “A pitch determination algorithm based on subharmonic-to-harmonic ratio,” Proc. Int. Conf. Spoken Language Process. **4**, 676-679.

Takeshima, H., Suzuki, Y., Ozawa, K., Kumagai, M., and Sone, T. (**2003**). ‘‘Comparison of loudness functions suitable for drawing equal-loudness level contours,’’ Acoust. Sci. Tech. **24**, 61–68.

Verschuure, J., van Meeteren A.A. (**1975**). “The effect of intensity on pitch,” Acustica **32**, 33–44.

von Helmholtz, H. (**1863**). *On the Sensations of Tone as a Physiological Basis for the Theory of Music* (Kessinger Publishing).

Wang, M. and Lin, M. (**2004**). “An Analysis of Pitch in Chinese Spontaneous Speech,” Int. Symp. on Tonal Aspects of Tone Languages, Beijing, China.

Wiegrebe, L., and Patterson, R. D. (**1998**). “Temporal dynamics of pitch strength in regular interval noises,” J. Acoust. Soc. Am. **104**, 2307-2313.

Huang, X. Acero, A., and Hon, H. W. (**2001**) *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development* (Prentice Hall, New Jersey).

Yost, W. A. (**1996**). “Pitch strength of iterated rippled noise,” J. Acoust. Soc. Am. **100**, 3329-3335.

Yumoto E., Gould W. J., Baer T. (1982). “Harmonics-to-noise ratio as an index of the degree of hoarseness,” J. Acoust. Soc. Am. **71**, 1544-1549.

BIOGRAPHICAL SKETCH

Arturo Camacho was born in San Jose, Costa Rica, on October 21, 1972. He did his elementary school at Centro Educativo Roberto Cantillano Vindas and his high school at Liceo Salvador Umaña Castro. After that, he studied Music at the Universidad Nacional, and at the same time he performed as pianist in some of the most popular Costa Rican Latin music bands. He also studied Computer and Information Science at the Universidad de Costa Rica, where he obtained his B.S. degree in 2001. He worked for a short time as a software engineer in Banco Central de Costa Rica during that year, but soon he moved to the United States to pursue graduate studies in Computer Engineering at the University of Florida. He received his M.S. and Ph.D. degrees in 2003 and 2007, respectively.

Arturo's research interests span all areas of automatic music analysis, from the lowest level tasks like pitch estimation and timbre identification, to the highest levels tasks like analysis of harmony and gender. His dream is to have one day a computer program that allows him (and everyone) to analyze music as well or better than a well-trained musician would do.

Currently, Arturo lives happily with his loved wife Alexandra, who is another Ph. D. gator in Computer Engineering and who he married in 2002, and their loved daughter Melissa, who was born in 2006.