D I P L O M A T H E S I S

Improving the Estimation of Trajectories of Partials for Additive Synthesis by the Use of the Ambiguity Function

Matthias Geier

June 2006

Supervisors:
Philippe Depalle Robert Höldrich


Schulich School of Music
McGill University, Montreal, Quebec, Canada

Institute of Electronic Music and Acoustics
University of Music and Dramatic Arts, Graz, Austria

University of Technology, Graz, Austria

Abstract

The *Additive Analysis/Synthesis* model represents sounds as the superposition of sinusoidal components with time-evolving parameters. Crucial parts of this technique are, among others, the estimation of the parameters (instantaneous frequency and amplitude and the initial phase of a single partial tone at a given time) and the tracking of the temporal evolution of corresponding partials by means of these parameters.

In this thesis we use the *Ambiguity Function* (a member of the family of *bilinear time-frequency distributions*) for the estimation of frequency and amplitude. The initial phase is calculated from the *Analytic Signal*. Finally, we show how a *Hidden Markov Model* can be used for the tracking of partials. In addition to the aforementioned parameters, the Ambiguity Function provides us with an estimator for the *chirp rate*, which can be used to improve the tracking of partials.

A prototype application developed for this thesis using MATLAB® shows an improvement of the estimators and the assignment of partials to trajectories compared to more traditional methods.

Zusammenfassung

Additive Analyse/Synthese modelliert Klänge als Überlagerung von Sinuskomponenten mit zeitvarianten Parametern. Zwei essenzielle Teilgebiete dieser Methode sind die Schätzung der Parameter (Frequenz, Amplitude und Initialphase zu einem gegebenen Zeitpunkt) und die zeitliche Verfolgung von zusammengehörigen Teiltönen mithilfe dieser Parameter.

In der vorliegenden Arbeit wird die *Ambiguitätsfunktion* (eine *bilineare Zeit-Frequenz Distribution*) verwendet, um Frequenz und Amplitude zu schätzen. Die Phase wird aus dem *analytischen Signal* berechnet. Außerdem wird gezeigt, wie man ein *Hidden-Markov Modell* zur Verfolgung von Partialtönen verwenden kann. Die Ambiguitätsfunktion liefert zusätzlich zu den genannten Parametern noch einen Schätzer für die *Chirp-Rate*, der die Bestimmung der Teiltontrajektorien verbessert.

Eine im Rahmen dieser Arbeit entwickelte MATLAB®-Applikation zeigt eine Verbesserung der Schätzer und der Trajektorienzuordnung verglichen mit traditionelleren Methoden.

Contents

Introduction

Additive synthesis is one of the oldest synthesis techniques in electronic music. It is in most cases used together with an automated analysis process which analyzes an input sound and provides a set of parameters which evolve over time and which represent the partial tones contained in the sound. These parameters can then be processed at will, can be altered in pitch, intensity and duration, merged with parameters of another sound, gradually morphed into the parameters of yet another sound and so on. A set of such parameters can then be used to synthesize a new sound.

There is not a single way to realize the parameter extraction mentioned above. In the history of additive synthesis several techniques were proposed and most likely a few will also be introduced in the future. This thesis describes a recent method for the estimation of parameters of partial tones. It is based on the *Ambiguity Function*, which has already been used for decades in very different fields such as radar technology but not very commonly in audio signal processing. The Ambiguity Function is a *bilinear time-frequency distribution*, a two-dimensional function of time and frequency.

Once the parameters of the partial tones have been extracted (with whichever technique chosen), they are sorted into *trajectories* to model the evolution of the partial tones within the sound. This is also called *partial tracking*. In this thesis we describe several different ways to implement this tracking of partials.

The Ambiguity Function provides, additional to frequency and amplitude, an estimator for the *chirp rate* (the amount of which the frequency of a partial tone changes per unit time). With the aid of the chirp rate the assignment of partial tones to their respective trajectories can be enhanced.

One of the techniques for tracking partials involves the *Hidden Markov Model* technique, a statistical framework used in many different fields like speech recognition, optical character recognition and genomics. In the following paragraphs a brief summary is given for each chapter:

In chapter 1 a general overview is given about the *Additive Analysis/Synthesis* scheme. The main parts of this set of techniques are shown with an emphasis on the tasks being discussed in later chapters, namely the estimation of parameters and the tracking of partials. The *sinusoidal model* and the *sinusoidal + noise model*

are presented. The main focus is on the analysis part; the synthesis part is only briefly mentioned.

Chapter 2 deals with the *Fourier transform* and with a time-frequency representation traditionally used in spectral estimation of audio data, namely the *Short Time Fourier Transform*, and its shortcomings in the current context.

Chapter 3 tries to ensure an unambiguous use of the terms *instantaneous frequency* and *instantaneous amplitude* and presents a means of determining these parameters in a mono-component signal, the *Analytic Signal*. We will recall that the *Hilbert transform* can be used to obtain an analytic signal from an arbitrary real signal.

In chapter 4, the concept of *Bilinear Time-Frequency Distributions* is presented. One of them (the *Ambiguity Function*) will be used later on to improve the estimation of instantaneous frequency and amplitude of partial tones and it will additionally provide us with an estimator for the *chirp rate*, which in turn will be used for the tracking of partials.

Chapter 5 is dedicated to the description of the basic building blocks of the prototype application implemented for this thesis. It shows how the input signal is split into frames, how it is filtered, how the *Ambiguity Function* is used to estimate frequency, amplitude and chirp rate, how the estimated partials are tracked, et cetera.

Chapter 6 explains in detail the estimation of the parameters from the *Ambiguity Function* and the estimation of the initial phase by means of the *analytic signal*.

In chapter 7, *Hidden Markov Models* are presented, including the three kinds of problems that can be solved in this context. The *Viterbi algorithm*, a solution to the second problem, is explained.

Chapter 8 shows how partial tones can be tracked based on the parameters estimated earlier. First, a method which acts locally between frames is presented, later on it is shown how a *Hidden Markov Model* can be used to globally optimize partial trajectories.

Chapter 9 shows how the presented algorithms perform in various circumstances. Examples using artificial signals but also sound recordings of single instruments are presented.

Finally, chapter 10 summarizes the main ideas of this thesis and lists some aspects which should be improved in future works.

The appendix shows a few calculations (concerning the *Ambiguity Function*) which are displayed in such detail that it would distract the reader from the point made in the respective chapter in the main part of the thesis. If one wants to recalculate the equations, however, it should be an easy task with the calculation steps given in the appendix.

The appendices also hold a section about implementational details which are related to the actual situation and to the discretization of some concepts which are mainly presented for continuous variables. However, they are not needed for the understanding of the general ideas.

Furthermore, a little section describes an analysis/synthesis technique called *Spectral Modeling Synthesis*, which was used during research for this thesis but was later replaced by our own MATLAB® code.

In the very end, we provide the reader with a list of the numerous abbreviations used throughout the text (except in this introduction), a list of mathematical symbols and, finally, the bibliography.

Chapter 1

Additive Analysis/Synthesis

Among the different sound processing techniques, chained analysis-synthesis of sound signals is very often used in music, but also in sound engineering, audio coding and recording. The basic goal of analysis/synthesis is to conceive relevant models to represent acoustical signals by a set of temporal functions (the control parameters). These temporal functions are extracted from a pre-recorded sound during the analysis process. Re-synthesis produces a signal that sounds perceptually identical to the original sound. By modifying and substituting control parameters, this analysis-synthesis scheme can provide for very refined and precise processing of sound material. For example, it may allow one to produce a family of synthetic sound signals derived from a single original one or to carry out a morphing between two key sounds.

Since the synthesis procedure consists of adding up the sinusoidal waveforms for each of the interpolated amplitudes, frequencies and phases, the procedure is particularly suited for performing time-scale modification by simply expanding or compressing the frequency tracks. The instantaneous frequency locations and magnitudes are preserved while modifying their rate of change in time.

Here we will briefly present the *Sinusoidal Model* and thereafter an extension, the *Sinusoidal + Noise Model*.

The analysis process is frame based, meaning that the signal is analyzed at a particular moment in time, providing an undersampled version of the control parameters, the analysis location is then advanced through the audio by using a sliding window, and the process is repeated.

1.1 Sinusoidal Representation

This representation is generally associated with the contribution by McAulay and Quatieri [MQ86], but there are several related techniques proposed by different authors. Here we try to keep the description of the model as general as possible.

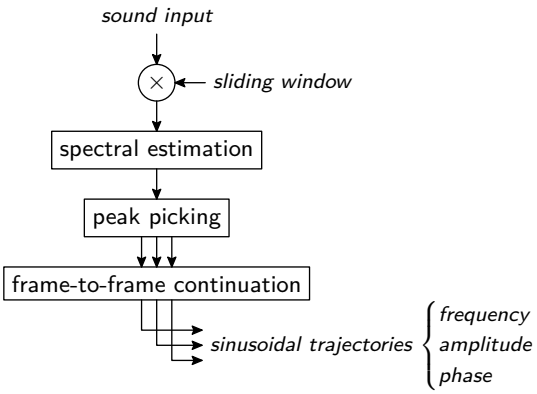

Figure 1.1: *The Sinusoidal Analysis Scheme*

The Sinusoidal Model is based on the assumption that a sound is composed of a deterministic component (i. e. sinusoids) with slowly varying frequency, amplitude and phase parameters. This can be stated mathematically as

$$s(t) = \mathcal{R}e\left\{\sum_{r=1}^{R} a_r(t)\, \mathrm{e}^{j\psi_r(t)}\right\}, \quad \text{where } \psi_r(t) = 2\pi \int_0^t f_r(\sigma)\, d\sigma + \phi_r, \tag{1.1}$$

with $f_r(t)$ the frequency track of the r^{th} sinusoidal component (out of a total number of R). ϕ_r represents a fixed phase offset (at time $t = 0$) which accounts for the initial state of each sine wave. ϕ_r is often called the *initial phase*.

The goal of the analysis process is to find the sinusoidal components within the sound and to determine the temporal function of the control parameters $a_r(t)$ and $\psi_r(t)$. As the control parameters are changing comparatively slowly they do not need to be determined for each audio sample. It is sufficient to split the signal under analysis into frames and to calculate the control parameters a_r, f_r and ϕ_r for each frame. As regards the synthesis, these sets of parameters changing at frame rate are interpolated to oversample the parameters at sampling rate.

The building blocks of the analysis based on the sinusoidal model, as shown in figure 1.1, are described below.

Windowing

As a first step, the time line is broken down into a contiguous sequence of frames. This can be done by simply cutting out a portion of the signal with the desired frame length, which in fact means using a rectangular window. However, as applying a window means convolving the spectrum of the signal with the window's spectrum, it is generally advisable to use a smoother window than the rectangular window in order to minimize the spectral spread.

The frame rate has to be chosen fast enough to avoid aliasing of the control parameters. To allow for a reasonably slow frame rate, one of the preconditions

for the additive synthesis model is a slow variation of the parameters frequency and amplitude.

Spectral Estimation

The purpose of this operation is to produce a general characterization of the frequency content of the signal within the frame. Frequency estimations of the sine waves will be extracted from this characterization and, at a later step of the process, manipulated to produce an overall time-varying spectral representation of the signal.

In many implementations the Short Time Fourier Transform (STFT) is used to calculate the spectrum. In this case, the windowing process mentioned above is constitutive of the STFT definition and computation (see section 2.2).

Peak Picking

In this step local maxima of the magnitude of the spectrum are determined. These show likely positions of the partial tones at a given time and form a set of so called *partial candidates*. Which of these candidates are finally selected to represent the partials, is determined in the following step, the peak tracking.

When, as in many practical implementations, a discrete Fourier transform is used for the spectral estimation, the frequency resolution is limited to discrete steps of sampling frequency divided by the number of Fourier transform bins. To allow for more precise frequency values between these given steps, interpolation can be used.

Also a subset of local maxima can be selected including only a given number of frequencies with the highest corresponding amplitudes or including only partial candidates with an amplitude bigger than a given threshold. Or only the partial candidates inside a given frequency range can be taken into account. It is desirable to have a robust parameter extraction algorithm since the signal in many cases is contaminated by "additive acoustic noise".

Peak Continuation

Once every spectral peak has been analyzed, the frequencies, amplitudes and initial phases of all partial candidates are passed to a *peak continuation* step. It is in the peak continuation step that this collection of instantaneous estimations will be layered into a coherent representation of partials.

The role of the peak continuation step is to identify the partials within the sound sample from the partial candidates. A partial is represented in what is called a *partial trajectory*. In the peak continuation step, partial candidates that are computed at the previous step are sorted into a set of appropriate partial trajectories. The resulting tracks are required at the synthesis stage, as synthesis parameters are interpolated along these tracks.

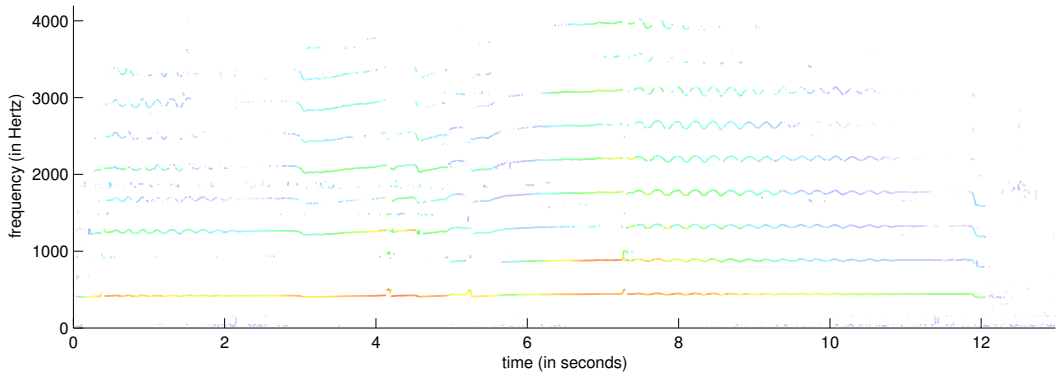

Figure 1.2: *Sinusoidal trajectories of the Shakuhachi recording from [WBF+00]. The red sections have the highest amplitudes.*

The number of peaks may change rapidly from frame to frame. To cope with that, a concept of "birth" and "death" of partial trajectories is introduced. When there are more partials in one frame than in the previous one it means that some partial tracks are "born". On the other hand, when there are less partials, some trajectories have "died". Also, when partials exhibit too large frequency difference to satisfy the continuation criteria it can mean the death of one track and the birth of another.

The continuation algorithm selects the two closest frequencies between two frames and if their difference is within a given threshold they are assigned to the same trajectory. This is repeated until the threshold is reached or until there are no more peaks left on one side. The remaining peaks in the first of the two frames have to "die", while the remaining peaks on the second frame are each assigned to a new trajectory.

The result is a collection of all partials present in the signal, where each partial is completely defined in terms of its individual frequencies, amplitudes and initial phases for each analysis frame. Figure 1.2 shows the sinusoidal tracks of a sound example, where the color of the trajectories represents the amplitude of the partials.

1.2 Sinusoidal + Noise Representation

A natural sound is normally not entirely representable by a purely sinusoidal model, especially if the number of partials shall be limited. The reason for that is that neither noise-like components nor transient events can be modeled accurately with a finite number of sinusoids. To account for that shortcoming, an

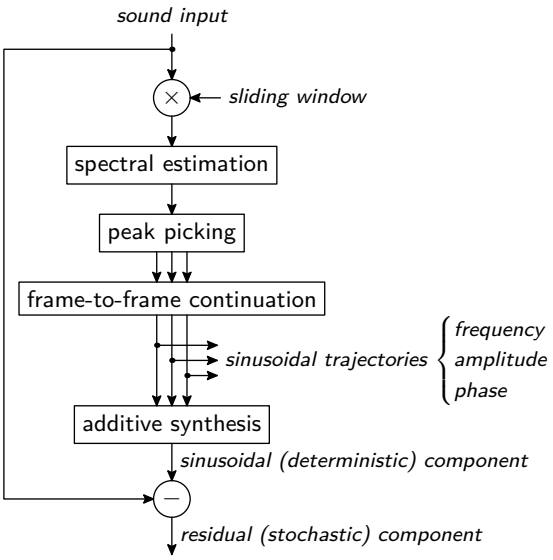

Figure 1.3: *The Deterministic + Stochastic Analysis Scheme*

extension of the McAulay/Quatieri model [MQ86] was introduced: The sinusoidal + noise model, also known as *Deterministic + Stochastic (D+S) Model* [SS90]. It is depicted schematically in figure 1.3. A well known implementation of this model is Spectral Modeling Synthesis (SMS). More information about SMS can be found in appendix C.

The D+S model can provide an excellent time-varying spectral description of a sound. The benefits of the separation of sinusoids from noise are twofold in that it provides maximum perceptual relevance to the signal, as well as maximum computational efficiency. Furthermore, the generality of the representation allows it to serve as the basis for many meaningful transformations.

The technique by which the sound is analyzed is a two part process of deterministic analysis and stochastic analysis. The deterministic analysis was already described in the previous section, the stochastic analysis does the following:

Calculation of the Residual Component

The sinusoidal tracks are resynthesized, generating the deterministic part of the representation. This deterministic component is then subtracted from the original input signal, which results in the stochastic component. When the analysis is accurate enough, this component does not hold any tonal information but only transient events and noise.

It is important that the synthesis uses the correct phase information because only with the correct initial phase a partial can be removed from the input sound by destructive interference. For a human listener, however, the phase information is not perceptible (except in transient parts).

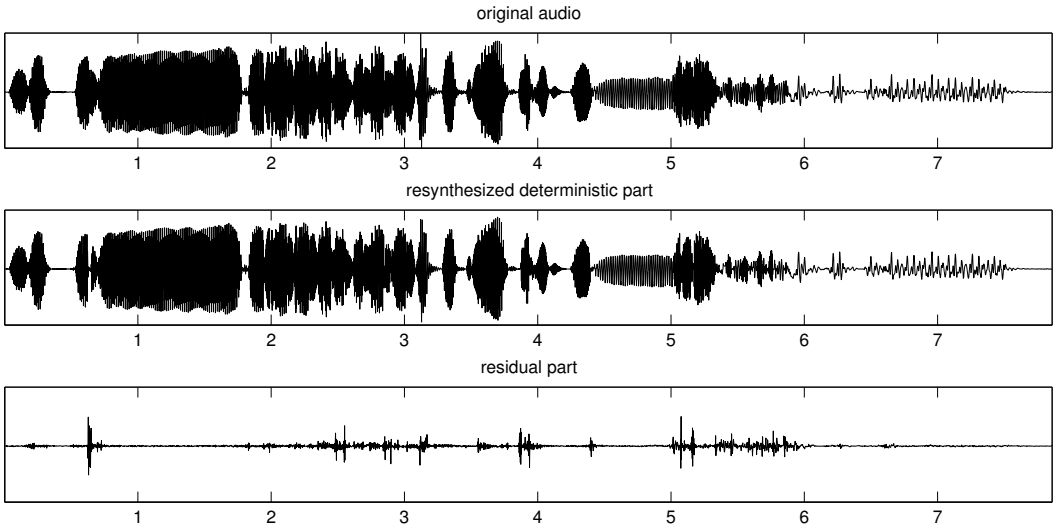

Figure 1.4: *Result of the resynthesis of the Suling flute recording from [WBF+00]. The residual part consists of modulated noise (mainly the breath of the performer) and some transient components caused by the articulation.*

Figure 1.4 shows the waveforms for an example audio file. It shows the original waveform, the resynthesized sinusoidal trajectories and the difference of the two signals, the residual waveform.

Stochastic Approximation

The residual part can be saved as a time signal without further modification. To gain flexibility and to reduce memory requirements, however, it can be modeled using a source-filter model, i. e. filtered white noise. This can be done in both time and frequency domain. In general, the frequency domain approach is much less computationally demanding than filtering in the time domain.

The residual component can also be saved as a single STFT per frame. As the frequency resolution of the residual part is less important, the size of the STFT can be much smaller than the one used for spectral estimation.

1.3 Synthesis

The synthesis of the D+S model is, like its analysis, a two-part process. The deterministic and stochastic components are synthesized separately and then summed. For a given frequency track and in between two frames a cubic function is used to unwrap and interpolate the phase such that the phase track is maximally smooth.

The derivation of this interpolating cubic function can be found in [MQ86]. The interpolated phase function is fed to a sine-wave generator, which is amplitude modulated and added to the other sine waves to give the deterministic output. The stochastic component is obtained by filtering white noise with the recorded filter parameters, or, if a STFT was used, by calculating the inverse STFT for each frame [SS90].

Some implementations realize both sinusoidal and residual synthesis in the frequency domain, add the two parts still in the frequency domain and then perform only a single inverse STFT per frame to save processing power [RD92].

1.4 Drawbacks

There are two drawbacks in the just technique described above, which are addressed in this research work.

Firstly, the frames are supposed to have stationary spectral characteristics for their whole duration, but, in contradiction to this, the model aims to analyze time-varying parameters. The methods presented later in this work, assume that the frequencies of the partial tones change linearly within a frame. The amplitudes are still handled as being constant during a frame, though.

Secondly, in the peak continuation process only the frequency difference is taken into account. This can lead to a wrong continuation of trajectories if frequencies are very close, changing rapidly or if they are crossing. In chapter 8 two improved algorithms using the *chirp rate* are presented.

Chapter 2

Frequency Representation of Time Signals

In many cases it is most natural to represent an audio signal as a function over time (with values proportional to air pressure or voltage or ...). It can, however, also be very interesting to have a representation over frequency, i. e. to know how the energy of the signal is spread over the different frequencies.

The mathematics of the frequency representation was conceptualized by Joseph Fourier in early 19th century in the context of thermodynamics [Fou22]. He found out that every signal can be represented by a superposition of an infinite number of sine waves. The characteristics of a signal are modeled by constructive and destructive interference between the sine waves.

The conversion from the *time domain* into the *frequency domain* is done by the so-called Fourier transform.

2.1 Fourier Transform

The Fourier transform is defined by

$$X(f) = \underset{t \to f}{\mathcal{F}}\{x(t)\} = \int x(t)\, \mathrm{e}^{-j2\pi ft}\, dt. \tag{2.1}$$

Its counterpart, the inverse Fourier transform, is given by

$$x(t) = \underset{t \leftarrow f}{\mathcal{F}^{-1}}\{X(f)\} = \frac{1}{2\pi} \int X(f)\, \mathrm{e}^{j2\pi ft}\, df. \tag{2.2}$$

As one can see in equation (2.1), the computation of the frequency content $X(f_0)$ of a signal at a single frequency value f_0 requires a complete knowledge of the past and future of the signal $x(t)$. On the other hand, as seen in equation (2.2), the computation of an instantaneous signal value $x(t_0)$ requires the knowledge of an infinite number of frequency components $X(f)$.

Any type of signal, except an infinitely long sinusoid with constant frequency and amplitude is modeled by a superposition of possibly infinitely many sinusoids. Even a single sinusoid modulated in frequency and/or amplitude is represented by an infinite number of unmodulated sinusoids.

The Fourier transform is a very powerful mathematical tool and has revolutionized mathematics and other fields. It is also heavily used in signal processing, especially since the introduction of the Fast Fourier Transform (FFT) in 1965 [CT65].

For the analysis of an audio signal, however, it has some severe drawbacks. Even though the signal can be perfectly reconstructed from the frequency representation $X(f)$, we cannot simply get any temporal information from it. We can see which frequencies are present in the signal and to which extent, but we cannot easily see which frequency is there at which time. Neither can we distinguish if one frequency is present in the signal for a long time with small amplitude or if it is there for a short time with high amplitude.

2.2 Short Time Fourier Transform

To get rid of the shortcomings mentioned before up to a certain extent, the Short Time Fourier Transform (STFT) was introduced. It is basically the same idea of the Fourier transform, but only applied to a local time interval. This is done by windowing the input signal $x(t)$ with a window $w(t)$ centered at the time of interest and then applying the Fourier transform to the windowed signal. The STFT is defined as [Coh95]:

$$\mathsf{STFT}_x(t, f; w) = \int x(\tau) w(t - \tau)\, \mathrm{e}^{-j2\pi f \tau}\, d\tau, \tag{2.3}$$

where the length on which the window $w(t)$ is non-zero is T. The type of window function used is a critical factor that determines the overall accuracy of the STFT. A good discussion on the use of window functions and considerations of their appropriateness is presented in [Har78].

The STFT does not completely solve the problems of locating events in time. Within a given window there is still no time information. The smaller the window, the better can an event be localized in time, but it is impossible to achieve arbitrarily fine resolution in both time and frequency simultaneously. This limitation is referred to as the *uncertainty principle* [Coh95].

Using the STFT for spectral analysis

The STFT has to be performed on a time segment of length $T > 0$. This is to say that any STFT analysis will form a spectral representation from this limited time segment, but cannot explicitly determine the instantaneous frequency of the signal. When the spectral representation of the STFT is analyzed to determine the

instantaneous signal characteristics at a particular frequency, there is a standard assumption that the signal is stable in frequency and amplitude for the entire duration of the analysis window. When the STFT is used to determine instantaneous signal parameters of a signal which is not completely stable for the duration of the window, there will be distortion of the physical reality of the signal in the spectral representation. This distortion will manifest itself in three characteristics that can be observerd from the spectrum, namely in frequency, amplitude and phase. Although the frequency, amplitude and phase extracted in this case can be used to correctly re-synthesize the signal, they cannot be individually interpreted or modified in a meaningful way.

When transforming a linear chirp, components of all frequencies which were passed during the frame (and even more) will be present. Using a smooth window it will be possible to detect the frequency at the center of the window, which represents the instantaneous frequency at this time. The amplitude of the chirp, however, cannot be recovered because it is distributed over all frequency components. The amount by which the amplitude is scaled is proportional to the chirp rate of the signal [MB95]. It is also shown in [MB95] that the STFT representation will suffer from a shift in the phase angle at the frequency of the peak as a result of the signal's non-stationary nature. Again, the amount by which the phase is shifted is proportional to the chirp rate of the partial.

When doing the analysis for additive synthesis, we expect the parameters to change continuously over time but we extract the values of the parameters only at the frame rate. When using the STFT for the analysis, however, we assume that the same parameters are not changing at all during a frame.

Chapter 3

About Instantaneous Frequency and Amplitude

In this chapter we give a short overview about the concept of instantaneous frequency and amplitude and how to extract them from an audio signal. The interested reader can find much more information and deeper insight into this topic in [Pic97, Boa92a, Boa92b].

The simplest time-varying signal is the sinusoid. It is a solution to many differential equations and it is common in nature. It is characterized by a constant amplitude A and a constant frequency f_0:

$$x(t) = A\cos(2\pi f_0 t).$$

One could now try to generalize the simplicity of the sinusoid by hoping that a general signal can be written in the form

$$x(t) = A(t)\cos(\theta(t)),$$

and may be tempted to believe that $A(t)$ would be the instantaneous amplitude and $\theta(t)$ would be the instantaneous phase of the signal $x(t)$. Furthermore, the instantaneous frequency of the signal $x(t)$ would be

$$f(t) = \frac{1}{2\pi}\frac{d\theta(t)}{dt}.$$

Unfortunately, this is not the case. The problem is, that for a given signal $x(t)$ there can be found infinitely many pairs of $A(t)$ and $\theta(t)$ which generate the same signal.

3.1 Analytic Associate

This ambiguity can be avoided if we use a complex signal of the form

$$z(t) = a(t)\,\mathrm{e}^{j\phi(t)} = x(t) + jy(t),$$

where $a(t)$, $\phi(t)$, $x(t)$ and $y(t)$ are real-valued and $a(t) \geq 0$.

From this signal the instantaneous amplitude $a(t)$ and the instantaneous phase $\phi(t)$ can be unambiguously extracted by determining the magnitude and phase of its complex values:

$$a_x(t) \equiv |z_x(t)|,$$
$$f_x(t) \equiv \frac{1}{2\pi}\frac{d}{dt}\arg z_x(t).$$

The signal $z(t)$ is called the *analytic associate* of $x(t)$. The yet unknown signal $y(t)$ has to be created in a way that $z(t)$ has the properties of an *analytic signal*. Those properties and how to obtain $y(t)$ is shown in the rest of this chapter. In-depth coverage of this topic including all necessary proofs can be found in [Coh95, Pic97].

It turns out that the signal $y(t)$ must have a phase lag of 90° to the input signal $x(t)$. This is called to be in *quadrature* to the input signal.

3.2 Hilbert Transform

The unknown signal $y(t)$ (the imaginary part of the desired analytic signal) can be obtained by means of the so called Hilbert transform $\mathcal{H}\{x(t)\}$ and with it the analytic associate of $x(t)$ can be written as

$$z_x(t) = x(t) + j\mathcal{H}\{x(t)\},$$

with

$$\mathcal{H}\{x(t)\} = \underset{t\leftarrow f}{\mathcal{F}^{-1}}\left\{-j\,\mathrm{sgn}(f)\,\underset{t\rightarrow f}{\mathcal{F}}\{x(t)\}\right\}. \tag{3.2}$$

The 90° phase shift can be seen nicely in the following examples:

$$\mathcal{H}\{\cos(2\pi f_0 t)\} = \sin(2\pi f_0 t) \text{ and}$$
$$\mathcal{H}\{\sin(2\pi f_0 t)\} = -\cos(2\pi f_0 t).$$

The transfer function of the Hilbert transform is $-j\,\mathrm{sgn}(f)$ as seen in equation (3.2), its impulse response is $\mathcal{F}^{-1}\{-j\,\mathrm{sgn}(f)\} = \frac{1}{\pi t}$. The Hilbert transform can therefore also be defined in the time domain:

$$\mathcal{H}\{x(t)\} = (x * u)(t), \quad \text{with } u = \frac{1}{\pi t}. \tag{3.3}$$

For the calculation of this convolution we have to evaluate the Cauchy principal value of an improper integral. Because of that, the frequency domain approach is mathematically simpler than the time domain approach. In a software implementation both approaches are possible but the frequency domain approach is normally easier to implement and computationally faster. Details for the time domain implementation can be found in [RFB94] and for the frequency domain implementation in [Mar99].

3.3 Properties of the Analytic Signal

The following points are explained in detail including proofs in [Coh95]:

- The analytic signal has a one-sided Fourier transform, i. e. there are no negative frequencies.
- The energy of the analytic signal is twice the energy of the original signal.
- The energy of the real part is equal to the energy of the imaginary part.
- The analytic procedure puts the low frequency content in the amplitude and the high frequency content in the term $e^{j\phi(t)}$.
- The analytic associate of an already analytic signal is the signal times two.
- The convolution of an analytic signal with an arbitrary function results in an analytic signal.
- The product of an analytic signal $a(t)\, e^{j\phi(t)}$ with an arbitrary signal $s(t)$ results only in an analytic singnal if the highest frequency in $s(t)$ is lower then (or equal to) the highest frequency in $a(t)$.

We will be using the analytic signal as input to the Ambiguity Function (AF) described in section 4.3. This has two benefits:

1. The AF needs normally an input signal which is oversampled at twice the sampling rate to avoid aliasing. When using the analytic signal, this oversampling can be omitted because the analytic associate uses only half the bandwidth of the original real signal.
2. Using the analytic signal avoids cross components between positive and negative frequencies that would arise when using a real valued input signal.

The analytic associate is also used in section 6.5 to estimate the initial phase, because the initial phase information is lost when calculating the AF.

Chapter 4
Time-Frequency Distributions

In chapter 2 we mentioned that it is strictly speaking not possible to extract the instantaneous frequency (see chapter 3) from the Short Time Fourier Transform (STFT). We cannot get information for a given time instant from the STFT because its frequency and amplitude information is a mean value over the duration of the window. We also showed in chapter 2 that the amplitude is altered depending on the chirp rate of the signal.

We now present a class of distributions which can solve the aforementioned problems: the bilinear Time-Frequency Distributions (TFDs). Although they can help us to estimate instantaneous frequency and amplitude, they have some drawbacks themselves.

Bilinear structures do not adhere to the principle of linear superposition, i. e. when we analyze a sum of two (or more) signals, we get not the same result as when we analyze the two (or more) components separately and then add the results. Actually, we get the sum of the results plus additional components, so-called *interference terms* or *cross terms*.

Bilinear systems can generally not be used to determine the initial phase of a signal. This is because the autocorrelation functions used in their construction effectively eliminate the phase information of the signal. Therefore, we use the analytic signal (see chapter 3) for the estimation of the initial phase.

4.1 Instantaneous Autocorrelation Function

Basis of all bilinear time-frequency distributions presented here is the Instantaneous Autocorrelation Function (IAF). It is defined by

$$\mathcal{K}_x(t,\tau) = x\Big(t+\frac{\tau}{2}\Big)\, x^*\Big(t-\frac{\tau}{2}\Big). \tag{4.1}$$

The IAF of $x(t)$ is the product of the signal $x(t)$ with its complex conjugate. The two functions are time shifted with a relative lag of τ.

4.2 Wigner-Ville Distribution

Eugene Wigner first introduced this distribution in 1932 in the context of quantum statistical mechanics [WIG32]; in 1948, Jean Ville applied it to signal processing [VIL48]. The Wigner Distribution (WD) and the Wigner-Ville Distribution (WVD) are formally the same. The difference is only the field in which it is used and that the WVD is normally used with the analytic associate (see section 3.1) of the signal under analysis. The WVD is defined as

$$\mathcal{W}_z(t,f) = \int z\Big(t+\frac{\tau}{2}\Big)\, z^*\Big(t-\frac{\tau}{2}\Big)\, \mathrm{e}^{-j2\pi f\tau}\, d\tau. \tag{4.2}$$

More compactly, we can also write

$$\mathcal{W}_z(t,f) = \underset{\tau\to f}{\mathcal{F}}\{\mathcal{K}_z(t,\tau)\}.$$

As shown in [COH95], the WVD is always real-valued:

$$\begin{aligned}
\mathcal{W}_x^*(t,f) &= \int_{-\infty}^{\infty} x^*\Big(t+\frac{\tau}{2}\Big)\, x\Big(t-\frac{\tau}{2}\Big)\, \mathrm{e}^{j2\pi f\tau}\, d\tau && \text{(4.3a)}\\
&= -\int_{\infty}^{-\infty} x^*\Big(t-\frac{\tau}{2}\Big)\, x\Big(t+\frac{\tau}{2}\Big)\, \mathrm{e}^{-j2\pi f\tau}\, d\tau && \text{(4.3b)}\\
&= \int_{-\infty}^{\infty} x^*\Big(t-\frac{\tau}{2}\Big)\, x\Big(t+\frac{\tau}{2}\Big)\, \mathrm{e}^{-j2\pi f\tau}\, d\tau && \text{(4.3c)}\\
&= \mathcal{W}_x(t,f)
\end{aligned}$$

Equation (4.3a) shows the complex conjugate of the WVD defined in equation (4.2). From line (4.3a) to line (4.3b) τ was substituted by $-\tau$. Hence $d\tau$ became $-d\tau$ which is the reason for the minus sign in front of the integral. Also, if τ is integrated from $-\infty$ to ∞, as a result $-\tau$ is integrated from ∞ to $-\infty$. From line (4.3b) to line (4.3c) the direction of the integration is reversed and therefore the minus in the beginning is removed.

Due to the quadratic nature of the WVD, the WVD of the sum of two signals $x(t)$ and $y(t)$ is not the sum of their respective WVDs, but rather

$$\mathcal{W}_{x+y} = \mathcal{W}_x + \mathcal{W}_y + \mathcal{W}_{xy} + \mathcal{W}_{yx}. \tag{4.4}$$

The terms with two different subscript indices are called *cross terms* and the ones with a single index are called *auto terms* because they involve a cross correlation and an auto correlation, respectively. The cross WVD is defined as

$$\mathcal{W}_{xy}(t,f) = \int x\Big(t+\frac{\tau}{2}\Big)\, y^*\Big(t-\frac{\tau}{2}\Big)\, \mathrm{e}^{-j2\pi f\tau}\, d\tau. \tag{4.5}$$

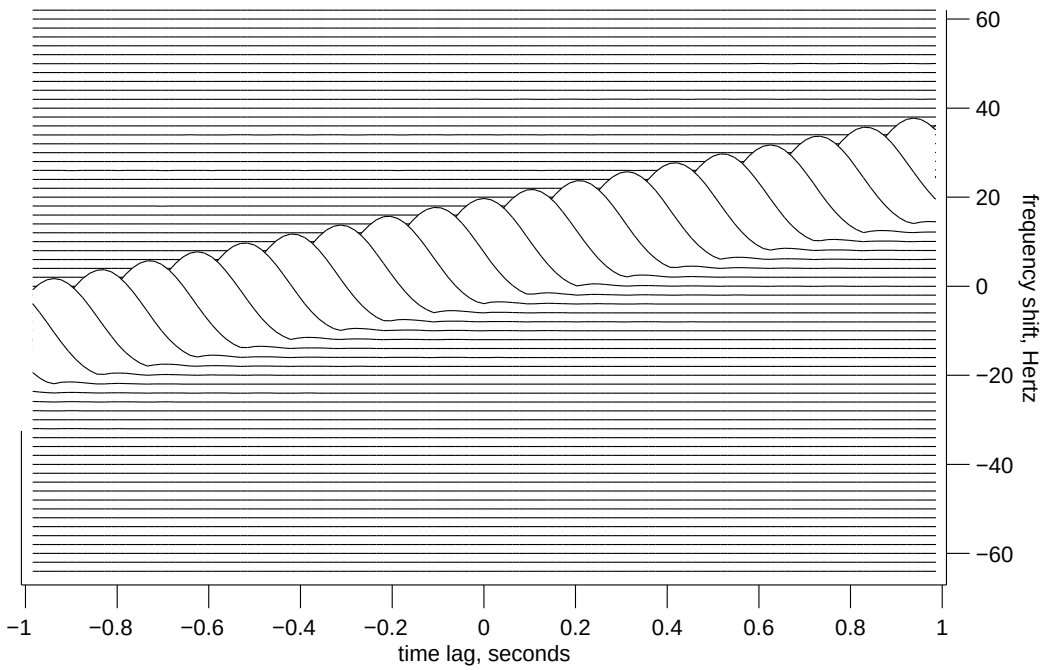

Figure 4.1: *Magnitude of the Windowed Ambiguity Function of a linear chirp with a chirp rate of* 19 *Hz/s. The length of the Hanning window used for the AF is* 64 *samples, the sampling frequency is* 128 *Hz. The absolute frequency cannot be seen in this representation because this information is only contained in the phase of the AF.*

The cross WVD is complex-valued. However, since $\mathcal{W}_{xy} = \mathcal{W}_{yx}^*$ and therefore $\mathcal{W}_{xy} + \mathcal{W}_{yx}$ is real, equation (4.4) on the preceding page can be written as

$$\mathcal{W}_{x+y} = \mathcal{W}_x + \mathcal{W}_y + 2\,\mathcal{R}e\{\mathcal{W}_{xy}\}\,.$$

For a signal $s(t)$ with N partials this becomes [Coh95]:

$$s(t) = \sum_{n=1}^{N} a_n s_n(t)$$

$$\mathcal{W}_s = \sum_{n=1}^{N} |a_n|^2 \mathcal{W}_{s_n} + 2 \sum_{n=1}^{N-1} \sum_{k=n+1}^{N} \mathcal{R}e\{a_n a_k^* \mathcal{W}_{s_n s_k}\}\,.$$

Here we see that the WVD of a signal with N partials will contain N auto terms and $N(N-1)/2$ cross terms.

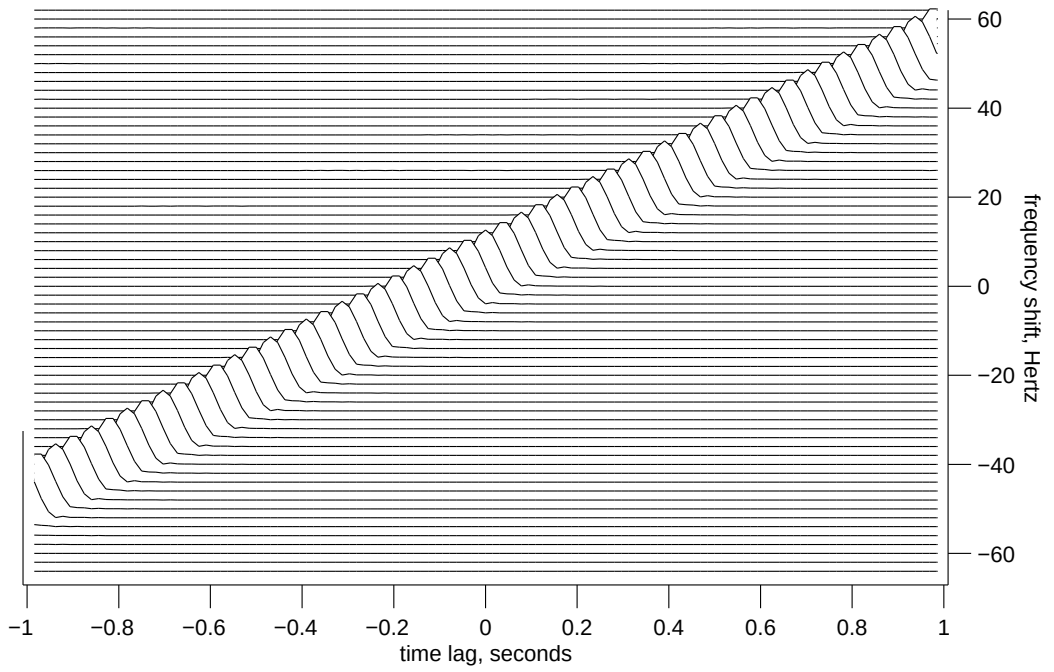

Figure 4.2: *Magnitude of the Windowed Ambiguity Function of a linear chirp with a chirp rate of* 51 *Hz/s.*

4.3 Ambiguity Function

The Ambiguity Function (AF) is closely related to the WVD. However, its output domain is not the (t, f)-domain but the *Doppler-lag* (ν, τ)-domain, also known as the *ambiguity plane*. This is because the IAF $\mathcal{K}_z(t, \tau)$ is not integrated with respect to the lag τ but with respect to the time t. Therefore the AF is defined as

$$\mathcal{A}_z(\nu, \tau) = \int z\Big(t + \frac{\tau}{2}\Big)\, z^*\Big(t - \frac{\tau}{2}\Big)\, \mathrm{e}^{-j2\pi\nu t}\, dt, \tag{4.6}$$

or, more compactly

$$\mathcal{A}_x(\nu, \tau) = \underset{t \to \nu}{\mathcal{F}} \left\{ \mathcal{K}_z(t, \tau) \right\}. \tag{4.7}$$

Figures 4.1 and 4.2 show the AFs of two linear chirps. The AF suffers from the same non-linear superposition behaviour as the WVD and all other bilinear TFDs. The AF of a sum of two components is

$$\mathcal{A}_{x+y} = \mathcal{A}_x + \mathcal{A}_y + \mathcal{A}_{xy} + \mathcal{A}_{yx}, \tag{4.8}$$

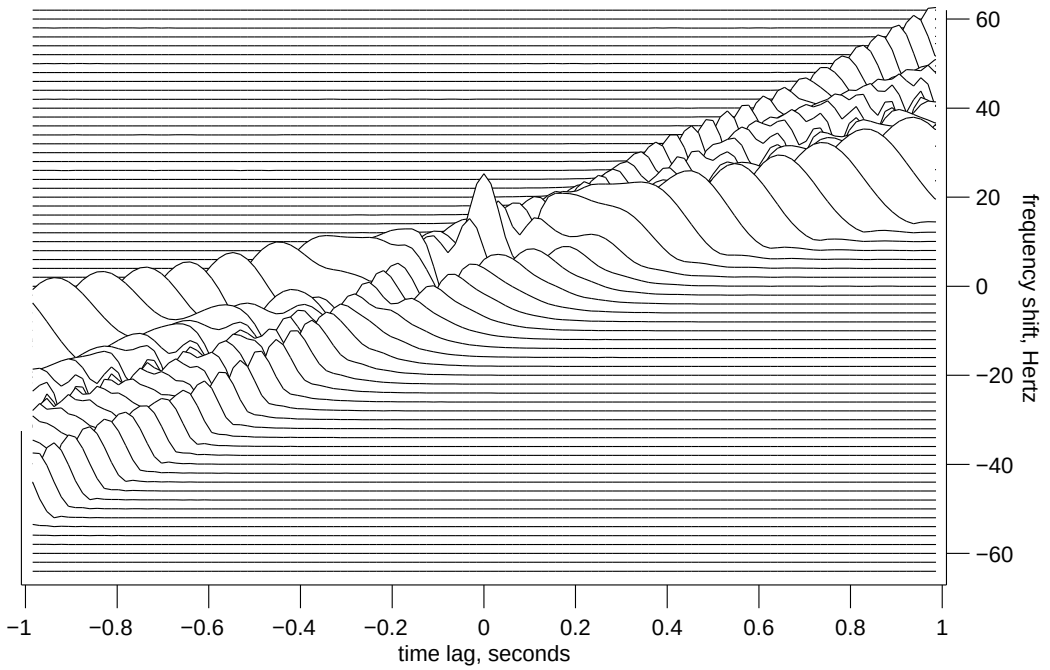

Figure 4.3: *Magnitude of the Windowed Ambiguity Function of the sum of two linear chirps with the chirp rates* 19 *Hz/s and* 51 *Hz/s, respectively. The amplitude of the second one is* 20 *% lower compared to the first one.*

with $\mathcal{A}_{xy}$ being the cross AF

$$\mathcal{A}_{xy}(\nu,\tau) = \int x\Big(t+\frac{\tau}{2}\Big)\, y^*\Big(t-\frac{\tau}{2}\Big)\, \mathrm{e}^{-j2\pi\nu t}\, dt.$$

Figure 4.3 shows the AF of the sum of the two chirps from figures 4.1 and 4.2. The interference terms can be clearly seen between the two ridges. For a signal $s(t)$ with N partials equation (4.8) on the preceding page becomes:

$$s(t) = \sum_{n=1}^{N} a_n s_n(t)$$
$$\mathcal{A}_s = \sum_{n=1}^{N}\sum_{k=1}^{N} a_n a_k \mathcal{A}_{s_n s_k}.$$

As with the WVD, terms with two times the same index are called *auto terms* and terms with differing indices are called *cross terms*.

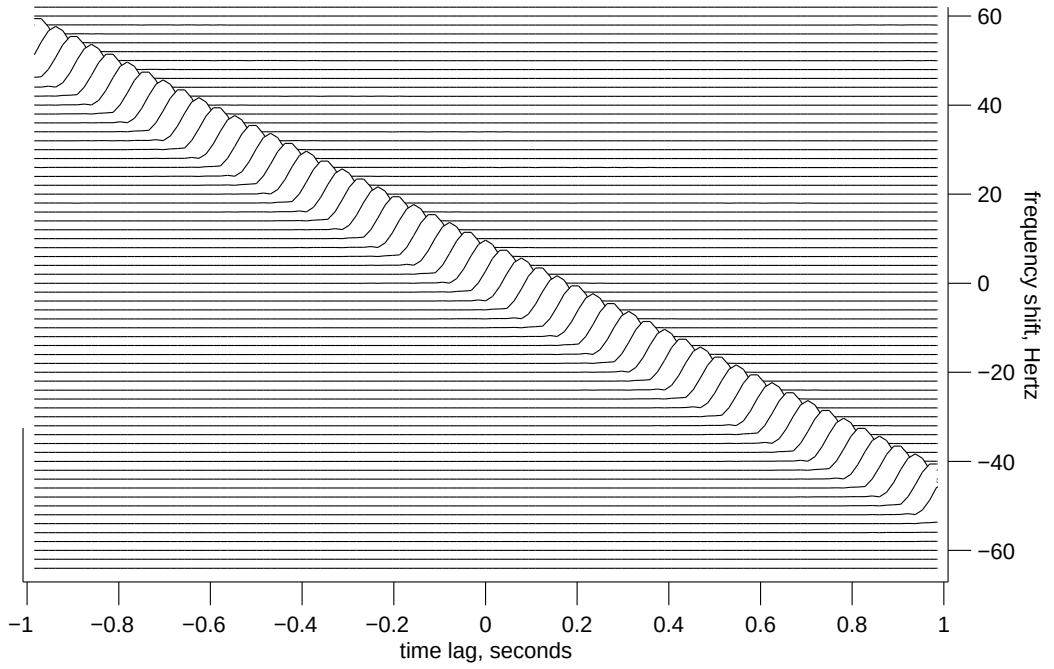

Figure 4.4: *Magnitude of the Windowed Ambiguity Function of a linear chirp with a chirp rate of* $-51\,Hz/s$.

Figure 4.5 on the facing page shows the AF of the sum of chirp signals whose AFs are depicted in figures 4.1, 4.2 and 4.4. The AF has the advantage that it maps the auto terms to the center (origin) of the ambiguity plane and cross terms away from the center. The AF shows a symmetry around the origin described by $\mathcal{A}(\nu,\tau) = \mathcal{A}^*(-\nu,-\tau)$ [Coh95].

We can get a somehow different point of view if we rewrite the definition of the AF in equation (4.6) on page 30 like this [Boa03]:

$$\mathcal{A}_z(\nu,\tau) = \int z\Big(t+\frac{\tau}{2}\Big)\left[z\Big(t-\frac{\tau}{2}\Big)\,\mathrm{e}^{j2\pi\nu t}\right]^* dt.$$

The expression in square brackets can be obtained by delaying $z\big(t+\frac{\tau}{2}\big)$ in time by τ and shifting it in frequency by ν, indicating that $\mathcal{A}_z(\nu,\tau)$ is the correlation of the signal with a time-delayed and frequency-shifted version of itself.

This correlation is well known in radar theory as the Sussman ambiguity function; the name "ambiguity" arises from the equivalence between time-shifting and frequency-shifting for linear FM signals, which are frequently used in radar. Hence the Doppler-lag (ν,τ) domain is also called the *ambiguity domain*.

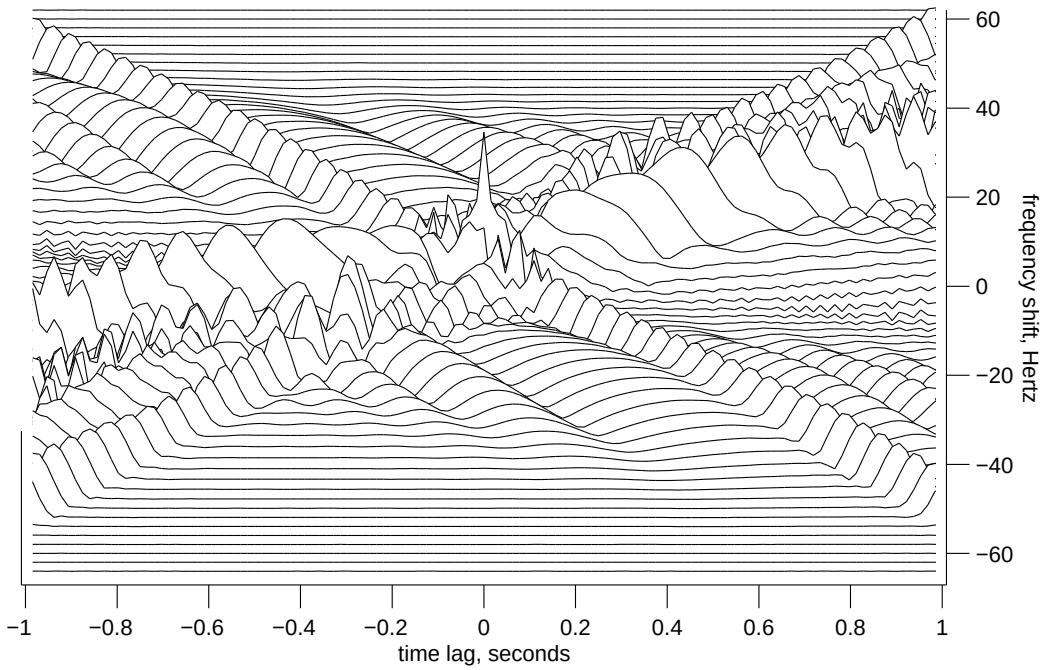

Figure 4.5: *Magnitude of the Windowed Ambiguity Function of the sum of three linear chirps with the chirp rates* 19 *Hz/s,* 51 *Hz/s and* −51 *Hz/s, respectively. Compared to the first one, the amplitude of the second and third one is* 20 *% and* 30 *% lower, respectively.*

4.4 Cohen's Class Distributions

All TFDs discussed here are actually members of the same class of distribution often called *Cohen's class* (although by Cohen himself it is called *general class* [Coh95]). This class characterizes TFDs by an auxiliary function, the *kernel function*. The properties of a distribution are reflected by simple constraints on the kernel. By examining the kernel one can readily ascertain the properties of the distribution. For a list of kernels, their properties and properties of their respective TFDs see [Coh95, Boa03]. There cannot be an exhaustive list, because infinitely many kernels can be generated. The gerneral class, from which *all* TFDs can be obtained, is defined in [Boa03] and [Fla99] as

$$\rho_x(t, f; \gamma) = \iint \mathcal{W}_x(s, \xi)\gamma(s - t, \xi - f)\, ds\, d\xi, \tag{4.9}$$

where $\gamma(t, f)$ is the two dimensional *kernel*.

Chapter 5

The Experimental Setup

In this chapter we show the setup used for applying the concepts discussed in previous chapters – namely the analytic signal and the Ambiguity Function (AF) – to additive analysis. This setup is implemented in MATLAB® using mainly standard commands and a few functions from the Signal Processing Toolbox.

Initially, the free software program SMSTools[1] was used as a part of the setup, but the relevant functions have since been reimplemented in MATLAB® as explained in appendix C.

Figure 5.1 on the following page shows the functional units of the used setup. In the next few paragraphs we will briefly describe each component and how they are linked up with each other. Chapters 6 and 8 describe in more detail the key features *parameter estimation* and *peak continuation*.

Ambiguity Function

The AF is the centerpiece of the whole process, so we will start our description with it. If its input is a linear chirp, we expect to find a linear ridge in the magnitude of its output, passing through the origin. From the slope of this ridge we can estimate the chirp rate and it is also along this ridge where we will get our frequency and amplitude estimators. As we cannot use the AF for estimating the initial phase, we will use the analytic signal to do that. Chapter 6 explains the parameter estimation both based on the AF and the analytic signal.

Frame Segmentation

As mentioned earlier, the estimators based on the AF expect the signal to be a linear chirp. In general, when analyzing music or speech signals, this will not be the case. However, if we take parts of the signal with a sufficiently short duration, we can assume that the frequency progression will be almost linear during this short time.

[1] `http://clam.iua.upf.edu/`

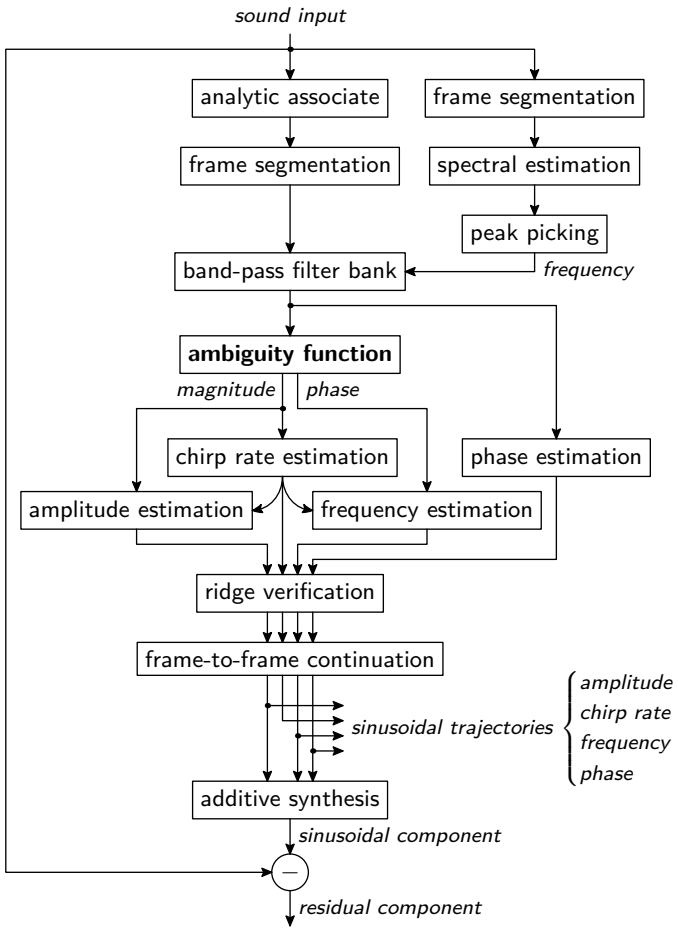

Figure 5.1: *The Prototype Application*

Band-Pass Filter Bank

Another prerequisite for using the AF as an estimator is that its input is a mono-component signal. To fulfill this requirement, we split a general audio frame into a set of single partial tones by filtering the frame with a band-pass filter which has as center frequency the frequency of the partial tone we want to isolate. More details on the filter can be found in appendix B.2. Once we have generated a set of partial tones, we run the estimation process on each of them separately. To be able to apply the appropriate band-pass filters, we have first to find a set of frequencies where we expect the partial tones to be.

Spectral Estimation

To obtain the frequencies of likely partial tones that can be used to control the band-pass filters, we first of all split the signal into frames at the same frame rate as in the main branch. One of the reasons we use a distinct frame separation is

to be able to use a different frame size and/or windowing for each branch. The other reason is that in the main branch the analytic associate is calculated before the frame segmentation process.

For each frame then the Short Time Fourier Transform (STFT) is computed. This is in fact done by windowing each frame with a smooth window and then calculating the Fast Fourier Transform (FFT) of the windowed frame.

From the magnitude of the STFT all local maxima are determined. A certain number of the highest maxima is then used as center frequencies in the band-pass filters. The maximum number of selected frequencies can be set according to the expected number of partial tones in the signal under analysis.

Analytic Associate

The benefits of using the analytic associate of the signal rather than the signal itself are threefold:

- First, it gives us a possibility to estimate the initial phase, which cannot be obtained from the AF.
- Furthermore, within the AF it avoids cross-components that would arise between the positive and negative frequencies of a real signal.
- The third benefit is that the analytic signal needs only half the bandwidth of the original real signal. Hence we can use it, without need for any oversampling, as input to the AF. A real signal would have to be oversampled at twice the original sampling rate to avoid aliasing.

As depicted in figure 5.1 on the preceding page, we are calculating the analytic associate of the whole input signal before the frame segmentation. As all filtering processes, the generation of the analytic signal has side effects at the beginning and the end of the audio data. If we would obtain the analytic associate for each frame, we would have to deal with these effects at the beginning and end of each frame, but when doing the calculation before the frame segmentation we encounter them only in the very first and very last frame. Additionally, as we normally use overlapping frames, the frame-wise calculation of the analytic associate would result in much more samples to be filtered in total.

We must bear in mind, however, that the multiplication of an analytic signal with an arbitrary function does not generally result in an analytic signal. In the windowing process incorporated in the calculation of the (windowed) AF we therefore have to apply certain restrictions to the windowing function. The windowed signal is only an analytic signal if the highest frequency contained in the window function is lower than the lowest frequency in the analytic signal to be windowed [Coh95].

Ridge Verification

The peak picking done at the beginning selects the most prominent peaks from the spectrum but it can of course not tell if there are really partial tones at the peak frequencies or if the peaks are caused by transient events or by noise. At the ridge verification step we try to find out how close the output of the AF is to our ideal template, the linear chirp. This is done by calculating the estimators for the frequency using all phase values along the detected ridge. Then we calculate the standard deviation of all the estimated frequencies. If the standard deviation is below a given threshold, we suppose that we are dealing with a partial tone, but if not, the calculated estimators will not give a reliable result, because they only work with linear chirps. If the ridge verification indicates that the current filtered frame does not contain a linear chirp, or at least something similar, the estimators are ignored and not added to any trajectory.

Peak Continuation

All sets of estimators for a frame which passed the ridge verification are now compared to the estimators of the previous and the next frame. We try to find out which of these partial tones belong to the same trajectory. How this works in detail is shown in chapter 8. It is also explained there how a new trajectory can be "born" and, if its time has come, how a trajectory "dies".

After the peak continuation we have already the sinusoidal representation of the signal. The estimators for all frames together with the information to which trajectory they belong can now be saved to be used for further processing like pitch shifting or time stretching or many others.

Calculation of the Residual Part

The sinusoidal representation can now be used to obtain the residual signal. The residual signal also shows if partials are detected correctly. If all tonal components are detected, the residual signal should contain only transients and noise.

To calculate the residual part, we first resynthesize the sinusoidal tracks and then subtract the sinusoidal component from the original signal. It is very important that we use the correct initial phase information because only then the sinusoidal component can be removed by destructive interference.

For the resynthesis of the sinusoidal trajectories we use a matlab function written by Dan Ellis, available from his homepage[2] at Columbia University in New York.

[2] http://www.ee.columbia.edu/~dpwe/resources/matlab/sinemodel/

Chapter 6

Improving the Estimation

This chapter shows how to estimate chirp rate, frequency, amplitude, and initial phase from the Ambiguity Function (AF) and the analytic signal respectively.

A requirement for the following estimators is, that the signal we analyze is a mono-component signal, i. e. only one isolated partial tone. We meet this requirement by band-pass filtering the input frames based on the frequency information obtained from a raw estimation based on the Short Time Fourier Transform (STFT). See appendix B.2 for implementational details about the band-pass filter.

6.1 Windowed Ambiguity Function

We cannot use the AF directly as defined in equation (4.6) on page 30 in a practical implementation, because it includes an integral from $-\infty$ to ∞ which means that we would also need an infinitely long input signal. To be able to deal with finite input signals we could simply restrict the integration interval to a certain time T, i. e. integrate only from $-T/2$ to $T/2$. This requires the input signal $z(t)$ to be defined in a range $-(T/2 + \tau/2) < t < (T/2 + \tau/2)$. The calculation of the AF of a linear chirp under this time restriction can be found in appendix A.2.

Setting limits to the time axis essentially imposes a rectangular window to the Instantaneous Autocorrelation Function (IAF) before calculating the Fourier transform. Often, however, it is desirable to use window functions other than the rectangular window. This is particularly the case when one may wish to suppress cross-terms that may arise as a result of partials near the chronological end of the windowed signal. When using an arbitrary window $w(t)$, we get the *windowed Ambiguity Function*

$$\begin{aligned} \mathcal{A}_z(\nu, \tau; w) &= \underset{t \to \nu}{\mathcal{F}} \{ w(t) \mathcal{K}_z(t, \tau) \}, \text{ or more verbosely,} \\ \mathcal{A}_z(\nu, \tau; w) &= \int w(t) z\Big(t + \frac{\tau}{2}\Big)\, z^*\Big(t - \frac{\tau}{2}\Big)\, \mathrm{e}^{-j2\pi\nu t}\, dt. \end{aligned} \tag{6.1}$$

The effect of this window must be taken into account when estimating the amplitude later in this chapter.

6.2 Chirp Rate Estimation

If the input to the AF is a linear chirp, we can expect a linear ridge in the magnitude of the output. To find this ridge, and especially its slope, we search for the absolute maxima of the AF along the Doppler (ν) axis.

As the frequency axis is discretized in steps of f_s/N Hertz (where f_s is the sampling frequency and N is the size of the FFT), we use parabolic interpolation to allow for a finer frequency resolution.

The maxima theoretically lie on a straight line passing through the origin and only one time-frequency point would suffice to exactly determine the ridge, but practically this will not be exactly the case. Therefore we use all maxima and try to find a straight line which comes closest to them. A meaningful way to do that is to find a line from which the mean squared difference to the points is minimal. Such a line fitting procedure is called *least square linear regression*. Implementational details to the ridge detection process can be found in appendix B.4.

As we expect that the line passes through the origin, the only output of the regression process is the slope of the ridge.

6.3 Frequency and Amplitude Estimation

As shown in appendix A.3, the windowed AF of a linear chirp is

$$\mathcal{A}_z(\nu, \tau; w) = a^2 W(\nu - \alpha\tau)\, \mathrm{e}^{j2\pi f_0 \tau}, \tag{6.2}$$

where $W(f)$ is the Fourier transform of the window $w(t)$, f_0 is the frequency at time $t = 0$ (which represents the temporal center of the frame) and α is the chirp rate (in Hz/s).

The frequency information is only contained in the complex exponential, therefore we have to calculate the phase angle $\Phi(\nu, \tau)$ of the AF and extract the frequency from it. Theoretically, we could just calculate the frequency at arbitrary values of ν and τ by using:

$$f_0 = \frac{\Phi(\nu, \tau)}{2\pi\tau}, \text{ where } \Phi(\nu, \tau) = \arg(\mathcal{A}_z(\nu, \tau; w)). \tag{6.3}$$

Although the values of (ν, τ) where the estimator is computed should not matter, the estimation works best at relatively large magnitude values of the AF. At small magnitudes phase values can be altered due to numerical errors. This is the reason

why we estimate frequencies near the detected ridge (where $\alpha\tau = \nu$). Also, in theory one reading of the phase value would be sufficient, but we obtain the values along the whole ridge and take the mean average of all of them to make the estimator consistent.

It also turned out that a more stable estimator can be found by multiplying equation (6.3) on the preceding page by τ and then differentiate both sides getting

$$f_0 = \frac{1}{2\pi}\frac{\partial\Phi(\nu,\tau)}{\partial\tau}. \tag{6.4}$$

Our final frequency estimator is the average value

$$\overline{f_0} = \frac{1}{2\pi}\overline{\frac{\partial\Phi(\nu,\tau)}{\partial\tau}} \quad \forall(\nu,\tau) \text{ where } \alpha\tau = \nu.$$

Based on equation (6.2) on the facing page an estimator for the amplitude can be found by

$$a = \sqrt{\frac{\overline{\left|\mathcal{A}_z(\nu,\tau;w)\right|}}{\left|W(0)\right|}}, \text{ wherever } \alpha\tau = \nu.$$

Here we do not use a single point of (ν,τ) either, but estimate the amplitude as the average value of all amplitude values along the detected ridge. We exclude the amplitude at the origin when estimating the amplitude, because this is the point where all components contribute to, including eventual noise components in the signal which we do not want to affect the amplitude estimator of the partial tone.

The frequency and amplitude could of course also be estimated from the analytic signal by the derivation of the phase (divided by 2π) and the magnitude, respectively. The estimators based on the AF, however, are more stable and more exact, particularly when the signal is contaminated by strong noise.

6.4 Ridge Verification

The spectral estimation based on the STFT provides a set of partial candidates per frame. It does not, however, give any information if a partial candidate really represents a partial tone of the input sound or if it was detected because of noise or another error. For the presented estimation process to work correctly we must be sure that the input meets our given restrictions. Only if it is a mono-component linear chirp, we can successfully use our estimators.

If a filtered frame does not have the necessary properties, the estimators will be wrong and they will distort the resynthesized sound. Therefore it is indispensable for us to verify if a given set of estimators is valid or not.

In case of a perfectly linear chirp the frequency estimator from equation (6.4) on the previous page has the same value for any (ν, τ) along the ridge. Of course in practice these values will vary. When the variation is comparatively small, we can assume that we are dealing with a signal sufficiently close to a linear chirp, if not, we know that the peak was detected spuriously and can be discarded. To measure the variation we compute the standard deviation of all frequency estimations along the ridge.

6.5 Estimation of Initial Phase

As shown in a previous chapter, the instantaneous frequency and amplitude for each partial tone in each frame is not enough information to resynthesize a signal. It is unescapable to also estimate the initial phase at the center of each frame. Only with the correct phase information a partial tone can be subtracted successfully. Unfortunately, the initial phase information is lost during the calculation of the AF (see appendix A.2).

We estimate then the initial phase by means of the analytic signal. As shown in figure 5.1 on page 36 the analytic associate is already computed before the frame segmentation, and later each partial tone is separated by a band-pass filter, so it is sufficient to just measure the phase angle of the filtered signal at the center of the frame.

Chapter 7
Hidden Markov Models

A Hidden Markov Model (HMM) is a doubly stochastic process with an underlying stochastic process that is *not* observable (that is why it is called "hidden") but can be observed through another set of stochastic processes that produce a sequence of observations. A very good introduction to HMMs is given in [RJ86].

Among the many areas where HMMs are successfully used are speech recognition, optical character recognition and bioinformatics. In [DGR93] it is shown how HMMs can be used for the tracking of partials in additive analysis/synthesis.

In this chapter we give a general overview of the HMM technique and in section 8.2 we show how HMMs can be used for tracking partial tones.

7.1 Structure

A HMM has N hidden *states* $q_1, q_2, \ldots, q_N$. At each clock time t a new state is entered based on a *transition probability* which depends on both the previous and the current state. The probability to change to state q_j provided that the previous state was q_i is a_{ij}. Note that the process can remain in the previous state q_k if the probability a_{kk} is greater than zero. All probabilities a_{ij} form the *transition matrix* A.

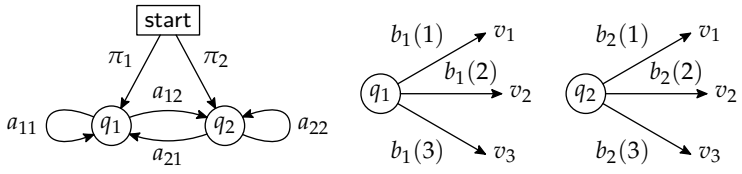

Figure 7.1: *Example for a Hidden Markov Model with $N = 2$ and $M = 3$. The left part shows the two states and their state transition probabilities, the part on the right shows the output probabilities for each of the states.*

The model can generate M observable *output symbols* $v_1, v_2, \ldots, v_M$. Each state q_j has its own probability distribution $b_j(k)$ to emit the output symbol v_k. As there are N states, there are also N probability distributions $b_1(k), b_2(k), \ldots, b_N(k)$ which are combined into an *output probability matrix* B. We are concentrating on HMMs with discrete output symbols, but there are also models with continuous observation variables.

After each transition is made, an observation output symbol O_t is produced according to the output probability distribution b_j of the current state q_j. The whole *observation sequence* O consists of T observations $O_1, O_2, \ldots, O_T$.

At clock time $t = 1$, the first state has to be selected. This is done with the *initial state distribution* $\pi = \{\pi_1, \pi_2, \ldots, \pi_N\}$, where π_i is the probability that the model begins with state q_i at time $t = 1$.

The observation sequence O is generated as follows:

1. Choose an initial state i_1, according to the initial state distribution π
2. Set $t = 1$
3. Output O_t according to $b_{i_t}(k)$, the symbol probability distribution in state i_t
4. Choose i_{t+1} according to $a_{i_t i_{t+1}}$, the state transition probability distribution for state i_t
5. Set $t = t + 1$; return to step 3 if $t < T$, otherwise terminate the procedure

A HMM can be described in compact notation by the probabilities $\lambda = (A, B, \pi)$ and by the two numbers M and N. Figure 7.1 on the previous page shows a simple example for a HMM with two states and three output symbols.

7.2 The Three Problems

A HMM can be used in real world applications solving one (or several) of the following three problems:

Problem 1 How to compute the probability $\mathcal{P}(O|\lambda)$ that a given output sequence O is produced by a given HMM $\lambda = (A, B, \pi)$. This is called the *evaluation problem*. It can also be useful, if two or more competing models are possible, to find out which one has the highest probability of producing the output sequence O.

Problem 2 How to choose a state sequence $I = i_1, i_2, \ldots, i_T$ which is the most probable sequence given the output sequence O. In this problem we try to uncover the hidden state sequence.

The tracking of partial tones falls into this category (see section 8.2).

Problem 3 How to optimize the model parameters $\lambda = (A, B, \pi)$ to get a maximum probability $\mathcal{P}(O|\lambda)$ for a given output sequence O. This is called a *training sequence*.

Problem 1 can either be solved by directly calculating and summing the probabilities for all possible state sequences (this is normally computationally unfeasible) or using the so-called *forward-backward procedure*. Problem 2 can be solved with the *Viterbi algorithm*, and problem 3 can be solved with the *Baum-Welch method*.

For the tracking of partials we are interested only in problem 2, therefore we will describe the appropriate algorithm (the Viterbi algorithm) in the following section. A description of the strategies to solve the two other problems can be found in [RJ86].

7.3 Viterbi Algorithm

The Viterbi algorithm finds for each observation sequence the most probable state sequence of the HMM that would have produced the observations. As described in [RJ86], the Viterbi algorithm uses the following recursive process:

Step 1 – Initialization (for $1 \leq i \leq N$)
For every state q_i the probability that it generates the given first output symbol O_1 at time $t = 1$ is calculated. This probability δ_1 consists of the product of the initial probability π_i and the output probability $b_i(O_1)$ for each state q_i.

- $\delta_1(i) = \pi_i b_i(O_1)$
- $\Psi_1(i) = 0$

Step 2 – Recursion (for $2 \leq t \leq T$ and $1 \leq j \leq N$)
Starting from the probabilities δ_{t-1} calculated in the previous step, all possible state transitions and the probabilities that they generate the given output symbol O_t are evaluated. The maximum probability per state is stored in δ_t.

- $\delta_t(j) = \max\limits_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}]\, b_j(O_t)$

To be able to track back the state sequence in step 4, we have to save the state index from the previous time $t - 1$ with the maximum probability to change to the actual state. This index is saved to Ψ_t for each state.

- $\Psi_t(j) = \operatorname{argmax}\limits_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}]$

Step 3 – Termination
In the set of probabilities of the last time instant $t = T$, a maximum is found which shows the probability P^* of the most probable path taken through the states. The index i_T^* shows the state in which it ended.

- $P^* = \max_{1 \leq i \leq N} [\delta_T(i)]$
- $i_T^* = \operatorname{argmax}_{1 \leq i \leq N} [\delta_T(i)]$

Step 4 – Path (state sequence) backtracking (for $t = T-1, T-2, \ldots, 1$)
To get not only the last state but all the states visited on the optimal path we have to find our way back through the matrix Ψ.

- $i_t^* = \Psi_{t+1}(i_{t+1}^*)$

The resulting state sequence $i_1^*, i_2^*, \ldots, i_T^*$ solves problem 2.

Chapter 8

Tracking of Partials

Once the parameters frequency, amplitude, initial phase and chirp rate have been estimated for each partial candidate in each analysis frame, we try to find possible trajectories that partial tones could have taken during the progression of the original audio signal. The grouping of partials into trajectories has two interests, both of them making the additive analysis/synthesis scheme as powerful and effective as it is.

The first reason is to reduce the number of partial candidates. In general, not all partial candidates can be meaningfully assigned to valid partial trajectories. Those that cannot are not used for synthesis and can be discarded. Hence memory requirements and computational complexity are reduced. Additionally, one may want to use only trajectories with a given minimum length for the same reasons.

The second reason why partials are grouped into trajectories is because the parameters are interpolated along the trajectories during synthesis. This way we store only one value of frequency, amplitude and phase per frame, but we are still able to reconstruct a continuous evolution of the parameters at sampling rate.

8.1 "Straightforward" Approach

In section 1.1 we presented the peak continuation algorithm used by McAulay and Quatieri [MQ86]. This procedure compares the frequencies of partial candidates of adjacent frames and selects those partials as belonging to the same trajectory which have the smallest frequency difference. Additionally, a threshold is used that limits the maximum frequency change a trajectory can incorporate from one frame to the next.

An improvement of this technique was presented in [Kos05]. It uses the chirp rate, estimated by means of the Ambiguity Function (AF), *and* the frequency of each partial candidate to detect if two partial candidates belong to the same trajectory or not. This has the advantage that if there are several partial candidates very close

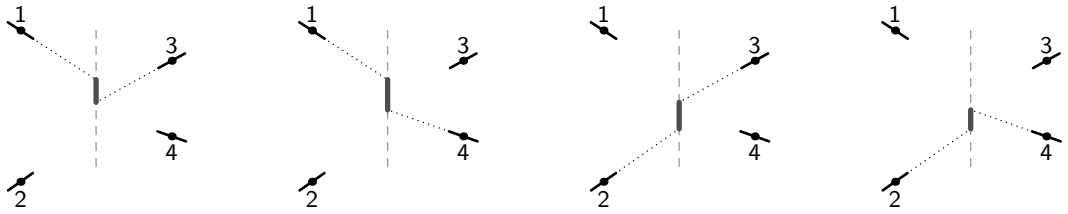

Figure 8.1: *Example where the tracking method used in [Koso5] does not lead to the correct result. The figure to the very right has the smallest frequency difference ($\Delta[2,4]$) and therefore the partial candidates 2 and 4 would be wrongly assigned to the same trajectory.*

in frequency, they can still be correctly assigned to their respective trajectories if their chirp rates are distinct enough.

The method described in [Koso5] calculates the expected frequencies midway between two frames k and $k+1$:

$$f_1[i,k] = f_{i,k} + \frac{T}{2}\,\alpha_{i,k} \quad \text{and} \quad f_2[j,k] = f_{j,k+1} - \frac{T}{2}\,\alpha_{j,k+1},$$

where $f_1[i,k]$ is the expected frequency between frames k and $k+1$ based on the i^{th} frequency and chirp rate of the k^{th} frame, $f_2[j,k]$ is the expected frequency at the same time but based on the j^{th} frequency and chirp rate of the $(k+1)^{\text{th}}$ frame. The indices i and j are limited to $1 \leq i \leq M$ and $1 \leq j \leq N$ with M and N being the number of partial candidates in the k^{th} and $(k+1)^{\text{th}}$ frame, respectively.

Now f_1 and f_2 are compared and the pairs with the smallest frequency difference

$$\Delta_k[i,j] = \left|f_1[i,k] - f_2[j,k]\right| \tag{8.1}$$

are assigned to corresponding trajectories. For further details see [Koso5].

This algorithm is without doubt an improvement over the one used by McAulay and Quatieri [MQ86], but it still can give wrong results, especially when partials are crossing. Figure 8.1 shows a situation of partial candidates and their chirp rates where the algorithm gives the wrong result. Clearly partials 1 and 4 belong to the same trajectory and partials 2 and 3 belong to a different one. The smallest frequency difference, regarding to equation (8.1), is $\Delta[2,4]$, which would mean that partial 2 and 4 would be assigned to one trajectory and partial 1 and 3 to another one.

In this thesis we introduce a slightly modified method for the tracking of partials, which should avoid the aforementioned shortcoming. We do not use the difference between the predicted frequencies between two frames but their differ-

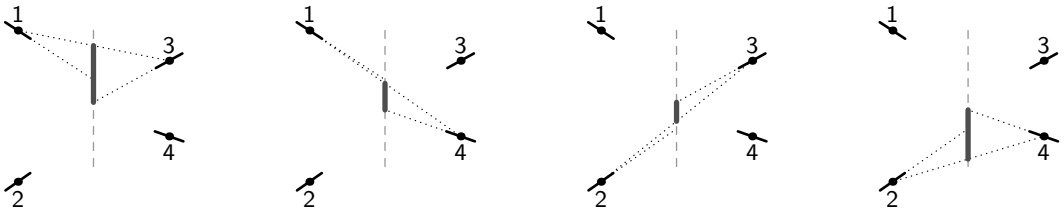

Figure 8.2: *Example for the tracking method proposed in this thesis. The two figures in the middle have the smallest frequency differences ($\Delta[2,3]$ and $\Delta[1,4]$) and therefore the partial candidates are assigned to the correct trajectories.*

ence to the mean frequency of a pair of partial candidates. The new frequency distances are

$$\Delta_k[i,j] = \max\left(\left|f_1[i,k] - \frac{f_{i,k} + f_{j,k+1}}{2}\right|, \left|f_2[i,k] - \frac{f_{i,k} + f_{j,k+1}}{2}\right|\right). \tag{8.2}$$

Using these frequency differences, we can find the corresponding pairs of partial candidates even if their trajectories are crossing. Figure 8.2 shows an example for crossing trajectories (the same as in figure 8.1 on the preceding page), where, using the frequency distances from equation (8.2), the partial candidates are assigned to the correct trajectories.

The matching mechanism for tracking the partials from frame k to frame $k+1$ finds the smallest $\Delta_k[i,j]$ considering all pairs of (i,j). If the smallest distance, $\Delta_k[i_1,j_1]$, is below a given threshold, the partial candidates with the frequencies $f_{i_1,k}$ and $f_{j_1,k+1}$ are assigned to the same trajectory. Then the process is repeated by finding the smallest $\Delta_k[i,j]$ considering all pairs of (i,j) except the ones containing i_1 or j_1. The algorithm continues like that until the smallest difference is bigger than the frequency threshold. All remaining partial candidates are discarded.

8.2 Using A Hidden Markov Model

The most significant advantage of using a Hidden Markov Model (HMM) for the tracking of partials compared to using the other presented techniques is, that the HMM based approach optimizes the trajectories globally, i. e. taking the whole signal into consideration, and not only optimizes locally from frame to frame. The most significant difficulty is, that, as shown below, several free parameters (at least five) have to be manipulated to tune the performance.

In the HMM used for partial tracking [DGR93], the observations are only the numbers of spectral peaks on each side of each frame transition. The states represent the links (i. e. the partials) that connect these peaks. The frequency, amplitude and

chirp rate information is used to compute the transition probabilities between the states.

One important difference of this HMM when compared to the type of HMM described in [RJ86] is the fact that the state transition probabilities evolve over time, since they are obtained from the parameters (frequency, amplitude and chirp rate) of the partial candidates.

The HMM described in [DGR93] uses the following notation: At time k, there are h_k peaks $P_k[j]$, with $0 \leq j < h_k$, ordered by growing frequency. The goal is to assign an index $I_k[j]$ to each peak $P_k[j]$. If a peak does not belong to any trajectory, it is associated with the null index $I_k[j] = 0$.

At each time k, a *state* S_k is defined by an ordered pair of vectors (I_{k-1}, I_k) and the *observation* is defined by an ordered pair of integers (h_{k-1}, h_k). As mentioned above, the frequency and amplitude parameters are not taken as observations. The output probabilities are either one or zero. The output probability of observation of (m, n) is always one for the states defined by ordered pairs of vectors of size m and n, and it is zero for all others.

To calculate the most probable trajectories, the Viterbi algorithm (see section 7.3) is used. From the optimal sequence of states S_k we derive the corresponding sequence of vectors I_k.

The crucial part of the partial tracking algorithm is the calculation of the state transition probabilities. For the transition from a state S_{k-1} into a state S_k, three frames of peaks are concerned, corresponding to times $k-2$, $k-1$ and k, as the states S_{k-1} and S_k are defined by the vectors I_{k-2}, I_{k-1} and I_k. Let $f_k(j)$ and $a_k(j)$ be the frequency and amplitude of the peak $P_k[j]$.

For each peak $P_k[j]$, $0 \leq j < h_k$, we evaluate a *matching criterion* $\theta_k(j)$ which depends on two other peaks $P_{k-2}[t]$ and $P_{k-1}[r]$, such that $I_{k-2}(t) = I_{k-1}(r) = I_k(j)$. This matching criterion is defined by

$$\theta_k(j) = \begin{cases} \theta_k^{(1)}(j) & \text{if } I_k(j) > 0 \\ \theta_k^{(2)}(j) & \text{if } I_k(j) = 0 \end{cases} \tag{8.3a}$$

with

$$\theta_k^{(1)}(j) = \exp\left\{ -\frac{[\Delta f_k(j,r) - \Delta f_{k-1}(r,t)]^2}{\sigma_f^2} - \frac{[\Delta a_k(j,r) - \Delta a_{k-1}(r,t)]^2}{a_k^2(j)\sigma_a^2} \right\} \tag{8.3b}$$

and

$$\theta_k^{(2)}(j) = \left(1 - (1-\mu)\exp\left\{-\frac{[\Delta f_k(j,r) - \Delta f_{k-1}(r,t)]^2}{\sigma_{fn}^2}\right\}\right) \times \left(1 - (1-\mu)\exp\left\{-\frac{[\Delta a_k(j,r) - \Delta a_{k-1}(r,t)]^2}{a_k^2(j)\sigma_{an}^2}\right\}\right), \quad (8.3c)$$

where $\Delta a_k(j,r) = a_k(j) - a_{k-1}(r)$, $f_k(j,r) = f_k(j) - f_{k-1}(r)$ and μ, σ_f, σ_a, σ_{fn} and σ_{an} are free parameters which may vary in time.

Let the states currently considered at times $k-1$ and k be $S_{k-1} = x$ and $S_k = y$, respectively. To obtain the state transition probabilities, we calculate for each frame k a global score $g_{xy}(k)$ over all peaks j:

$$g_{xy}(k) = \prod_{j=0}^{h_k - 1} \theta_k(j).$$

Since the sum over all probabilities from one state x to all other states S_k must be one, we have to normalize the global scores $g_{xy}(k)$ to get the state transition probabilities

$$A_{xy}(k) = \frac{g_{xy}(k)}{\sum_{S_k} g_{xS_k}(k)}.$$

With the state transition probabilities $A_{xy}(k)$ and by means of the Viterbi algorithm, the sequence of the state vectors I_k can be obtained.

"Births" and "deaths" are handled with a sliding analysis window. Within an analysis window the number of trajectories is constant. The minimum length of trajectories is set by the length of the analysis window. As this analysis window is displaced only one frame at a time, "births" and "deaths" can still be detected at each frame.

This analysis window allows a trade-off between computation time and global optimization of partial tracks. Ideally, the whole duration of the input sound would be analyzed at one time, to globally optimize the trajectories. However, for long signals, the computation time would be too long. By using a sliding analysis window, we can gain computation efficiency by sacrificing the global optimization and optimize only within the window.

Including the Chirp Rate

Here we present two different ways of how to include the chirp rate into the calculation of the state transition probabilities. Both are based on the matching criterion

from [DGR93] shown in equation (8.3) on the previous page. The success of the matching criteria heavily depends on the setting of the numerous parameters.

One way to include the chirp rate is to add another term which evaluates the continuity of the slopes of the chirp rates analog to the frequencies and amplitudes in equation (8.3). In this case the new matching criterion is

$$\theta_k^{(1)}(j) = \exp\left\{-\frac{[\Delta f_k(j,r) - \Delta f_{k-1}(r,t)]^2}{\sigma_f^2}\right\} \times \exp\left\{-\frac{[\Delta a_k(j,r) - \Delta a_{k-1}(r,t)]^2}{a_k^2(j)\sigma_a^2}\right\} \times \exp\left\{-\frac{[\Delta \alpha_k(j,r) - \Delta \alpha_{k-1}(r,t)]^2}{\sigma_\alpha^2}\right\} \quad (8.4b)$$

and

$$\theta_k^{(2)}(j) = \left(1 - (1-\mu)\exp\left\{-\frac{[\Delta f_k(j,r) - \Delta f_{k-1}(r,t)]^2}{\sigma_{fn}^2}\right\}\right) \times \left(1 - (1-\mu)\exp\left\{-\frac{[\Delta a_k(j,r) - \Delta a_{k-1}(r,t)]^2}{a_k^2(j)\sigma_{an}^2}\right\}\right) \times \left(1 - (1-\mu)\exp\left\{-\frac{[\Delta \alpha_k(j,r) - \Delta \alpha_{k-1}(r,t)]^2}{\sigma_{\alpha n}^2}\right\}\right), \quad (8.4c)$$

where $\alpha_k(j,r) = \alpha_k(j) - \alpha_{k-1}(r)$, $\alpha_k(j)$ is the j^{th} partial in the k^{th} frame and σ_α and $\sigma_{\alpha n}$ are two additional free parameters which may vary in time.

The second way to incorporate the chirp rate does not use the continuity of the slopes of the parameters but the continuity of the parameters themselves. Because the slope of the frequency holds ideally the same information as the chirp rate, we use only the difference of the frequencies from one frame to the next instead of the slope of the frequencies. The difference of the chirp rates is included in a separate term. This way, the matching criterion becomes

$$\theta_k^{(1)}(j) = \exp\left\{-\frac{[f_k(j) - f_{k-1}(r)]^2}{\sigma_f^2} - \frac{[a_k(j) - a_{k-1}(r)]^2}{a_k^2(j)\sigma_a^2} - \frac{[\alpha_k(j) - \alpha_{k-1}(r)]^2}{\sigma_\alpha^2}\right\} \quad (8.5b)$$

and

$$\theta_k^{(2)}(j) = \left(1 - (1-\mu)\exp\left\{-\frac{[f_k(j) - f_{k-1}(r)]^2}{\sigma_{fn}^2}\right\}\right) \times \left(1 - (1-\mu)\exp\left\{-\frac{[a_k(j) - a_{k-1}(r)]^2}{a_k^2(j)\sigma_{an}^2}\right\}\right) \times \left(1 - (1-\mu)\exp\left\{-\frac{[\alpha_k(j) - \alpha_{k-1}(r)]^2}{\sigma_{\dot{\alpha}n}^2}\right\}\right). \quad (8.5c)$$

Chapter 9
Results

A prototype application was implemented using the concepts shown in chapters 5 and 6 for finding partial candidates and the estimation of the parameters and the algorithm for the tracking of partial candidates presented in section 8.1. This application was tested with sound recordings and with artificial signals.

9.1 Real Recordings

The sound recordings used were the ones from the Sound Description Interchange Format (SDIF) comparison session [WBF+00] at the International Computer Music Conference in the year 2000. These recordings are available for free download (see references).

We compared the two analysis methods, the one based on the Short Time Fourier Transform (STFT) and the one based on the Ambiguity Function (AF), by subtracting the resynthesized sinusoidal representations from the original signal. If one of these residuals has less energy, that means the corresponding algorithm extracted more energy from the sinusoidal part of the sound, thus giving a more accurate representation.

Table 9.1 on the following page shows the amount of attenuation of the residual compared to the original signal. The examples listed first consist of single voiced instrumental phrases. These result in a smaller residual when using the AF analysis. The last two, which have more transient and noisy parts yield a better result using the STFT based analysis. In both cases the analysis was done with a maximum of 25 partials per frame.

9.2 Artificial Signals

The errors of the estimators were evaluated with a single linear chirp contaminated by Gaussian white noise. Tables 9.2 to 9.5 show the estimation errors averaged over 200 frames. The frequency at the center of the frame was 5000 Hz, the chirp

Soundfile	STFT based analysis (dB)	AF based analysis (dB)
leonard.trim	−8.1	−12.8
harris.trim	−9.9	−12.4
suling-phrase	−17.0	−19.4
tatum	−8.1	−9.6
berimbao	−8.0	−7.2
shaku	−19.7	−17.2

Table 9.1: *Power of the residual signals compared to the original signal*

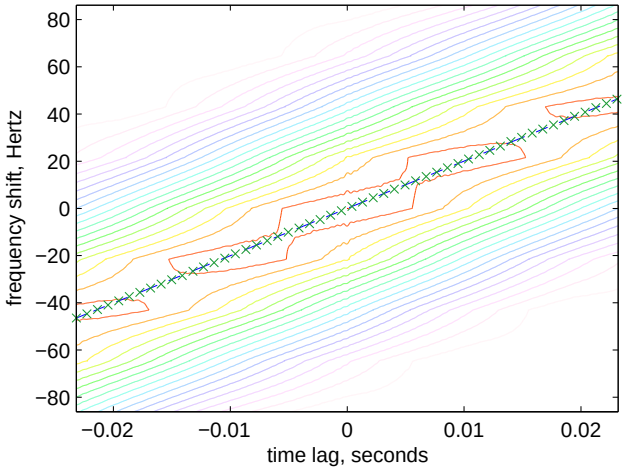

Figure 9.1: *AF of a linear chirp plus white noise. The SNR with respect to a 100 Hz band is 39 dB*

rate was 2000 Hz/s. Figures 9.1 to 9.4 show the AFs for different Signal to Noise Ratios (SNRs). The SNR was calculated with respect to a 100 Hz frequency band. Up to a SNR of 21 dB all the estimators work reasonably well. In figure 9.4 on page 60 the ridge caused by the chirp is still visible and detectable. The green crosses show the detected maxima in the magnitude of the AF and the blue dashed line represents the straight line which was fitted to the maxima by linear regression. According to table 9.5 on page 58, at a SNR smaller than 21 dB the chirp rate has very big errors and is not reliable any more.

To successfully detect crossing partials, the frequencies of the frames next to the crossing have to be sufficiently well separated. The STFT used for spectral analysis can only yield two separate maxima if there is at least one bin between them with lower magnitude. When using a sampling rate of 44.1 kHz and an STFT length of 2048 bins, the two frequency components have to be more than 22 Hz apart to detect them as two separate peaks. That means, to analyze, for

	AF			STFT		
SNR (dB)	mean error (Hz)	std. dev. (Hz)	max. abs. error (Hz)	mean error (Hz)	std. dev. (Hz)	max. abs. error (Hz)
21	−0.04	0.6	1.9	0.2	0.9	2.4
18	−0.03	0.8	2.1	0.3	1.4	3.9
15	0.2	3.3	21.7	0.4	1.9	4.4
12	−0.5	4.9	37.9	0.3	2.9	8.0
9	2.3	19.0	92.3	0.2	4.3	16.0
6	1.2	46.9	187.6	2.9	53.2	343.2
3	−5.1	67.1	185.6	4.4	73.4	350.3

Table 9.2: *Average errors of the frequency estimators for a linear chirp in presence of white Gaussian noise. The SNR is given relative to a* 100 *Hz band.*

	AF			STFT		
SNR (dB)	mean error (%)	std. dev. (%)	max. abs. error (%)	mean error (%)	std. dev. (%)	max. abs. error (%)
21	0.55	3.8	16.2	12.4	3.7	28.1
18	0.63	5.6	15.0	11.5	5.4	29.8
15	0.1	7.3	26.3	11.3	6.5	28.5
12	0.1	11.0	39.8	10.4	9.8	33.0
9	−3.3	15.8	46.5	7.6	14.4	37.8
6	−5.6	16.1	44.3	3.7	14.8	38.3
3	−13.1	16.5	66.0	−15.3	22.0	62.3

Table 9.3: *Average errors of the amplitude estimators for a linear chirp in presence of white Gaussian noise. The SNR is given relative to a* 100 *Hz band.*

	AF			STFT		
SNR (dB)	mean err. (π rad)	std. dev. (π rad)	max. abs. err. (π rad)	mean err. (π rad)	std. dev. (π rad)	max. abs. err. (π rad)
21	−0.01	0.06	0.17	−0.11	0.01	0.15
18	−0.01	0.1	0.26	−0.11	0.02	0.15
15	−0.01	0.14	0.42	−0.11	0.03	0.2
12	−0.03	0.24	0.94	−0.11	0.04	0.21
9	0.01	0.29	0.91	−0.1	0.05	0.26
6	0.02	0.37	0.98	−0.08	0.13	0.53
3	−0.06	0.47	0.99	−0.09	0.26	0.92

Table 9.4: *Average errors of the initial phase estimators for a linear chirp in presence of white Gaussian noise. The SNR is given relative to a 100 Hz band.*

	AF		
SNR (dB)	mean error (Hz/s)	std. dev. (Hz/s)	max. abs. error (Hz/s)
21	−3	210	689
18	87	469	2459
15	24	719	4883
12	127	1129	4708
9	422	2605	10769

Table 9.5: *Average errors of the chirp rate estimators for a linear chirp in presence of white Gaussian noise. The SNR is given relative to a 100 Hz band.*

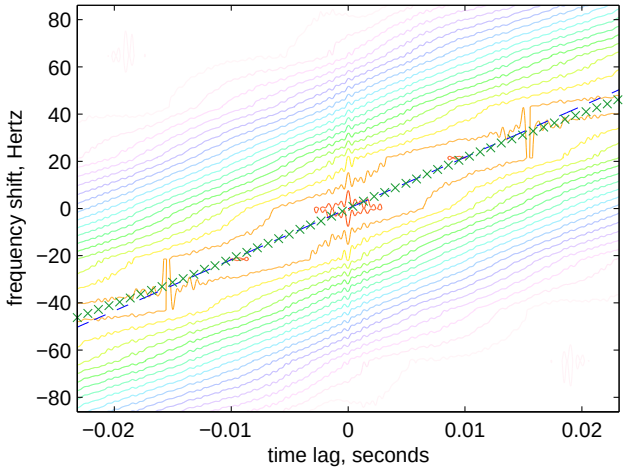

Figure 9.2: *AF of a linear chirp plus white noise. The SNR with respect to a 100 Hz band is 33 dB*

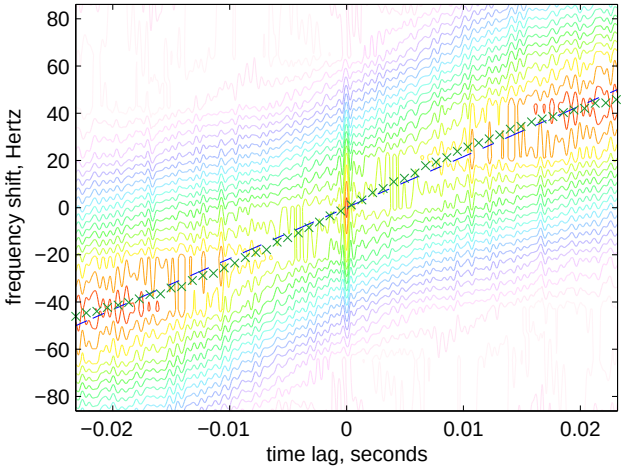

Figure 9.3: *AF of a linear chirp plus white noise. The SNR with respect to a 100 Hz band is 27 dB*

example, two chirps with chirp rates of 2000 Hz/s and −2000 Hz/s, respectively, the center of the frame has to be more than 250 samples away from the point of crossing, which means on the other hand that the hopsize has to be at least 500 samples. A tradeoff has to be found between minimum detected chirp rate difference, STFT length, sampling frequency and hopsize. Any attempt to separate partials which are closer in frequency to each other, results in some sort of reduced time resolution.

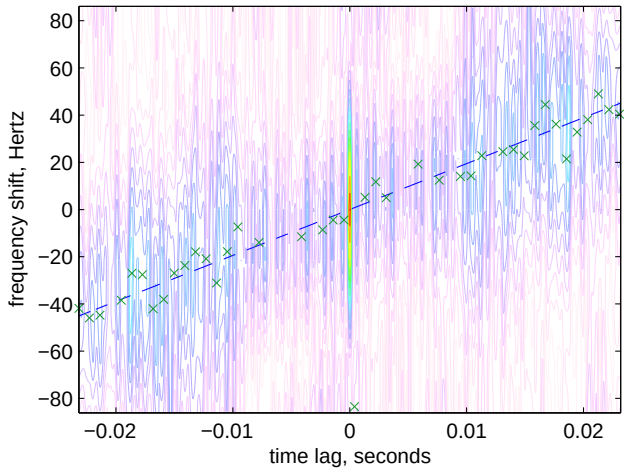

Figure 9.4: *AF of a linear chirp plus white noise. The SNR with respect to a* 100 *Hz band is* 21 *dB*

Chapter 10

Conclusions and Future Work

We presented a method for parameter estimation which uses the Ambiguity Function (AF), a bilinear time-frequency distribution, to improve the estimated values for frequency, amplitude and initial phase compared to the more traditional method using Short Time Fourier Transforms (STFTs). Comparing the two methods using artificial signals with known parameters showed that the AF based amplitude estimators are significantly better and the frequency estimators are most of the times slightly better than the STFT based estimators. The AF also provides an estimator for the *chirp rate*, which the STFT does not.

Two different possibilities to find trajectories of partials using the estimated parameters were presented thereafter. One based on the work in [MQ86] and [Kos05], the other one based on a Hidden Markov Model (HMM).

Crossing Trajectories

The peak tracking algorithm used in [Kos05] was enhanced to handle eventually crossing partial tones in a better way. However, as mentioned earlier, the problem with crossing partials is that the components can not be separated sufficiently well before AF processing. One future task could be to examine, if two (or even more) partials at the same time can be detected in the output of the AF. In this case, care has to be taken of the cross terms.

Linear Chirps

In this thesis we expected the partial tones to be linear chirps (within a given frame). The estimators for frequency, amplitude and chirp rate were based on this assumption. One could try to find estimators for partials with less restrictions, for example assume a quadratic chirp or an exponential chirp.

Constant Amplitude

We were estimating amplitude in a way which assumed constant amplitude per frame. More flexibility would be reached, if one would allow amplitude changes within a frame (for example, a linear amplitude evolution).

Synthesis

The chirp rate estimated by means of the AF was used to improve the tracking of partial tone trajectories. It could also be used in the synthesis process. As shown in [GMDM+03], this could significantly improve the quality of synthesis.

Computational Efficiency

Not much effort has been made to increase the computational efficiency of the implementation. One way to improve efficiency tremendously would be to use undersampling after the band-pass filtering. As the bandwidth is reduced by the band-pass filters, the sampling rate can also be reduced without losing any information. With the reduced sampling rate the number of points calculated for the AF would be reduced. As the AF is two-dimensional this would increase the performance significantly.

Improvements in performance could also be made by reimplementing the programs in a lower level programming language than MATLAB®. In the current state of the implementation, however, using MATLAB® is a good choice, because it encourages rapid prototyping and significant changes in algorithms can be realized faster than in a low level language.

Appendices

Appendix A

Some Calculations

In this section of the appendix we do a few calculations which are too long to make it into the main part of the thesis. Nonetheless we did not want to shorten them to make it easy to reproduce the calculations.

A.1 A Linear Chirp

An analytic sinusoidal oscillation with time-varying frequency can be defined as

$$z(t) = a\,\mathrm{e}^{j\phi(t)},$$

with a being its (in this case constant) amplitude and $\phi(t)$ being its instantaneous phase. The instantaneous frequency is proportional to the derivative of the instantaneous phase:

$$f(t) = \frac{1}{2\pi}\frac{d\phi}{dt}.$$

A linear chirp features a linear frequency variation defined by f_0 (the instantaneous frequency at time $t = 0$) and the *chirp rate* $\alpha = {}^{\Delta f}/_{\Delta t}$, which denotes the frequency difference per unit time. Hence, the instantaneous frequency of a linear chirp can be written as

$$f(t) = f_0 + \alpha t.$$

We get the instantaneous phase $\phi(t)$ by integrating the instantaneous frequency from a given time t_s to t (the constant $\phi_s = \phi(t_s)$ denotes the instantaneous phase at time t_s):

$$\begin{aligned}\phi(t) &= 2\pi \int_{t_s}^{t} f(\xi)\,d\xi + \phi_s = 2\pi \int_{t_s}^{t} (f_0 + \alpha\xi)\,d\xi + \phi_s \\ &= 2\pi \left[f_0\xi + \alpha\frac{\xi^2}{2} \right]_{\xi=t_s}^{t} + \phi_s = 2\pi \left(f_0 t + \alpha\frac{t^2}{2} - f_0 t_s - \alpha\frac{t_s^2}{2} \right) + \phi_s.\end{aligned}$$

We can combine the constant phase contributions to $\phi_0 = \phi_s - 2\pi f_0 t_s - \pi\alpha t_s^2$, where ϕ_0 is the *initial phase*, i. e. the instantaneous phase at time $t = 0$. Using this, a linear chirp can be written as

$$z(t) = a\,\mathrm{e}^{j\left(2\pi f_0 t + \pi\alpha t^2 + \phi_0\right)}. \tag{A.1}$$

A.2 The Ambiguity Function of a Linear Chirp

Inserting the linear chirp signal $z(t)$ from equation (A.1) into the definition of the Ambiguity Function (AF) in equation (4.6) on page 30 gives

$$\mathcal{A}_z(\nu,\tau) = \int a\,\mathrm{e}^{j\left(2\pi f_0\left(t+\frac{\tau}{2}\right)+\pi\alpha\left(t+\frac{\tau}{2}\right)^2+\phi_0\right)}\, a\,\mathrm{e}^{-j\left(2\pi f_0\left(t-\frac{\tau}{2}\right)+\pi\alpha\left(t-\frac{\tau}{2}\right)^2+\phi_0\right)}\,\mathrm{e}^{-j2\pi\nu t}\,dt.$$

When we add the exponents, many terms cancel each other out – including the constant phase terms – resulting in

$$\mathcal{A}_z(\nu,\tau) = a^2\,\mathrm{e}^{j2\pi f_0\tau}\int \mathrm{e}^{j2\pi(\alpha\tau-\nu)t}\,dt. \tag{A.2}$$

This shows clearly that any initial phase information is lost when calculating the AF. Thus we have to find other means of determining the initial phase in the analysis process. We use the *analytic signal* for that task (see section 6.5).

Solving the integral in the previous equation produces an expression for the AF of a linear chirp:

$$\mathcal{A}_z(\nu,\tau) = a^2\,\delta(\nu-\alpha\tau)\,\mathrm{e}^{j2\pi f_0\tau}, \tag{A.3}$$

where $\delta(f)$ represents the *Dirac distribution*. The magnitude of this expression has the value a^2 at all points where $\alpha\tau = \nu$, i. e. on a straight line through the origin of the ambiguity plane. In all other cases the magnitude is zero. The phase is independent of ν and linearly related to τ.

In practice, however, we are never dealing with infinite duration signals. Thus we have to impose finite limits to the integral in equation (A.2). When we use an integration interval of T and hence set the limits to $-T/2$ and $T/2$ this entails that the signal $z(t)$ must be defined for all t in the range $-(T/2+\tau/2) < t < (T/2+\tau/2)$. Solving the integral with the new limits leads to

$$\begin{aligned}\mathcal{A}_z(\nu,\tau) &= \frac{a^2\,\mathrm{e}^{j2\pi f_0\tau}}{j2\pi(\alpha\tau-\nu)}\left[\mathrm{e}^{j2\pi t(\alpha\tau-\nu)}\right]_{t=-T/2}^{T/2}\\ &= \frac{a^2\,\mathrm{e}^{j2\pi f_0\tau}}{j2\pi(\alpha\tau-\nu)}\left(\mathrm{e}^{j\pi T(\alpha\tau-\nu)} - \mathrm{e}^{-j\pi T(\alpha\tau-\nu)}\right).\end{aligned}$$

Knowing that $\frac{e^{jx}-e^{-jx}}{2j} = \sin x$, we get

$$\mathcal{A}_z(\nu,\tau) = a^2 \, \frac{\sin(\pi T(\alpha\tau-\nu))}{\pi(\alpha\tau-\nu)} \, e^{j2\pi f_0 \tau} .$$

Finally, using the *sinus cardinalis*, defined by $\operatorname{sinc} x = \frac{\sin x}{x}$, the AF of a linear chirp becomes

$$\mathcal{A}_z(\nu,\tau) = a^2 T \operatorname{sinc}(\pi T(\alpha\tau-\nu)) \, e^{j2\pi f_0 \tau} . \tag{A.4}$$

As in equation (A.3) on the facing page, the magnitude of this expression has its maximum where $\alpha\tau = \nu$. The temporal windowing only turns the Dirac impulses into two-dimensional sinc functions.

In this case an estimator for the amplitude is easily found by taking the square root of the magnitude, divided by T, at any point on the ridge:

$$a = \sqrt{\frac{|\mathcal{A}_z(\nu,\tau)|}{T}}, \text{ wherever } \alpha\tau = \nu. \tag{A.5}$$

The phase of the AF, however, is not affected at all by the time restriction.

A.3 The Windowed Ambiguity Function of a Linear Chirp

This time we insert the linear chirp signal from equation (A.1) on the preceding page into the definition of the windowed AF, shown in equation (6.1) on page 39. Simplifying analogously to section A.2 yields

$$\mathcal{A}_z(\nu,\tau;w) = a^2 \, e^{j2\pi f_0 \tau} \int w(t) \, e^{j2\pi(\alpha\tau-\nu)t} \, dt.$$

In this equation, the integral over the window and the exponential function can be seen as the Fourier transform of the window $w(t)$

$$W(\eta) = \underset{t\to\eta}{\mathcal{F}}\{w(t)\} = \int w(t) \, e^{-j2\pi\eta t} \, dt,$$

with the frequency η substituted by the term $(\nu - \alpha\tau)$. With this definition the windowed AF of a linear chirp becomes

$$\mathcal{A}_z(\nu,\tau;w) = a^2 W(\nu-\alpha\tau) \, e^{j2\pi f_0 \tau}. \tag{A.6}$$

Compared with equation (A.4) this shows clearly that the time restriction to the interval T means applying a rectangular window, whose Fourier transform has the shape of a sinc function.

Equation (A.6) on the preceding page provides us with the following estimator for the amplitude:

$$a = \sqrt{\frac{\left|\mathcal{A}_z(\nu, \tau; w)\right|}{\left|W(0)\right|}}, \text{ wherever } \alpha\tau = \nu. \tag{A.7}$$

When using digital signals, $W(0)$ can be easily obtained by summing all samples of the window $w(t)$ without the need to actually perform the discrete Fourier transform.

As we already saw in the previous chapter, the phase of the AF is independent of the window.

Appendix B

Implementational Details

In the main part of this thesis most of the presented concepts use continuous variables. When implementing them in software, however, we must use discrete variables and therefore we have to adapt some of the formulations. Sometimes when discretizing continuous variables, problems appear which did not exist in the continuous domain. Those adaptations and problems (with solutions) are shown here. As they are not necessary for the general understanding but only for the understanding of the actual software implementation, they are presented in the appendix.

B.1 Hilbert Transform/Analytic Signal

The Hilbert transform, which provides the analytic signal, has an infinite impulse response and cannot be implemented in discrete time without loss of information. There is no perfect implementation, only approximations.

As shown in chapter 3, the analytic signal has a one-sided Fourier transform, i. e. there are no negative frequencies. To approximate the analytic signal, one can calculate the Fast Fourier Transform (FFT) of the input sequence, replace those FFT coefficients which correspond to negative frequencies with zeros and calculate the inverse FFT of the result. In detail, we use a four-step algorithm [Mar99]:

1. Calculate the FFT of the input sequence, storing the result in a vector y. Before transforming, zero pad the input sequence so its length N is the closest power of two, if necessary. This ensures the most efficient FFT computation.

2. Create a vector h whose elements $h(i)$ have the values

   - 1 for $i = 1, {}^{N}/_{2} + 1$,
   - 2 for $i = 2, 3, \ldots, {}^{N}/_{2}$,
   - 0 for $i = {}^{N}/_{2} + 2, \ldots, N$.

3. Calculate the element-wise product of y and h.
4. Calculate the inverse FFT of the sequence obtained in step 3 and return the first N elements of the result.

B.2 Filter Design

It is very important that the band-pass filters used to isolate the partial tones of each frame do not change the relative phase of the partials, because only if the phase of the isolated partial tones is the same as the phase of the partial tones within the original sound (and if the phase is estimated correctly), the resynthesized sound can be subtracted successfully and all tonal components can be removed when calculating the residual signal. We use *linear phase filters* (the phase response of the filter is a linear function of frequency) with a Finite Impulse Response (FIR), because they satisfy the aforementioned requirements.

There are exactly four types of linear phase filters. The type of the filter is determined by the type of symmetry and whether the number of samples is even or odd. Each type of linear phase filter has specific phase and magnitude response characteristics that help determine their applicability. For example types 3 and 4, which both have odd impulse response symmetry, introduce a 90° degree phase shift to the signal they filter. A type 2 linear phase filter (even impulse response symmetry and even number of coefficients) always has a magnitude response of zero at the Nyquist frequency. This study will use type 1 linear phase filters, which have even symmetry and an odd number of samples. Type 1 FIR filters have a benefit that no other type of FIR filters possess in that they can be used to implement any arbitrary magnitude response and have uniform phase delay. The phase delay for type 1 linear phase filters is fixed for all frequencies. For a N sample filter the phase delay is $\frac{N-1}{2}$ samples. This study is not performed in real time, thus arbitrarily long filters can be created without concern for the perceptual consequence of a large delay; the delays can simply be subtracted from the filtered signal.

An efficient algorithm for the construction of a type 1 linear phase filter with an arbitrary magnitude response is presented in [Sal98]. In this method, the response of the filter is constructed in the discrete frequency domain, and the impulse response is computed by means of an inverse FFT.

Because the magnitude and phase response are symmetric, they need only to be defined for half the sequence, since the second half of the response can be determined from the first. This is shown for the magnitude response $A[k]$ as

$$A[k] = \begin{cases} A[k] & \text{for } 0 \leq k \leq \frac{N-1}{2} \\ A[N-k] & \text{for } \frac{N+1}{2} \leq k \leq N-1 \end{cases}$$

and the phase $\theta[k]$ as

$$\theta[k] = \begin{cases} -\pi k \frac{N-1}{N} & \text{for } 0 \leq k \leq \frac{N-1}{2} \\ \pi(N-k)\frac{N-1}{N} & \text{for } \frac{N+1}{2} \leq k \leq N-1. \end{cases}$$

The complex frequency response is defined in the frequency domain as

$$H[k] = |H[k]|\, \mathrm{e}^{j\theta[k]},$$

from which the impulse response of the filter can be computed by taking the inverse Fourier transform.

B.3 Discrete Ambiguity Function

We cannot discretize equation (4.6) on page 30 by simply replacing the continuous variables t, τ and ν by discrete variables because the equation contains the term $\frac{\tau}{2}$. But when we rewrite the expression, replacing $\frac{\tau}{2}$ with θ, we get

$$\mathcal{A}_z(\nu, \theta) = \int z(t+\theta)\, z^*(t-\theta)\, \mathrm{e}^{-j2\pi\nu t}\, dt, \tag{B.1}$$

which can now easily be discretized. However, this modification means essentially an undersampling of the variable τ (which is a time variable) and the consequence is that the allowed bandwidth of the signal $z[n]$ is limited to half the *Nyquist* frequency. The maximum allowed frequency which avoids aliasing in the ambiguity domain is $f_s/4$ (where f_s is the sampling frequency). See [Coh95] or [Boa03] for a proof.

In our case we can fortunately still use signals $x[n]$ with frequencies up to the Nyquist frequency $f_s/2$ when we transform it to $z[n]$, the analytic associate of $x[n]$, which has only half the bandwidth (see section 3.3).

When discretizing equation (B.1), we have to replace the infinite integral with a finite sum. When we calculate the sum over N time samples, use k for the discrete counterpart of ν, m for the discrete θ and n for the discrete time variable, we get

$$\mathcal{A}_z[k, m] = \sum_{|n|<\frac{N}{2}} z[n+m]\, z^*[n-m]\, \mathrm{e}^{-j2\pi kn/N}. \tag{B.2}$$

To calculate $\mathcal{A}_z[k, m]$ for arbitrary k and m, we need the values of $z[n]$ for $-\frac{N}{2} - m \leq n \leq \frac{N}{2} + m$. Equation (B.2) can be written more compactly as

$$\mathcal{A}_z[k, m] = \underset{n\to k}{\mathsf{DFT}} \left\{ z[n+m]\, z^*[n-m] \right\},$$

where $\mathsf{DFT}\{x[n]\}$ stands for the *discrete Fourier transform* of $x[n]$, defined by

$$\underset{n\to k}{\mathsf{DFT}}\{x[n]\} = \sum_{n=0}^{N-1} x[n]\, \mathrm{e}^{-j2\pi kn/N}\,.$$

Analogously, the *discrete windowed AF* of $z[n]$, defined for continuous variables in equation (6.1) on page 39, can be written as

$$\mathcal{A}_z[k,m;w] = \underset{n\to k}{\mathsf{DFT}}\{w[n]z[n+m]\, z^*[n-m]\}\,, \tag{B.3}$$

where $w[n]$ is the discrete window function.

B.4 Ridge Detection

To obtain the chirp rate estimator for a filtered frame, we search for the absolute maxima of $\mathcal{A}_z[k,m;w]$, defined in equation (B.3), along the frequency shift (k) dimension.

Parabolic Interpolation

In the discrete domain, when using a FFT-length of N to calculate the Discrete Fourier Transform (DFT) in $\mathcal{A}_z[k,m;w]$, the frequency shift values are limited to $\nu = \frac{kf_s}{N}$ Hz, for k from 0 to $N-1$. To remove this restriction on the set of possible Doppler values and extend the range to any floating point number, we use parabolic interpolation [KM02].

Parabolic interpolation is derived from the notion that a local maximum of a Fourier transform resembles a parabola when viewed in a logarithmic scale. This is also true for the Ambiguity Function (AF) along the k-dimension, since it is the Fourier transform of the Instantaneous Autocorrelation Function (IAF) as shown in equation (4.7) on page 30.

To calculate the parabolic interpolation, three points have to be known: The sample of the local maximum at $k_{\max}$ with the value $A(k_{\max})$, the preceding sample at $k_{\max}-1$ with the value $A(k_{\max}-1)$ and the one at $k_{\max}+1$ with the value $A(k_{\max}+1)$. To simplify the equations, we introduce

$$l = k - k_{\max}. \tag{B.4}$$

This way, the new indices l have the values -1, 0, and $+1$ and we call the three corresponding function values A_{-1}, A_0 and A_1, respectively. The parameters a, b and c of the parabolic function

$$A = a + bl + cl^2 \tag{B.5}$$

have to be determined in a way that the equation holds true for all of the three given points. When we insert the known pairs $(-1, A_{-1})$, $(0, A_0)$ and $(1, A_1)$ into equation (B.5), we get

$$a = A_0,\ b = \frac{A_1 - A_{-1}}{2} \text{ and } c = \frac{A_{-1} + A_1 - 2A_0}{2}.$$

We can now find a maximum of equation (B.5) on the preceding page by differentiating it and setting it to zero:

$$\frac{dA}{dl} = b + 2cl = 0.$$

Consequently, the maximum is at $l_{\text{interp}} = -\frac{b}{2c}$. Using the three given A-values, this can be written as

$$l_{\text{interp}} = \frac{1}{2}\frac{A_{-1} - A_1}{A_{-1} + A_1 - 2A_0}.$$

To get the position of the maximum found by interpolation we just have to undo the change of the variable from k to l by

$$k_{\text{interp}} = l_{\text{interp}} + k_{\text{max}},$$

and finally, the value of the maximum at k_{interp} is

$$A(k_{\text{interp}}) = A_0 - \frac{l_{\text{interp}}}{4}(A_{-1} - A_1).$$

Zero Padding of the FFT in the AF

If $z[n]$ is a linear chirp, the magnitude of the discrete windowed AF is (compare equation (A.6) on page 67):

$$|\mathcal{A}_z[k, m; w]| = \left| a^2\, W\left[k - \frac{\alpha 2N}{f_s^2} m\right] \right|.$$

If, for example, $w[n]$ is a rectangular window of length T then the above expression in the k-dimension (on which we search the maxima) has the shape of a sinc. The width of the main lobe depends on the length T of the window $w[n]$. If $T = N$, the main lobe is only two samples wide, which means that only two of the three samples needed for parabolic interpolation are actually inside the main lobe, leading to a wrong result of the interpolation. This problem can be fixed by zero-padding the signal before the FFT, which results in $N > T$.

Alternatively (or additionally), a window with a broader main lobe in its Fourier transform can be chosen. In either case, the three samples used for the parabolic

interpolation should be well inside the main lobe of the Fourier transform $W[k]$ of the window $w[n]$.

Linear Regression

Theoretically, all global maxima along the k-dimension of the AF of a linear chirp lie on a straight line. To estimate the chirp rate, it would then be sufficient to use the maximal k for an arbitrary value of m.

Practically, this is not the case. We have to find a straight line which is as close as possible to the maxima for all m. This process is called *linear regression*. We search for a straight line $k = am + b$ which is closest to the extracted maxima (m_i, k_i) in the *least square* sense. This means, that the sum of the squared differences from the points to the line is minimal.

We assume that the line passes through the origin of the ambiguity plane and therefore its equation simplifies to $k = am$. The parameter a is proportional to the chirp rate α. The sum of the squared differences is

$$\sum_i \Delta_i^2 = \sum_i (am_i - k_i)^2.$$

Differentiating this with respect to a and setting it to zero leads to

$$2\sum_i (am_i - k_i)m_i = 0 \quad \text{and}$$

$$a = \frac{\sum m_i k_i}{\sum m_i^2}.$$

We can scale a to obtain the estimated chirp rate α in Hz/s:

$$\alpha = \frac{\Delta f}{\Delta t} = a\frac{f_s^2}{2N}.$$

Appendix C

Spectral Modeling Synthesis

In [Kos05] a software called SMSTools was used to obtain frequencies for the band-pass filters which ideally extract single isolated partials from a frame. SMSTools is freely available from the CLAM homepage[1] at Pompeu Fabra University (UPF). It is an implementation of the Spectral Modeling Synthesis (SMS) concept[2].

Spectral Modeling Synthesis

SMS is a variant of the Deterministic + Stochastic (D+S) model presented in section 1.2. It uses Short Time Fourier Transforms (STFTs) for spectral estimation and it also uses STFTs for the representation of the residual part. SMS is described in [SBHL97].

SMSTools writes the data for the sinusoidal trajectories (consisting of frequency, amplitude and initial phase values for each partial tone in each frame) and the residual part into data files using the Sound Description Interchange Format (SDIF).

Sound Description Interchange Format

The SDIF was developed beginning from 1995 at the Center for New Music and Audio Technologies (CNMAT)[3] in Berkeley and at the Institut de Recherche et Coordination Acoustique/Musique (IRCAM)[4] in Paris. Details about the file format can be found in [WCF+99].

sdif-matlab

SDIF files can be loaded within MATLAB® using the sdif-matlab[5] import functions.

So far, so good

In this thesis, however, we decided to implement the spectral estimation within the MATLAB® application, so that there is no need for data exchange with an ex-

[1] http://clam.iua.upf.edu/
[2] http://www.iua.upf.edu/sms/
[3] http://www.cnmat.berkeley.edu/SDIF/
[4] http://www.ircam.fr/sdif/
[5] http://recherche.ircam.fr/equipes/analyse-synthese/sdif/download/sdif-matlab/

ternal application. The spectral estimation procedure employed in our prototype application is described in chapter 5.

Another reason not to use SMSTools anymore was, because it performs a partial tracking and the calculation of the residual part, while we only need a set of candidate frequencies per frame to control the band-pass filter bank. This frequency information had to be extracted from the representation of the trajectories. It turned out to be much more straightforward to implement the frequency estimation within the prototype application.

List of Abbreviations

AF Ambiguity Function

CNMAT Center for New Music and Audio Technologies (Berkeley, USA)

DFT Discrete Fourier Transform

D+S Deterministic + Stochastic

FFT Fast Fourier Transform

FIR Finite Impulse Response (filters)

FM Frequency Modulation

HMM Hidden Markov Model

IAF Instantaneous Autocorrelation Function

IRCAM Institut de Recherche et Coordination Acoustique/Musique (Paris, France)

SDIF Sound Description Interchange Format

SMS Spectral Modeling Synthesis

SNR Signal to Noise Ratio

STFT Short Time Fourier Transform

TFD Time-Frequency Distribution

UPF Pompeu Fabra University (Barcelona, Spain)

WD Wigner Distribution

WVD Wigner-Ville Distribution

List of Symbols

t, τ	time variables
f, ν	frequency variables
$x(t)$	continuous time signal
$x[n]$	discrete time signal
j	$\sqrt{-1}$
z^*	complex conjugate of z
$\lvert z \rvert$	magnitude of z
$\mathcal{R}e\{z\}$	real value of z
$\int$	integral; if no limits are specified, $\int_{-\infty}^{\infty}$ is assumed
$\delta(f)$	Dirac distribution: $\delta(f) = 0$ for $f \neq 0$, $\int \delta(f)\, df = 1$
f_s	sampling frequency
$\mathcal{F}\{x(t)\}$	Fourier transform of $x(t)$, page 17
$\mathcal{F}^{-1}\{x(t)\}$	inverse Fourier transform of $x(t)$, page 17
$\mathcal{H}\{x(t)\}$	Hilbert transform of $x(t)$, page 22
$\mathcal{K}_x(t, \tau)$	instantaneous autocorrelation function of $x(t)$, page 25
$\mathcal{W}_x(t, f)$	Wigner(-Ville) distribution of $x(t)$, page 25
$\mathcal{A}_x(\nu, \tau)$	ambiguity function of $x(t)$, page 27
$\rho_x(t, f; \gamma)$	Cohen's class distribution of $x(t)$ with kernel γ, page 28
$\mathsf{DFT}\{x[n]\}$	discrete Fourier transform of $x[n]$, page 56

Bibliography

[BOA92A] Boualem Boashash. *Estimating and interpreting the instantaneous frequency of a signal—part 1: Fundamentals*. Proceedings of the IEEE, 80(4):520–538, April 1992.

[BOA92B] Boualem Boashash. *Estimating and interpreting the instantaneous frequency of a signal—part 2: Algorithms and applications*. Proceedings of the IEEE, 80(4):540–568, April 1992.

[BOA03] Boualem Boashash, editor. *Time Frequency Signal Analysis and Processing, A Comprehensive Reference*. Elsevier Ltd., 2003.

[COH95] Leon Cohen. *Time-Frequency Analysis: Theory and Applications*. Prentice Hall PTR, 1995.

[CT65] James W. Cooley and John W. Tukey. *An algorithm for the machine calculation of complex Fourier series*. Mathematics of Computation, 19(90):297–301, April 1965.

[DGR93] Philippe Depalle, Guillermo García and Xavier Rodet. *Tracking of partials for additive sound synthesis using hidden Markov models*. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 1, pages 225–228. Minneapolis, MN, USA, April 1993.

[FLA99] Patrick Flandrin. *Time-Frequency/Time-Scale Analysis*, volume 10 of *Wavelet Analysis and Its Applications*. Academic Press, 1999. Translated from the French by Joachim Stöckler.

[FOU22] Joseph Fourier. *Théorie analytique de la chaleur*. Firmin Didot, Paris, 1822. Reprint by Éditions Jacques Gabay, 1988.

[GMDM+03] Laurent Girin, Sylvain Marchand, Joseph di Martino, Axel Röbel and Geoffroy Peeters. *Comparing the order of a polynomial phase model for the synthesis of quasi-harmonic audio signals*. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 193–196. New Paltz, NY, USA, October 2003.

[Har78] Frederic J. Harris. *On the use of windows for harmonic analysis with the discrete fourier transform*. Proceedings of the IEEE, 66(1):51–83, January 1978.

[KM02] Florian Keiler and Sylvain Marchand. *Survey on extraction of sinusoids in stationary sounds*. In *Proceedings of the 5th International Conference on Digital Audio Effects (DAFx-02)*, pages 51–58. Hamburg, Germany, September 2002.

[Kos05] Paul Kosek. *Improved Analysis of Musical Sounds Using Time-Frequency Distributions*. Master's thesis, McGill University, Montreal, QC, Canada, September 2005.

[Mar99] S. Lawrence Marple, Jr. *Computing the discrete-time "analytic" signal via FFT*. IEEE Transactions on Signal Processing, 47(9):2600–2603, September 1999.

[MB95] Paul Masri and Andrew Bateman. *Identification of non-stationary audio signals using the FFT*. In *Proceedings of the IEE Colloquium on Audio Engineering*, pages 11/1–6. London, UK, May 1995.

[MQ86] Robert J. McAulay and Thomas F. Quatieri. *Speech analysis/synthesis based on a sinusoidal representation*. IEEE Transactions on Acoustics, Speech, and Signal Processing, ASSP-34(4):744–754, August 1986.

[Pic97] Bernard Picinbono. *On instantaneous amplitude and phase of signals*. IEEE Transactions on Signal Processing, 45(3):552–560, March 1997.

[RD92] Xavier Rodet and Philippe Depalle. *Spectral envelopes and inverse FFT synthesis*. In *Proceedings of the Audio Engineering Society 93rd Convention*. San Francisco, CA, USA, October 1992.

[RFB94] Andrew Reilly, Gordon Frazer and Boualem Boashash. *Analytic signal generation—tips and traps*. IEEE Transactions on Signal Processing, 42(11):3241–3245, November 1994.

[RJ86] Lawrence R. Rabiner and Biing-Hwang Juang. *An introduction to hidden Markov models*. IEEE ASSP Magazine, pages 4–16, January 1986.

[Sal98] Sana Salous. *The design of linear phase FIR filters using the IDFT*. IEEE Transactions on Education, 41(3):229–231, 1998.

[SBHL97] Xavier Serra, Jordi Bonada, Perfecto Herrera and Ramon Loureiro. *Integrating complementary spectral models in the design of a musical synthesizer*. In *Proceedings of the International Computer Music Conference*. *Θεσσαλονίκη*, Greece, September 1997.

[SS90] Xavier Serra and Julius Smith III. *Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition*. Computer Music Journal, 14(4):12–24, 1990.

[Vil48] Jean A. Ville. *Théorie et applications de la notion de signal analytique*. Câbles et Transmission, 2A:61–74, 1948.

[WBF+00] Matthew Wright, James Beauchamp, Kelly Fitz, Xavier Rodet, Axel Röbel, Xavier Serra and Gregory Wakefield. *Analysis/synthesis comparison*. Organised Sound, 5(3):173–189, 2000.
Audio files: `http://www.cnmat.berkeley.edu/SDIF/ICMC2000/`.

[WCF+99] Matthew Wright, Amar Chaudhary, Adrian Freed, Sami Khoury and David Wessel. *Audio applications of the sound description interchange format standard*. In *Proceedings of the Audio Engineering Society 107th Convention*. New York, NY, USA, September 1999.

[Wig32] Eugene Paul Wigner. *On the quantum correction for thermodynamic equilibrium*. Physics Review, 40:749–759, 1932.