Audio Source Separation via the Grouping of Partials in the Sum-of-Sinusoids Model

Nicholas Esterer

Department of Music Research
Schulich School of Music
McGill University
Montreal, Canada

August 2016

A thesis submitted to McGill University in partial fulfilment of the requirements of the degree of Master of Arts.

Abstract

Perceptual studies have shown that signals with common amplitude- and frequency-modulation are heard as coming from the same source [30], [29]. This thesis examines the classification of partials via their measured modulations. To include these modulations in a sum-of-sinusoids model a superlinear polynomial phase function is adopted whose parameters are estimated using the Distribution Derivative Method (DDM) [1]. For better estimation accuracy, a window is designed that has lower side-lobes than the canonical Hann window but that is also once-differentiable — a requirement of the DDM. These estimated parameters are used in a new partial tracking algorithm based on linear programming. The resulting partials are classified using the clustering technique of Gaussian mixture models [11] on frequency- and amplitude-modulation data. Once the partials have been classified into sources, the sources are synthesized from the measured sinusoidal parameters. The additional information provided by the DDM is incorporated into interpolating polynomials for the phase and amplitude of sinusoids. The quality of different model-orders for these polynomials is assessed on synthetic signals. The system is evaluated on both simulated data and on a mixture of real recordings of musical instruments.

Contents

List of Figures

Chapter 1

Introduction

In signal processing, a common task is the separation of a signal with known deterministic or statistical characteristics from another. This task has been well studied [23] [18] [38] and works well for problems of digital communication or object detection where these characteristics are well known. In digital communication, the signals and techniques to transmit them are often optimized by the designer to make them robust to corruption or interference. The designers of vehicles usually design them to be predictable and reliable and so their positions in time will reflect this. In this thesis we tacle a more difficult problem, that of the separation of a mixture of acoustic signals. The nature of these signals is different in that their design criteria are either mostly unknown or fundamentally different. For example, musical instruments are designed to have desirable acoustic properties which are generally subjective. As an example of one of the complications, the choirs of the orchestra are sets of instruments actually designed to blend well; to sound as one instrument. Ironically, it is this criterion that we use to guide the source separation techniques described in the following.

1.1 Notation

1.1.1 Vectors and matricies

While scalars are typeset normally — x is an example of a scalar — vectors and matrices are typeset in a boldface font, with matrices written with a capital letter, e.g., $\boldsymbol{x}$ is a vector and $\boldsymbol{X}$ a matrix. If a number is written instead of a symbol, we mean a vector all of that

number, e.g., $\mathbf{1}$ is the vector of all 1s, $\mathbf{0}$ the vector of all 0s. The ith entry of a vector $\boldsymbol{x}$ is written x_i and the entry in the ith row and jth column of a matrix X is written $X_{i,j}$. Both are scalars and therefore typeset normally. Sometimes we might find it convenient to extract a column vector or row vector from the matrix $\boldsymbol{X}$. We write $\boldsymbol{x}_{i,:}$ to extract all columns from the ith row and and $\boldsymbol{x}_{:,j}$ to extract all rows from the jth column. These are the ith row vector and jth column vector respectively. The orientation of a vector will be clear from the context, but in general $\boldsymbol{x}$ is a column vector while $\boldsymbol{y}_{i,:}$ and $\boldsymbol{x}^T$ are row vectors.

1.1.2 Operators

Inner product

We will be dealing with objects in vector spaces. The operator $\langle x, y\rangle$ takes two objects in a vector space V, $x, y \in V$ and maps them to an element $k \in \mathbb{K}$ of a field $\mathbb{K}$. For this thesis, the field will always be the field of real numbers $\mathbb{R}$ or complex numbers $\mathbb{C}$. The vector space can be the set of vectors of N elements in $\mathbb{R}^N$ or $\mathbb{C}^N$, in which case the inner product is defined, for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{K}^N$, $k \in \mathbb{K}$

$$\langle \boldsymbol{x}, \boldsymbol{y}\rangle = \boldsymbol{x}^T\boldsymbol{y} = k$$

The inner product is also defined on the vector space of functions Φ mapping from a set S to a field $\mathbb{K}$, $\Phi : \forall f$ s.t. $f(s) = k, s \in S, k \in \mathbb{K}$ in which case the inner product on $g, f \in \Phi$ is defined as

$$\langle g, f\rangle = \int_{-\infty}^{\infty} g(x)\overline{f(x)}dx$$

and $\overline{a}$ gives the complex conjugate of a.

General outer operators

The outer operator $\cdot \otimes_{\mathcal{O}} \cdot$ will only be defined for vectors in this thesis. It operates on the two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{K}^N$ and is defined as

$$\boldsymbol{x} \otimes_{\mathcal{O}} \boldsymbol{y} = \boldsymbol{W}$$

where the ith row and jth column of $\boldsymbol{W}$ are

$$w_{i,j} = \mathcal{O}(x_i, y_j)$$

Canonically, the operator $\mathcal{O}$ is multiplication in which case

$$\boldsymbol{x} \otimes_{\times} \boldsymbol{y} = \boldsymbol{W}$$

where the ith row and jth column of W are

$$W_{i,j} = x_i y_j$$

but $\mathcal{O}$ can be defined arbitrarily as any function taking two inputs a returning a single output. The general outer product is also known as the *Kronecker product*.

Point-wise operators

If an operator on matrices $\circ$ is written with a period preceding it, i.e., $.\circ$ it means perform that operation on each element individually. Some examples follow.

For matrix $\boldsymbol{X} \in \mathbb{K}^{M,N}$ and $p \in \mathbb{K}$

$$\boldsymbol{X}^{\cdot p} = \boldsymbol{W}$$

where

$$W_{i,j} = X_{i,j}^{p}$$

For matricies $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{K}^{M,N}$

$$\boldsymbol{X}.\boldsymbol{Y} = \boldsymbol{W}$$

where

$$W_{i,j} = X_{i,j} Y_{i,j}$$

(constrast these with canonical matrix multiplication).

1.1.3 Random variables

Many authors denote random variables with a normally typeset uppercase letter. We will use this convention only when convenient, but will always state explicitly that a certain variable is random.

1.1.4 Complex numbers

A complex number $z \in \mathbb{C}$ can be described in cartesian notation as

$$z = a + jb, a, b \in \mathbb{R}$$

or in polar notation as

$$z = \alpha \exp(j\omega), \alpha, \omega \in \mathbb{R}$$

where $j = \sqrt{-1}$. j is also often used to denote an index variable. It will be clear from the context when the imaginary number is meant and when the index.

1.1.5 Logarithms

The logarithm base-e[1] of x is written $\log(x)$. The logarithm of any other base b will be denoted as such: $\log_b(x)$.

[1] e is Euler's constant.

Chapter 2

Methodology

2.1 Previous work on auditory source separation

As this thesis is primarily concerned with musical signals, it would not be relevant to give an overview of all audio source separation techniques presented in the literature. The following is a brief overview of those applicable to musical signals.

2.2 Multiple fundamental frequency estimation

This technique assumes the signal considered can be described in a format akin to the musical score — a collection of notes each with times indexing their beginning and end and a frequency, the fundamental, describing the perceived pitch of the note. Multiple fundamental frequency estimation for the purposes of music transcription dates back to the 1970s [17, ch. 20] [35]. This is related to audio source separation because the resulting high-level representation — the estimated score — can be used to synthesize signals corresponding to subsets of notes in the score, e.g., if a particular instrument is desired, its notes are extracted and then a signal is synthesized using stored recordings of the instrument or instrument-modeling synthesis techniques. The technique has become quite developed, see [17, ch. 20] for a review of modern techniques.

There are some drawbacks to the technique. Many musical signals of interest such as unpitched percussion, do not always have a perceiveable fundamental frequency. A musical score describes notes as having distinct boundaries in time and frequency, which

is not always true when one considers musical gestures such as portamento or *dal niente*[1]. Nevertheless, the production of even a crude score from a musical signals is useful in applications such as automated music transcription (e.g., [45]) or music catalogue query [32].

2.3 Principal latent component analysis (PLCA) and non-negative matrix factorizations (NMF)

The power spectrum of a signal and consequently its spectrogram are always non-negative valued. If a signal is considered as consisting of a sum of original source signals, these source signals will have non-negative spectrograms as well. The following two techniques attempt to determine these spectrograms solely from a spectrogram of their mixture.

2.3.1 PLCA

In this formulation, the spectrogram (defined in Section 3.2), being non-negative like a probability distribution, is considered as such

$$c|X(t,f)| = \mathrm{p}(t,f)$$

where c is a constant so that the distrubution integrate to 1. Explicitly, we consider the probability that energy in the spectrogram lie in the vicinity of time t and frequency f. With the hope of retrieving the spectrograms of the P underlying sources, it is proposed that the distribution is actually the distribution of K random variables each being chosen with probability $\mathrm{p}(Z=k)$. The pair of random variables from component k, T_k and F_k are assumed independent, i.e., $\mathrm{p}(t,f|Z=k) = \mathrm{p}(t|Z=k)\mathrm{p}(f|Z=k)$. Each random variable, it is hoped, describes a source ($K=P$) or a part of a source ($K>P$). In addition, each of these underlying distributions has marginal distributions $\mathrm{p}(t|Z=k)$ and $\mathrm{p}(f|Z=k)$. The marginal distributions and $\mathrm{p}(Z=k)$ can be estimated using the expectation maximization

[1] "Out of nothing": usually accompanying a crescendo and indicating that the player start from silence and gradually increase their playing dynamic.

algorithm with the following update rules for the lth iteration of the algorithm [48]

$$\mathrm{p}_{l+1}(Z=k|t,f)=\frac{\mathrm{p}_l(Z=k)\mathrm{p}_l(t|Z=k)\mathrm{p}_l(f|Z=k)}{\sum_{j=0}^{K-1}\mathrm{p}_l(Z=j)\mathrm{p}_l(t|Z=j)\mathrm{p}_l(f|Z=j)}$$

$$\mathrm{p}_{l+1}(t|Z=k)=\frac{\sum_f \mathrm{p}(t,f)\mathrm{p}_{l+1}(Z=k|t,f)}{\sum_s\sum_f \mathrm{p}(s,f)\mathrm{p}_{l+1}(Z=k|s,f)}$$

$$\mathrm{p}_{l+1}(f|Z=k)=\frac{\sum_t \mathrm{p}(t,f)\mathrm{p}_{l+1}(Z=k|t,f)}{\sum_t\sum_g \mathrm{p}(t,g)\mathrm{p}_{l+1}(Z=k|t,g)}$$

$$\mathrm{p}_{l+1}(Z=k)\frac{\sum_t\sum_f \mathrm{p}(t,f)\mathrm{p}_{l+1}(Z=k|t,f)}{\sum_{j=0}^{K-1}\sum_t\sum_f \mathrm{p}(t,f)\mathrm{p}_{l+1}(Z=j|t,f)}$$

After convergence, the marginal distribution $\mathrm{p}_{l^*}(t|Z=k)$ gives the distribution of energy of the kth component over time. Similarly, the marginal distribution $\mathrm{p}_{l^*}(f|Z=k)$ gives the distribution of energy of the kth component over frequency. Once the set of components $\{\tilde{k}\}$ belonging to the pth source has been determined, we can synthesize the spectrogram of this source as

$$|X_p(t,f)|=\frac{1}{c}\sum_{j\in\{\tilde{k}\}}\mathrm{p}_{l^*}(t,f|Z=j)\mathrm{p}_{l^*}(Z=j)$$

2.3.2 NMF

Instead of considering $|X(t,f)|$ as a probability distribution, we consider it at discrete frequencies mc_f and times nc_t with $m,n\in\mathbb{N}$, the entry at the mth row and nth column of matrix $V_{m,n}=|X(nc_t,mc_f)|$ with non-negative entries. We seek an approximate factorization of $\boldsymbol{V}\in\mathbb{R}_+^{M\times N}$ into matrices $\boldsymbol{W}\in\mathbb{R}_+^{M\times K}$ and $\boldsymbol{H}\in\mathbb{R}_+^{K\times N}$ such that

$$\boldsymbol{V}\approx\boldsymbol{W}\boldsymbol{H}$$

This can be done by solving the program

$$\min\mathcal{D}(\boldsymbol{V},\boldsymbol{W}\boldsymbol{H})$$

subject to

$$\boldsymbol{V}\geq\boldsymbol{0}$$

$$\boldsymbol{W} \geq \mathbf{0}$$

$$\boldsymbol{H} \geq \mathbf{0}$$

The particular choice of function (D) that measures divergence leads to different update equations in the iterative procedure for finding $\boldsymbol{W}$ and $\boldsymbol{H}$.

The Kullback-Leibler divergence [26]

The *Kullback-Leibler* divergence function for measuring the divergence between two matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$ is

$$\mathcal{D}_{\mathrm{KL}}(\boldsymbol{X}, \boldsymbol{Y}) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} X_{m,n} \log(Y_{m,n}) - X_{m,n} + Y_{m,n}$$

can be minimzed using the following update equations for the lth iteration

$$H_{a,b}^{l+1} = H_{a,b}^{l} \frac{\sum_{j=0}^{M-1} W_{j,a}^{l} V_{j,b}^{l} / (W^{l} H^{l})_{j,m}}{\sum_{j=0}^{M-1} W_{j,a}^{l}} W_{a,b}^{l+1} = W_{a,b}^{l} \frac{\sum_{j=0}^{N-1} H_{b,j}^{l+1} V_{a,j}^{l} / (W^{l} H^{l+1})_{a,j}}{\sum_{j=0}^{N-1} H_{b,j}^{l+1}}$$

It can be shown that these update equations are equivalent to those for PLCA [47].

The Itakura-Saito divergence [8]

Another divergence popular for factorizing spectrograms is the *Itakura-Saito* divergence

$$\mathcal{D}_{\mathrm{IS}}(\boldsymbol{X}, \boldsymbol{Y}) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \frac{X_{m,n}}{Y_{m,n}} - \log\left(\frac{X_{m,n}}{Y_{m,n}}\right) - 1$$

This divergence is scale-invariant, meaning that $\mathcal{D}_{\mathrm{IS}}(c\boldsymbol{X}, c\boldsymbol{Y}) = \mathcal{D}_{\mathrm{IS}}(\boldsymbol{X}, \boldsymbol{Y})$, which makes it well suited for audio signals which have a large dynamic range. Put another way, divergences involving large values in $\boldsymbol{V}$ and $\boldsymbol{WH}$ will be weighted similarly to divergences involving small values, which is not the case with the Kulback-Leibler divergence. The Itakura-Saito divergence is minimized through the following update equations

$$\boldsymbol{H}^{l+1} = \boldsymbol{H}^{l} . \frac{\boldsymbol{W}^{lT}((\boldsymbol{W}^{l}\boldsymbol{H}^{l})^{\cdot -2} . \boldsymbol{V}^{l})}{\boldsymbol{W}^{lT}(\boldsymbol{W}^{l}\boldsymbol{H}^{l})^{\cdot -1}}$$

$$\boldsymbol{W}^{l+1} = \boldsymbol{W}^{l} . \frac{((\boldsymbol{W}^{l}\boldsymbol{H}^{l+1})^{\cdot -2} . \boldsymbol{V}^{l})\boldsymbol{H}^{l+1T}}{(\boldsymbol{W}^{l}\boldsymbol{H}^{l+1})^{\cdot -1}\boldsymbol{H}^{l+1T}}$$

Once convergence has been obtained the kth column of matrix W will contain values representing the level of activation of the kth component at the frequency corresponding to the row index and the kth row of H the level of activation of the kth component at the time corresponding to the column index. If the set of components $\{\tilde{k}\}$ belonging to the pth source has been determined, we can synthesize the spectrogram of this source as

$$|X_p(nc_t, mc_f)| = \sum_{j \in \{\tilde{k}\}} W_{,j} H_{j,}$$

2.3.3 Synthesis of factorized spectrograms

Synthesizing the original signal is less straightforward as the phase information contained in the STFT was discarded to obtain a non-negative spectrogram. We can simply use the original phases of the STFT used to compute the spectrum with the new magnitude information from $|X_p(t, f)|$ but the resulting signal may have artifacts due to the phase information of unwanted sources that remains in the STFT. A technique to lessen these artifacts using constrained Wiener filtering has been proposed in [25]. One may also choose to invert the spectrogram without any phase information by using an algorithm that iteratively reconstructs the phase part of the STFT while minimizing the error between the spectrogram of the reconstructed signal and its original power spectrum, starting from an initial guess [14]. Each iteration requires transforming a the signal to its spectrogram and then back to a time-domain signal, requiring considerable computational effort.

2.4 An approach using amplitude- and frequency-modulation

Perceptual studies have shown that sounds modulated synchronously in amplitude or frequency are heard as one sound, whereas asynchronously modulated sounds are heard as distinct [30] [29]. Here we define the modulation of parameters θ_i and θ_j as being synchronous if they are given as functions of time, $\theta_i = f_i(t)$ and $\theta_j = f_j(t)$ and there is an affine transform $\mathscr{A}$ such that $\mathscr{A}\{f_i\}(t) \approx Af_j(t) + B$ where A and B are constants that do not vary with time (at least for the time that we observe the signal). If we can accurately measure these parameters and they are typical of the sounds we are trying to separate,

then we can design techniques to reliably separate these sounds from acoustic mixtures. This involves picking those parameters classified as belonging to the same sound, discarding the rest, and resynthesizing from these parameters. The task of audio source separation therefore comprises the following tasks:

- Decide on a signal model for the sound of interest, with parameters that can be estimated and that are similar for similar sounds.
- Estimate the signal parameters.
- Classify the parameters and group them as sets of parameters coming from the same source: the sound of interest.
- Choose a group of parameters and synthesize the separated signal from them.

In this thesis we model the signals of interest as sums of sinsuoids with polynomial phase. This model is chosen because the derivatives of this polynomial are readily interpreted as the amplitude- and frequency-modulation functions of the sinusoids and techniques exist for estimating the polynomial coefficients from realizations of these signals. The polynomial phase sinusoid is defined in Section 3.3.

Chapter 3

Signal Modeling

3.1 Introduction

To build tools to analyse and synthesize signals some structure must be imposed on the signals. The structures chosen can reflect something about the behaviour of these signals as observed in the field, as we will see with sinusoidal models. Other structures are chosen phenomenalogically — we do not really know the underlying mechanism behind the production of these signals, but a particular structure is chosen for its mathematical or conceptual convenience, such as is the case when we consider higher order models for sinusoidal phase and amplitude.

We begin the chapter with what could be seen as a mathematical analog of the musical score: time-frequency representations. Through this we will justify the adoption of a sinusoidal model for musical signals. Finding this inadequate to describe the signals of interest with sufficient quality, the sinusoidal model is generalized to incorporate modulations in frequency and amplitude. A technique is described to estimate the parameters of these more complex models which requires windows that are everywhere differentiable — we design a new window having desirable properties close to those of well-known optimal windows, but that is everywhere differentiable.

3.2 Time-Frequency Representations

As most musical instruments are resonating media and excited resonating media are well described as linear time-invariant (LTI) auto-regressive (AR) structures, popular models

of musical audio are some variation of this description [9]. Strictly speaking, incorporating moving-average (MA) structures into a model of musical signals can improve its quality, but is more difficultly cast in the sum-of-sinusoids model adopted later in this thesis.

An LTI auto-regressive structure is a signal that can be desribed using the following *difference equation*:

$$x(n) = \sum_{k=1}^{K} a_k x(n-k) + b_0 v(n) \tag{3.1}$$

Here x is the output of the system (what is heard or measured) and v is the input. K is the order of the model. Both are general functions of time which, in the case of properly sampled digital audio, can be considered at discrete times $n \in \mathbb{Z}$ without any loss of information [5, ch. 2]. $a_k, b_0 \in \mathbb{C}$ and are constants. Casually[1] you can think of the output of the system at time n as being a linear combination of past outputs, plus some of the scaled input.

AR structures are excited in various ways: some are bowed, others struck, etc. To characterize the above structure we excite it with a simple signal, the *Kronecker delta*

$$\delta(n) = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

This Kronecker delta input will yield its *impulse response* from which we can derive many properties of the AR structure.

As an example take the case where $K = 1$ and $a_1 = r\exp(j\omega)$, r,$\omega \in \mathbb{R}$, $|r| < 1$. Then the difference equation is

$$x(n) = r\exp(j\omega)x(n-1) + v(n) \tag{3.3}$$

Exciting this with the Kronecker delta we get

$$\begin{aligned} x(0) &= 1 \\ x(1) &= r\exp(j\omega) \\ x(n) &= r^n \exp(j\omega n) \end{aligned} \tag{3.4}$$

[1] No pun intended.

which is a complex exponential starting at $n = 0$ and periodic in $n_T = \frac{2\pi}{\omega}$ multiplied by the real-valued exponential r^n. In other words, the output is a damped sinusoid. From this it is not hard to see that if we can estimate the coefficients a_k, we can then know the frequencies, amplitudes and damping factors of the sinusoids that are output when this structure is excited by an impulse (the Kronecker delta). This principle is presented as a motivation for the following techniques and is not pursued here. The interested reader is referred to [28] for more information.

An alternative method for determining the frequencies and amplitudes of sinusoids in mixture is to take the inner product of the signal with a complex exponential of known frequency and see what you get

$$X(\omega) = \sum_{n=-\infty}^{\infty} x(n) \exp(-j\omega n) \tag{3.5}$$

The function $X(\omega)$ will be large if $x(n)$ contains a complex exponential of frequency ω and small if it doesn't, effectively indicating which sinusoidal functions are present in the signal. This transformation of a signal as a function of time n into one as a function of frequency ω is known as the *Discrete-time Fourier Transform* (DTFT).

To create a variety of pitches and timbres, typically the media of musical instruments are not static, but vary in time. That means the sets of sinusoids describing the state of the media and its excitation also change in time. To account for this we consider many small intervals of signal where we assume its characteristics are roughly static. We can then piece these time-intervals together afterwards to get a description of the signal in both time and frequency. To do this, we multiply the signal by a window w which makes the signal 0 outside of the interval of interest. We then test what sinusoids with frequencies ω are present at different times τ, giving a function of two variables

$$X(\tau, \omega) = \sum_{n=-\infty}^{\infty} x(n) w(n - \tau) \exp(-j\omega n) \tag{3.6}$$

This transformation of a signal of time n into one of time τ and frequency ω is known as the *Discrete-time Short-time Fourier Transform* (DTSTFT).

One further point about the window should be discussed. The Fourier transform of the product of two functions, like we have in Equation 3.6, is equal to the convolution of the Fourier transform of each function separately. If we denote the Fourier transform operator

as $\mathcal{F}$, for functions g and f we have

$$\mathcal{F}(gf) = \mathcal{F}(g) * \mathcal{F}(f)$$

where $*$ is the convolution operator. The value $X(\tau, \omega)$, which will be a complex number, can be seen as describing the amplitude and phase of a sinusoid at that frequency and time. If the Fourier transform of the window function is not purely real its imaginary part will offset the phase of this sinusoid. It is usually simpler to avoid this complication. The Fourier transform of a real even function

$$f(n) = f(-n), f(n) \in \mathbb{R} \Rightarrow f \text{ is real and even.}$$

is real, so we choose windows with this property. See [16, p. 52] for a concrete illustration of this.

3.3 Polynomial Phase Models

The DTFT and DTSTFT are very useful because they are invertible [39] and fast algorithms exist for their computation by digital computer [52]. If the presence of a sinusoid is determined, e.g., by finding τ^* and ω^* such that X is maximized, that signal can be removed or altered easily.

One drawback of these transforms is they only project onto sinusoidal functions of linear phase, i.e., functions of constant frequency. In general, musical signals are not linear combinations of sinusoids of constant frequency (consider, for example, vibrato). We could decide to project onto a different family of functions and considerable effort has been devoted to finding alternatives (see [24] for a review). In the case of musical signals, however, we have some prior information about the mechanics of the their production and can make certain assumptions about the underlying functions.

3.3.1 Sinusoidal Representations

Many musical acoustic signals are quasi-harmonic [9], meaning that they consist of a sum of sinusoids whose frequencies are roughly integer multiples of a fundamental frequency. For these kinds of signals, most of the energy can be attributed to sinusoids and so, if

we neglect the noisy part of the signal, the signal can be described by a small number of sinusoids with slowly varying amplitude and phase, plus some noise. The model is

$$x(n) = \sum_{p=1}^{P} A_p(n) \exp(j\phi_p(n)) + \varepsilon \tag{3.7}$$

where $\varepsilon \sim \mathcal{N}(0, \sigma)$, σ^2 quantifies the power of the noise, and $A_p(n), \phi_p(n) \in \mathbb{R}$ are functions of amplitude and phase respectively. In the following, we consider equivalent sinusoidal mixtures of complex-valued polynomial phase exponentials

$$x(n) = \sum_{p=1}^{P} \exp(\mathcal{P}_p(n)) + \varepsilon \tag{3.8}$$

where

$$\mathcal{P}_p(n) = \sum_{q=0}^{Q} c_q n^q \tag{3.9}$$

and $c_q \in \mathbb{C}$. Note the form in Equation 3.7 can be retrieved as

$$\mathcal{P}_p(n) = \log(A_p(n)) + j\phi_p(n) = \Re\{\mathcal{P}_p(n)\} + j\Im\{\mathcal{P}_p(n)\}$$

In Chapter 5 we will see how estimations of these parameters at different times can be interpolated to create higher order phase functions with improved synthesis quality.

3.4 Polynomial phase parameter estimation

More recently a set of techniques have been developed that use some combination of derivatives of the analysis window w or the signal x to estimate the polynomial coefficients directly [15]. For this thesis we will only consider a technique that does not estimate derivatives of the signal and only requires a once-differentiable analysis window as it is relatively easy to implement and suits our purposes.

The following is adapted from [1]. Consider the inner product of the signal $x(n) = \exp(\mathcal{P}_p(n))$ and a known analysis *atom* $\psi(n)$

$$\langle x, \psi \rangle = \int_{-\infty}^{\infty} x(n)\overline{\psi}(n)dn$$

Differentiating with respect to n, we obtain by the product rule

$$\frac{dx}{dn}(n)\overline{\psi}(n) + x(n)\frac{d\overline{\psi}}{dn}(n) = \left(\sum_{q=1}^{Q} qc_q n^{q-1}\right) x(n)\overline{\psi}(n) + x(n)\frac{d\overline{\psi}}{dn}(n)$$

If $\psi(t)$ is 0 outside of some interval $n \in [-T, T]$ then

$$\sum_{q=1}^{Q} qc_q \int_{-T}^{T} n^{q-1} x(n)\overline{\psi}(n)dn + \left\langle x, \frac{d\overline{\psi}}{dn} \right\rangle = 0$$

If we define the operator $\mathcal{T}^\alpha : (\mathcal{T}^\alpha x)(n) = n^\alpha x(n)$ we can write

$$\sum_{q=1}^{Q} qc_q \left\langle \mathcal{T}^{q-1}x, \overline{\psi} \right\rangle = -\left\langle x, \frac{d\overline{\psi}}{dn} \right\rangle$$

From this we can see that to estimate the coefficients c_q, $1 \leq q \leq Q$ we simply need R atoms with $R \geq Q$ to solve the linear system of equations

$$\sum_{q=1}^{Q} qc_q \left\langle \mathcal{T}^{q-1}x, \overline{\psi}_r \right\rangle = -\left\langle x, \frac{d\overline{\psi}_r}{dn} \right\rangle \tag{3.10}$$

for $1 \leq r \leq R$. To estimate c_0 we write the signal we are analysing as

$$s(n) = \exp(c_0) \exp\left(\sum_{q=1}^{Q} c_q n^q\right) + \eta(n)$$

$\eta(n)$ is the error signal, or the part of the signal that is not explained by our model. We also define a function $\gamma(n)$, the part of the signal whose coefficients have already been estimated

$$\gamma(n) = \exp\left(\sum_{q=1}^{Q} c_q n^q\right)$$

Computing the inner product $\langle s, \gamma \rangle$, we have

$$\langle s, \gamma \rangle = \langle \exp(c_0)\gamma, \gamma \rangle + \langle \eta, \gamma \rangle$$

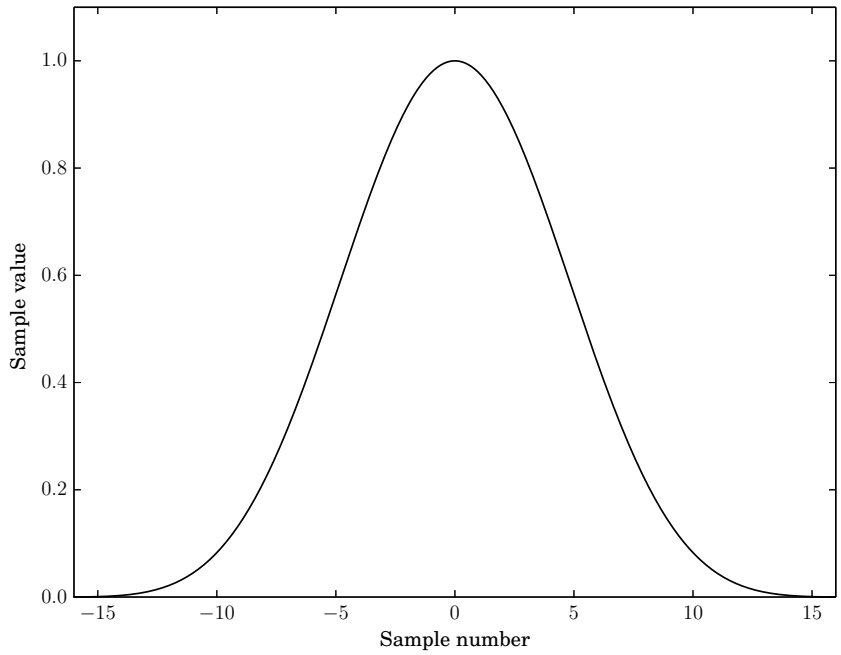

Fig. 3.1: **Minimum 4-Term Blackman-Harris: Time Domain**

The inner-product between η and γ is 0, by the orthogonality principle [23, ch. 12]. Furthermore, because $\exp(c_0)$ does not depend on n, we have

$$\langle s, \gamma \rangle = \exp(c_0) \langle \gamma, \gamma \rangle$$

so we can estimate c_0 as

$$c_0 = \log\left(\langle s, \gamma \rangle\right) - \log\left(\langle \gamma, \gamma \rangle\right) \tag{3.11}$$

The estimation of the coefficients of a phase polynomial using this method is known as the *Distribution Derivative Method (DDM)*.

3.5 Choosing atom ψ

As we are dealing with mixtures of sinusoids of small bandwidth, in addition to the finite time support constraint, we desire atoms whose inner-product is only significant within a finite bandwidth of interest. To construct these atoms, we multiply the Fourier atom by

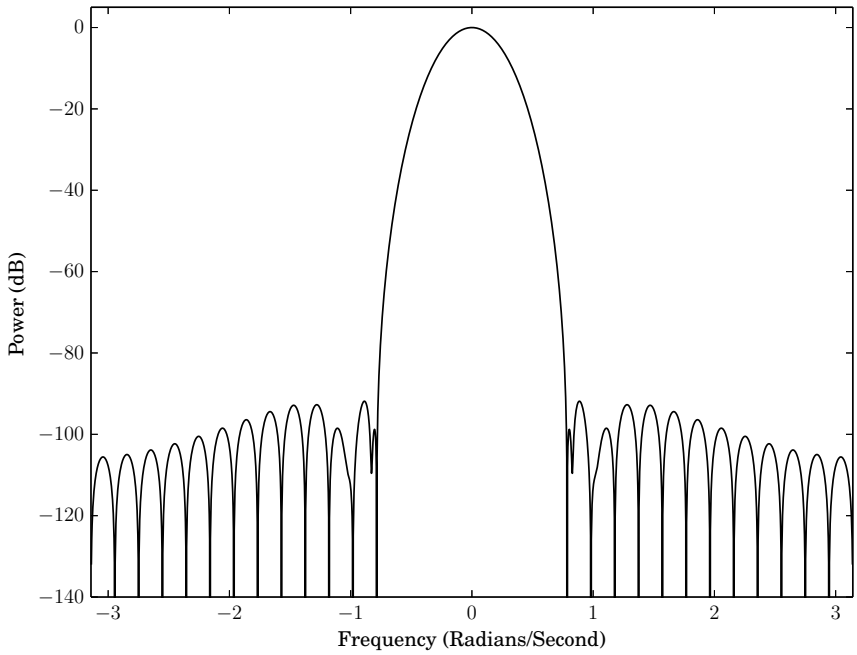

Fig. 3.2: **Minimum 4-Term Blackman-Harris: Frequency Domain**

the window w

$$\psi_{\tau,\omega}^{\mathcal{F}_w}(n) = w(n-\tau)\exp(-j\omega(n-\tau))$$

A good overview of different windows and their properties is given in [16]. We require that the window be at least once-differentiable and zero outside of a certain interval, therefore, somewhat informally, we require

$$\lim_{n\to T}\psi(n) = \psi(T) = 0$$

Table 3.1

Window	a_0	a_1	a_2	a_3
Minimum	0.35857	0.48829	0.14128	0.01168
$\mathcal{C}^1$	0.35874	0.48831	0.14127	0.01170

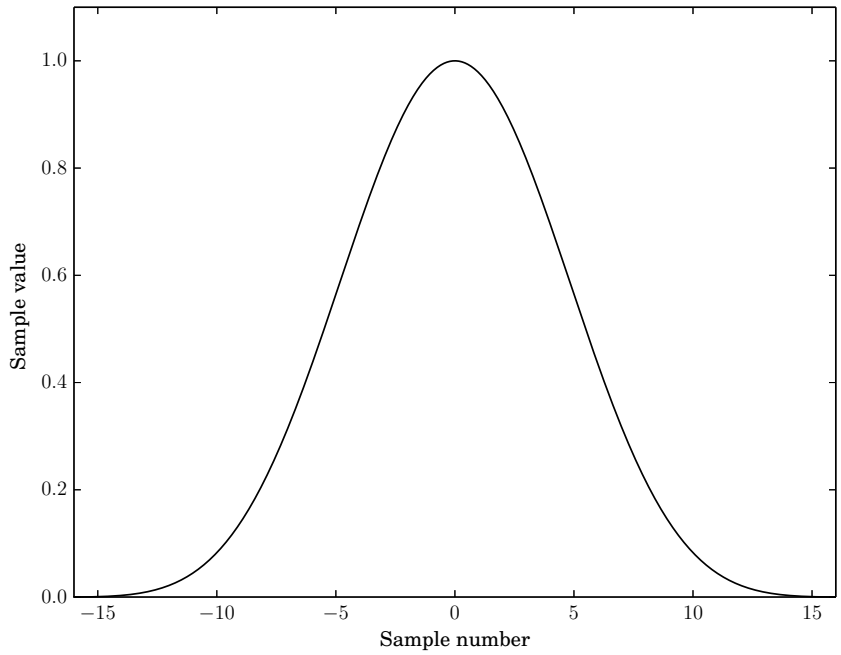

Fig. 3.3: $\mathcal{C}^1$ **4-Term Blackman-Harris: Time Domain**

3.5.1 The Hann window

The *Hann* window possesses this property

$$w_h(n) = \begin{cases} 0.5 + 0.5\cos\left(\frac{n}{T}\pi\right) & -T \leq n \leq T \\ 0 & \text{otherwise} \end{cases}$$

The Hann window is a member of a class of windows constructed by summing scaled harmonically related cosine functions, subject to the constraint that the scaling coefficients

Table 3.2

Window	Highest side-lobe level (dB)	6-dB bandwidth in bins
Minimum	-92	2.72
$\mathcal{C}^1$	-90	2.66

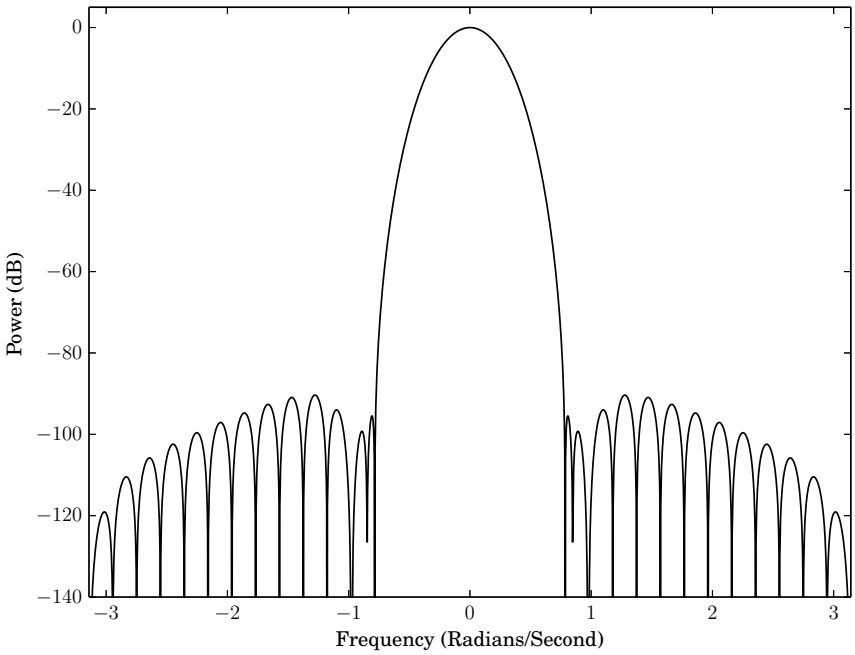

Fig. 3.4: $\mathcal{C}^1$ **4-Term Blackman-Harris: Frequency Domain**

sum to 1 so that the window have a value of 1 at $n = 0$. Letting $T = N/2$, where N is the length of the window

$$w(n) = \begin{cases} \sum_{m=0}^{M-1} a_m \cos\left(\frac{2\pi}{N} mn\right) & -\frac{N}{2} \leq n \leq \frac{N}{2} \\ 0 & \text{otherwise} \end{cases}$$

With $M = 2$ and $a_0 = a_1 = 0.5$, we have the Hann window.

The simple expression for its calculation and good trade-off between main-lobe width and side-lobe height make the Hann window a popular choice in many signal processing applications. The expression for its Fourier transform is such that fast digital implementations of windowing a signal by a Hann window involve no multiplies [16, p. 183]. A recursive implementation of the DTSTFT is possible when windowing with the Hann window, which is important for applications where little storage is available [51, p. 102]. In spite of all its merits, other windows have been proposed that have superior qualities, such as lower side-lobes.

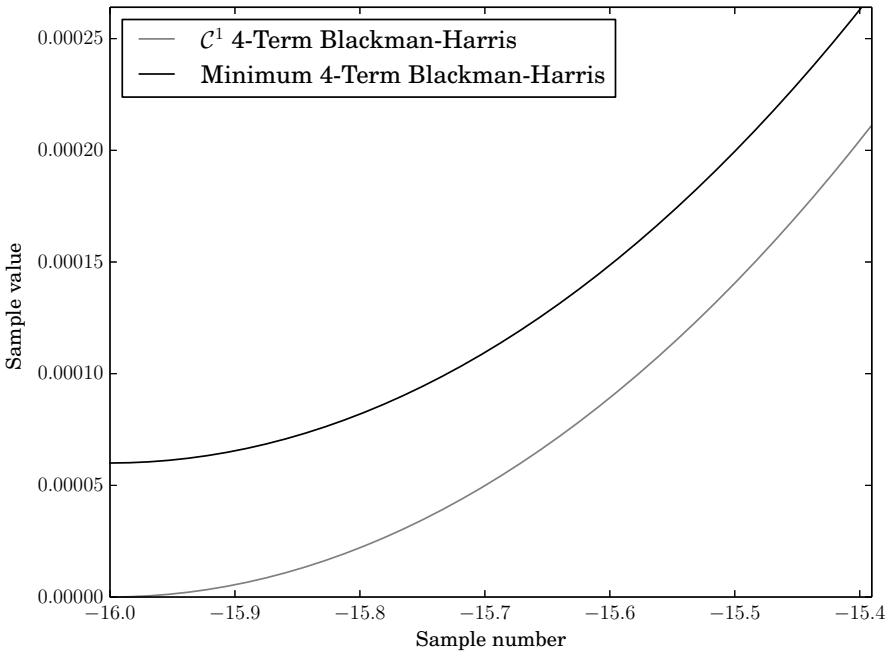

Fig. 3.5: **Comparison of endpoints of window in time-domain**

3.5.2 Continuous Blackman-Harris windows

A family of windows with certain properties superior to the Hann window is the *Blackman-Harris* family of windows. These are also sum-of-cosine windows. For these windows, optimization techniques were used to search for coefficients giving minimum height of the highest side-lobe (maximum out-of-band rejection) [41]. The 4-term window whose coefficients a are listed in Table 3.1 has a maximum side-lobe level of 92 dB, lower than the quantization noise of a 16-bit linear pulse code modulated signal. This window has a very large main-lobe which means two sinusoids of similar frequency will be difficult to resolve. Furthermore, the window has a discontinuity at its boundaries, e.g., $w\left(\frac{N}{2}\right) \neq 0$ and is not once-differentiable. In any case the window is valuble in that it effectively nulls any influence of signals outside of a bandwidth of interest.

It should be clarified that when we compare the widths of the main-lobes of two windows, we compare two windows of the same length. Of course, the bandwidth of a window can also be decreased by increasing its length, at the expense of time-resolution. When searching

for windows superior to the Hann window, we are motivated by our ability to describe the signal with more detail between two analysis frames than would be possible with a simple linear-phase sinusoid model. Using a longer window to decrease the main-lobe width is not problematic in this case, but we would still like a high level of signal rejection outside of the bandwidth of interest for improved estimation accuracy of the signal parameters. For this reason, we search for windows that have very low side-lobe height and are also once-differentiable, without caring so much about the width of the main-lobe. To find a window with properties similar to the 4-term Blackman-Harris window but without a discontinuity, we solve the optimization problem

$$\min ||a - \tilde{a}||_2$$

subject to

$$w_{\tilde{a}}\left(\frac{N}{2}\right) = w_{\tilde{a}}\left(\frac{-N}{2}\right) = 0$$

$$\sum_{m}^{M-1} a_m = 1$$

where

$$w_{\tilde{a}}(n) = \begin{cases} \sum_{m=0}^{M-1} \tilde{a}_m \cos\left(\frac{2\pi}{N} mn\right) & -\frac{N}{2} \leq n \leq \frac{N}{2} \\ 0 & \text{otherwise} \end{cases}$$

The solution $\tilde{a}^*$ is given in Table 3.1 and plots are given in Figure 3.1. This window will be referred to as the $\mathcal{C}^1$ 4-Term Blackman-Harris window. Some figures of merit for the two windows are compared in Table 3.2 in a similar fashion to [16]. We see that the $\mathcal{C}^1$ 4-Term Blackman-Harris window is not too different from the Minimum 4-term Blackman-Harris window, but has the additional desirable property of differentiability everywhere in its domain. A comparison of the windows's endpoints is presented in Figure 3.5.

3.6 Conclusion

In this chapter we have developed the rationale for adopting the sinusoidal model when analysing musical signals. In turn we have presented some techniques for estimating the parameters of these signals. Obviously these techniques only work as well as their assump-

tions are true — for the best results we should use these techniques only on signals that indeed contain sinusoids. For the signals considered in this thesis, we assume this to be true.

The techniques presented in this chapter are typically used to estimate the parameters of short signals as we see in the use of window functions to limit the time-frequency extent of our analysis. For the source separation problem we are interested in larger sinusoidal objects — partials — whose global properties are more readily classified. We could use a very large analysis window and a high order polynomial for phase when solving for the coefficients, but the size of the linear system to be solved in 3.10 will increase quadratically in the order of the model. Furthermore, it is difficult to account for situations where the signal is corrupted or briefly absent. In these situations we may prefer to use interpolation to reconstruct the signal in the corrupted region. For these reasons, we prefer to make multiple estimations of the parameters of low order models and connect those estimations thought of as belonging to a single partial. We will see in Chapter 5 that this will allow postulating a higher order phase model. Before that is possible, however, we must determine how to connect multiple estimations to form partials.

Chapter 4

Partial Tracking

In the previous chapter, we saw how to estimate parameters of sinusoids with polynomial phase. While theoretically applicable to signals of arbitrary length, for reasons of flexibility and efficiency, we usually estimate the local parameters of the signal under a low order model and connect multiple estimations to form a partial. We will call these local estimations "analysis points" or "parameter sets".

This chapter presents an interpretation of the *peak matching* procedure of McAulay and Quatieri[31], a classical approach to discovering partials. Our interpretation allows for the specification of an arbitrary cost function measuring the plausibility that a set of analysis points form the path of a partial. With this path interpretation, we present a technique that finds the optimal set of paths under a constraint on the number of paths. The chapter concludes with an example of partial tracking on a synthetic signal.

Typically the DTSTFT is computed for a block of contiguous samples, called a *frame* and these frames are computed every H samples, H being the *hop-size*. We will denote the M sets of parameters at local maxima in frame h as $\theta_0^h, \ldots, \theta_{M-1}^h$ and the N in frame $h+1$ as $\theta_0^{h+1}, \ldots, \theta_{N-1}^{h+1}$ where h and $h+1$ refer to adjacent frames. We are interested in paths that extend across K frames where each path touches only one parameter set and each parameter set is either exclusive to a single path or is not on a path.

4.1 A Greedy Method

In this section, we present the McAulay-Quatieri method of peak matching. It is conceptually simple and a set of short paths can be computed quickly, but it can be sensitive to

spurious peaks and is optimal only in the sense that the set of paths computed contains the best path possible — the quality of the other paths may be compromised under this criterion.

In [31, p. 748] the peak matching algorithm is described in a number of steps; we summarize them here in a way comparable with the linear programming formulation to be presented in the sequel. In that paper, the parameters are the instantaneous amplitude, phase and frequency and are indexed by frequency as $\omega_0^h, \ldots, \omega_{M-1}^h$ and $\omega_0^{h+1}, \ldots, \omega_{N-1}^{h+1}$ but we will allow for arbitrary parameter sets. Define a distance function $\mathcal{D}(\theta_i, \theta_j)$ that computes the similarity between $K = 2$ sets of parameters. We will now consider a method that finds L pairs of parameters that are closest.

We compute the cost matrix $\boldsymbol{C}$

$$\boldsymbol{C} = \theta^h \otimes_{\mathcal{D}} \theta^{h+1}$$

so that the ith row and jth column contain $C_{i,j} = \mathcal{D}\left(\theta_i^h, \theta_j^{h+1}\right)$. For each $l \in [0 \ldots L-1]$, find the indices i_l and j_l corresponding to the shortest distance, then remove the i_lth row and j_lth column from consideration and continue until L pairs have been determined or the distances exceed some threshold Δ. This is summarized in Algorithm 1

Input: the cost matrix $\boldsymbol{C}$
Output: L index pairs Γ_i and Γ_j
$\Gamma_i \leftarrow \varnothing$;
$\Gamma_j \leftarrow \varnothing$;
for $l \leftarrow 0$ *to* $L-1$ **do**
  $i_l, j_l = \underset{i \in [0,\ldots,M-1] \setminus \Gamma_i, j \in [0,\ldots,M-1] \setminus \Gamma_j}{\arg\min} C_{i,j}$;
  if $C_{i_l,j_l} > \Delta$ **then**
    return Γ_i, Γ_j
  end
  $\Gamma_i \leftarrow \Gamma i \cup i_l$;
  $\Gamma_j \leftarrow \Gamma i \cup j_l$;
end
return Γ_i, Γ_j

Algorithm 1:

This is a greedy algorithm because on every iteration the smallest cost is identified and

its indices are removed from consideration. Perhaps choosing a slightly higher cost in one iteration would allow smaller costs to be chosen in successive iterations. This algorithm does not allow for that. In other terms, the algorithm does not find a set of pairs that represent a globally minimal sum of costs. Another drawback of the algorithm is that it only works between two sucessive frames. The cost function could be extended to consider K frames (K arbitrary) of parameter sets, constructing a K-dimensional tensor instead of a matrix, but assuming equal numbers of parameter sets in all frames, the search space would grow exponentially with K. Nevertheless, the method is simple to implement, computationally negligble when K is small, and works well with a variety of signals encountered in audio [31] [50].

4.2 An Optimal Method

There is a way to find a set of paths over multiple frames ($K > 2$) having the lowest total cost if we restrict the search to exactly L paths. Instead of indexing parameters by their frame number h, we make h part of the parameter set so that it can be used by the distance function $\mathcal{D}$. Assume that over K frames there are M total parameter sets. In this context we will consider them as nodes in a graph. We define the vector $\boldsymbol{c} \in \mathbb{R}^{M^2}$ where the entry $\boldsymbol{c}_{i+Mj} = \mathcal{D}(\theta_i, \theta_j)$. If we have a set of connections $\Gamma_{i,j}$ we can calculate the total cost of these connections by defining the vector

$$\boldsymbol{x}_{i+Mj} = \begin{cases} 1 & \text{there is a connection between } i \text{ and } j \\ 0 & \text{otherwise} \end{cases}$$

and then forming the inner product

$$c_{\text{total}} = \langle \boldsymbol{c}, \boldsymbol{x} \rangle$$

Note that a node cannot be connected to itself. The question is how to find $\boldsymbol{x}^*$ so that c_{total} is minimized. If no constraints are placed on $\boldsymbol{x}$, the solution is trivial, but not useful. How do we constrain $\boldsymbol{x}$ to give us a solution to the partial tracking problem? Let us consider an example.

In Figure 4.1 we have an example of a simple graph or lattice. Such a graph represents

a plausible partial tracking situation: vertically aligned nodes are parameter sets estimated from the same analysis frame and we would like to connect these parameter sets between frames. The numbers are indices of nodes in the graph and the possible connections between them are indicated by lines, or *edges*. Imagine that we would like to find the two shortest paths. We will now examine the resulting paths from two algorithms using different criteria for shortness.

In Figure 4.3 we find the paths using an algorithm similar to Algorithm 1 but search instead over a tensor of distances $C \in \mathbb{R}^{3\times4\times2}$ whose entry $C_{i,j,h}$ represents the cost of travelling on the path connecting the ith node in layer 0, the jth node in layer 1 and the hth node in layer 2. This cost is the sum of the euclidean distances giving the lengths of the connections. This is the greedy method of searching for the best paths whose optimality criterion is to find the set of best paths containing the absolute best path. We see in Figure 4.3 that the absolute shortest path, $1 \to 4 \to 8$, is discovered, followed by the second shortest path not using the nodes of the first path, $2 \to 5 \to 7$.

4.2.1 L shortest paths via linear programming

To find a set of paths minimizing the total cost, we instead search for total solutions $\boldsymbol{x}$ that describe all paths in the graph. Assume for now that we can guarantee that the entries of $\boldsymbol{x}$ will be either 0 or 1. To find a set of constraints for our search, we consider the structure of a valid solution $\boldsymbol{x}^*$. To maintain that paths not overlap, a valid solution's nodes are only allowed to have one edge entering — coming from a node in a previous frame — and one edge leaving — going to a node in a successive frame. To translate this into a constraint, consider the node i and its possible R_i successive connecting nodes $j_0 \dots j_{R_i-1}$. Define the vector[1]

$$a^{\mathrm{s},i}_{i+Mj_r} = \begin{cases} 1 & \forall j_r \in [j_0 \dots j_{R_i-1}] \\ 0 & \text{otherwise} \end{cases}$$

As all the entries of $\boldsymbol{x}$ are either 0 or 1, we have

$$0 \leq \left\langle \boldsymbol{a}^{\mathrm{s},i}, \boldsymbol{x} \right\rangle \leq 1$$

[1]The superscript s stands for "successive".

Fig. 4.1

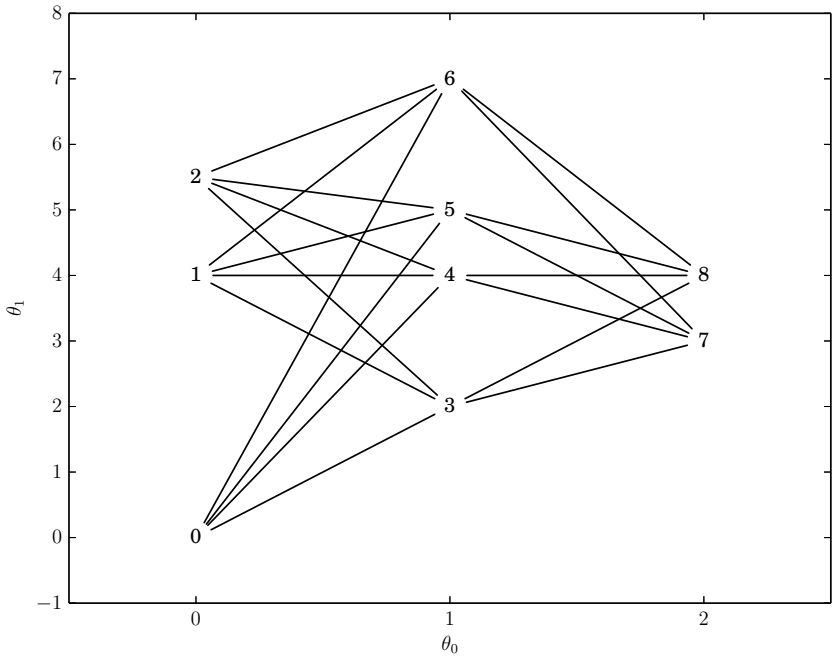

Fig. 4.2: **Possible graph connections**

so we can make this a constraint to ensure that a node has at most one path leaving. Similarly, if we consider the node j and its possible R_j previous connecting nodes $i_0 \dots i_{R_j-1}$, the vector[2]

$$a^{\mathrm{p},j}_{i_r+Mj} \begin{cases} 1 & \forall i_r \in \left[i_0 \dots i_{R_j-1}\right] \\ 0 & \text{otherwise} \end{cases}$$

constrains that node j have only one path entering through the constraint

$$0 \leq \left\langle \boldsymbol{a}^{\mathrm{p},j}, \boldsymbol{x} \right\rangle \leq 1$$

A node on a path will also have an edge entering and an edge leaving. To translate this into a constraint, we define a vector that counts the number of edges entering a node and subtracts then the number of edges leaving a node. The result should always be 0 for an

[2]The superscript p stands for "previous".

Fig. 4.3

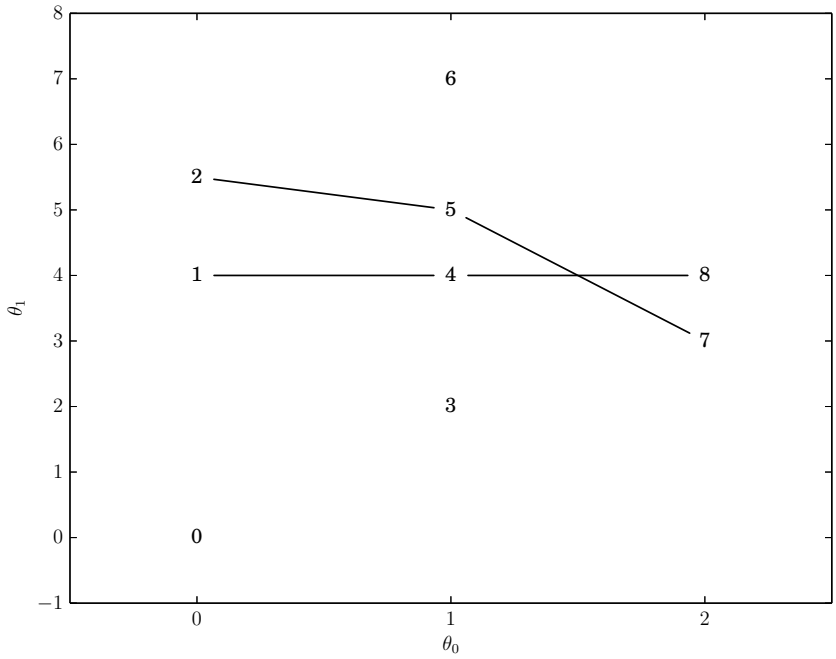

Fig. 4.4: **Two shortest paths using the greedy method**

equal number of edges entering and exiting a node. If r is the index of the node considered, the vector is simply [3]

$$\boldsymbol{a}^{\mathrm{b},r} = \boldsymbol{a}^{\mathrm{p},r} - \boldsymbol{a}^{\mathrm{s},r}$$

and the constraint

$$\left\langle \boldsymbol{a}^{\mathrm{b},r}, \boldsymbol{x} \right\rangle = 0$$

Finally we want to constrain that there be only L paths. We do this by noticing that if this is true, there will be L edges between frames h and $h+1$. We constrain the number of paths going from edges Γ_h in frame h to Γ_{h+1} by forming the vector [4]

$$\boldsymbol{a}^{\mathrm{c},h} = \sum_{j \in \Gamma_h} \boldsymbol{a}^{\mathrm{s},j}$$

[3] The superscript b stands for "balanced".
[4] The superscript c stands for "connections".

and asserting the constraint

$$\left\langle \boldsymbol{a}^{\mathrm{c},h}, \boldsymbol{x} \right\rangle = L$$

The length of $\boldsymbol{x}$ is M^2 so the total size of all the constraints is not insignificant, but most entries in the constraint vectors will be 0 and therefore the resulting constraint matrices very sparse, so sparse linear algebra routines can be used in computations. Furthermore, the $\boldsymbol{a}^b$ and $\boldsymbol{a}^c$ constraints are derived from $\boldsymbol{a}^p$ and $\boldsymbol{a}^s$, so only the latter need to be stored.

The complete *linear program (LP)* solving the L shortest paths problem is then

$$\min_{\boldsymbol{x}} \langle \boldsymbol{c}, \boldsymbol{x} \rangle$$

subject to

$$\mathbf{0} \leq \begin{bmatrix} \boldsymbol{A}_{\mathrm{s}} \\ \boldsymbol{A}_{\mathrm{p}} \end{bmatrix} \boldsymbol{x} \leq \mathbf{1}$$

$$\begin{bmatrix} \boldsymbol{A}_{\mathrm{b}} \\ \boldsymbol{A}_{\mathrm{c}} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \mathbf{0} \\ L\mathbf{1} \end{bmatrix}$$

$$\mathbf{0} \leq \boldsymbol{x} \leq \mathbf{1}$$

where $\boldsymbol{A}_{\mathrm{s}}$ is the matrix with $\boldsymbol{a}^{\mathrm{s},m}$ as its rows for $m \in [0 \ldots M-1]$ and $\boldsymbol{A}_{\mathrm{p}}$ is the matrix with $\boldsymbol{a}^{\mathrm{p},m}$ as its rows, etc.

The solution of the two best paths using the LP formulation is shown in Figure 4.5 and a comparison of the total costs is shown in Table 4.1

Table 4.1. Comparison of total costs

Greedy	LP
5.354102	4.946461

The LP formulation is inspired by a multiple object tracking algorithm for video [20]. A proof that the solution $\boldsymbol{x}^*$ will have entries equal to either 0 or 1 can be found in [37, p. 167]. The theoretical computational complexity of the linear program is polynomial in the number of variables, see [22] for a proof and the demonstration of a fast algorithm for finding its solution. In practice, to extract paths from the solution, we do not test equality with 0 or 1 but rather test if the solution vector's values are greather than some threshold.

Fig. 4.5

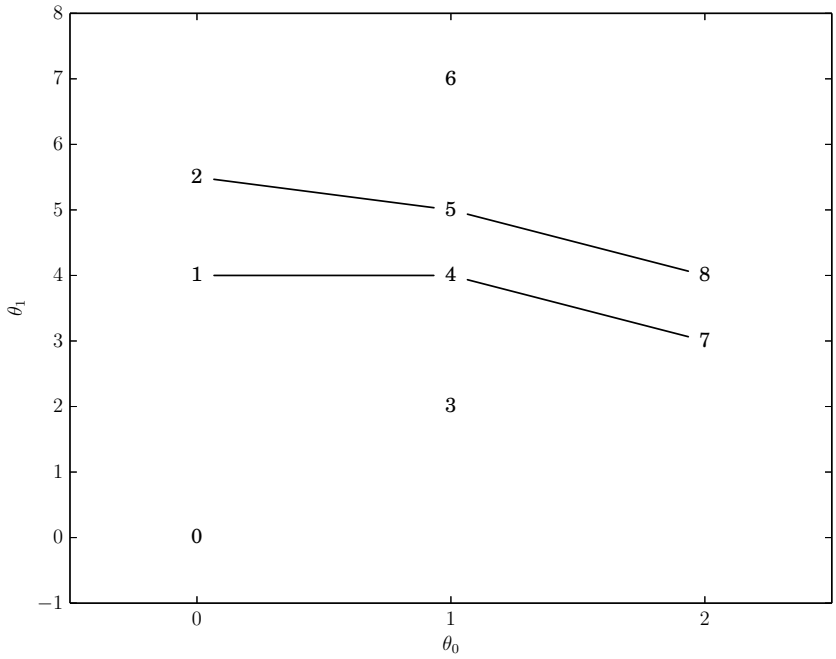

Fig. 4.6: **Two shortest paths using the LP method**

This may mean that suboptimal solutions may still be close enough. The tolerance of the solutions to suboptimality should be investigated, as if they are tolerant, fewer iterations of a barrier-based algorithm would be required to solve the problem. More information on linear programming and optimization in general can be found in [2].

4.2.2 Complexity

The LP formulation of the L-best paths problem gives results equivalent to the solution to the L-best paths problem proposed in [53]. The complexity of our algorithm is different from that in this paper. Assuming we use the algorithm in [22] to solve the LP, our program has a complexity of $O(M^7B^2)$ where M is the number of nodes (parameter sets) and B is the number of bits used to represent each number in the input. The complexity of the algorithm in [53] is equivalent to the Viterbi algorithm for finding the single best path through a trellis whose hth frame has $\binom{N_h}{L}\binom{N_{h+1}}{L}L!$ connections where N_h and N_{h+1} are

the number of nodes in two consecutive frames of the original lattice. Therefore, assuming a constant number N of nodes in each frame, its complexity is $O(\binom{N}{L}^2 L!)^2 T)$. If there are few nodes in each frame and a small number of paths are searched, the Viterbi formulation is superior as its complexity increases linearly with the number of frames in the lattice. On the other hand, if each frame has a large number of nodes or many paths are searched, the LP formulation is superior. Informally we have found this to agree with reality — both algorithms were tried when producing the figures in Section 4.3. Indeed the Viterbi formulation took prohibitvely long to compute when many paths were desired, as did the LP when many frames were considered.

It should be noted that in the special case that only 1 shortest path is searched an algorithm exists that requires on the order of N^2T calculations [40] where N is the number of nodes in each frame and T is the number of frames (assuming the same number of nodes in each frame): this algorithm is known as the Viterbi algorithm [10].

4.3 Partial paths on an example signal

We compare the greedy and LP based methods for peak matching on a synthetic signal. The signal is composed of $K = 6$ chirps of constant amplitude, the kth chirp s at sample n described by the equation

$$s_k(n) = \exp(j(\phi_k + \omega_k n + \frac{1}{2}\psi_k n^2))$$

The parameters for the 6 chirps are presented in Table 4.2.

Table 4.2. Parameters of kth chirp. f_0 and f_1 are the initial and final frequncy of the chirp in Hz.

k	ϕ_k	ω_k	ψ_k	f_0	f_1
0	0	0.20	2.45×10^{-6}	500	600
1	0	0.39	4.91×10^{-6}	1000	1200
2	0	0.59	7.36×10^{-6}	1500	1800
3	0	0.27	-7.36×10^{-6}	700	400
4	0	0.55	-1.47×10^{-5}	1400	800
5	0	0.82	-2.21×10^{-5}	2100	1200

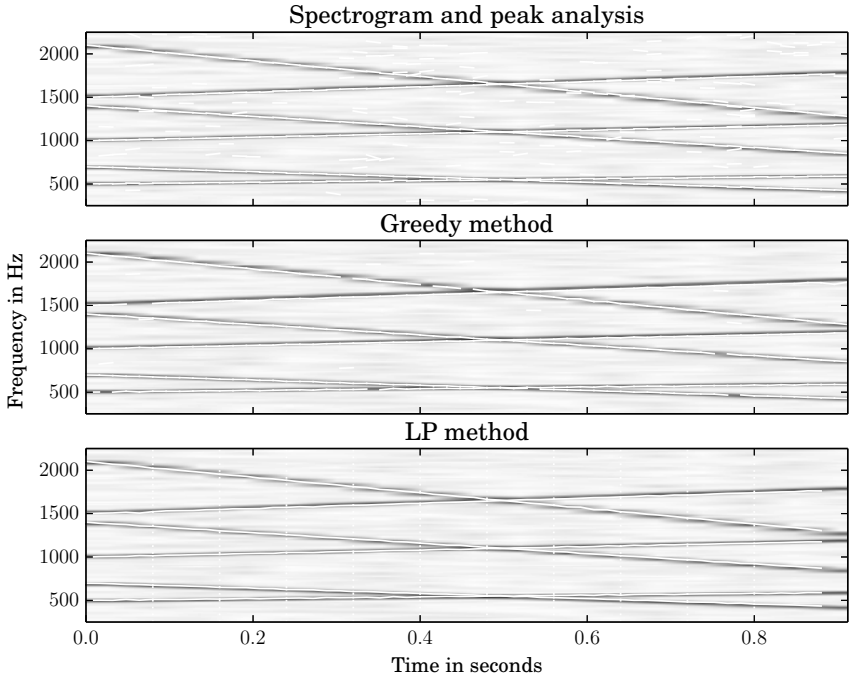

Fig. 4.7: **Compare greedy and LP partial tracking on chirps in noise, SNR 20 dB.** Line-segments representing the discovered partial paths.

Two 1 second long signals are synthesized at a sampling rate of 16000 Hz, the first with chrips 0–2, the second with chirps 3–5. We add Gaussian distributed white noise at several SNR to evalute the technique in the presence of noise.

A spectrogram of each signal is computed with an analysis window length of 1024 samples and a hop-size H of 256 samples. Local maxima are searched in 150 Hz wide bands spaced 75 Hz apart. A local maximum is only accepted if its amplitude is greater than -20 dB. At each local maximum the DDM is used to estimate the local chirp parameters, the ith set of parameters in frame h denoted $\theta_i^h = \left\{\phi_i^h, \omega_i^h, \psi_i^h\right\}$. The results of the analyses of both signals are lumped together and it is on this lumped data that we perform partial tracking.

We search for partial tracks using both the greedy and LP strategies. Both algorithms use the distance metric $\mathcal{D}_{\text{pr.}}$ between two parameters sets:

$$\mathcal{D}_{\text{pr.}}\left(\theta_i^h, \theta_{j,}^{h+1}\right) = \left(\omega_i^h + \psi_i^h H - \omega_j^{h+1}\right)$$

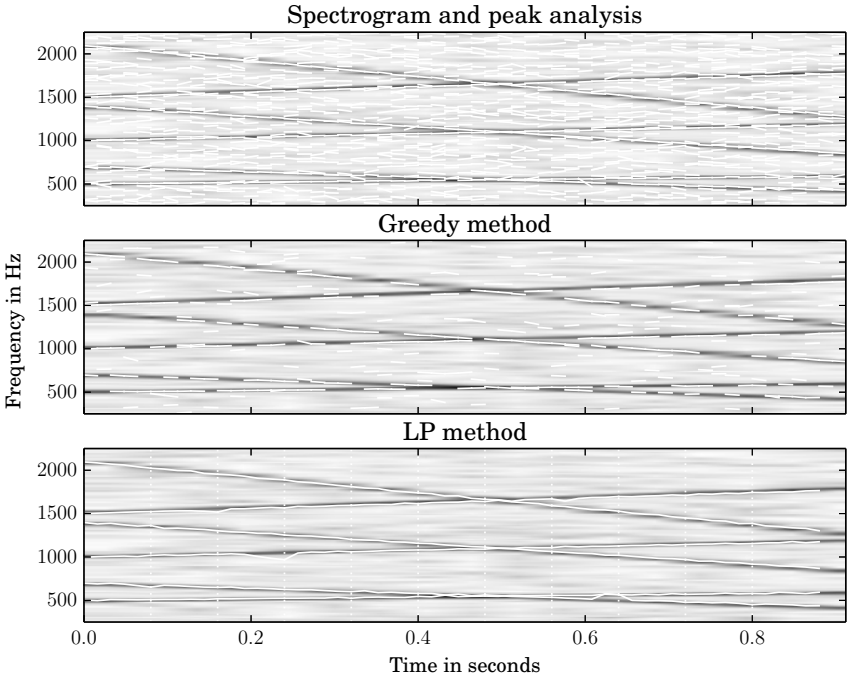

Fig. 4.8: **Compare greedy and LP partial tracking on chirps in noise, SNR 15 dB.** Line-segments representing the discovered partial paths.

Which is the error in predicting jth frequency in frame $h+1$ from the ith parameters in frame h. For the greedy method, the search for partial paths is restricted to one frame ahead like in [31]. For the LP method, to keep the computation time reasonable, we search over 6 frames for 6 best paths[5]. To maintain connected paths, the search on the next frames uses the end nodes of the last search as starting points. For both methods, the search is restricted to nodes between frequencies 250 to 2250 Hz.

Figures 4.9, 4.8 and 4.7 show discovered partial trajectories for signals with SNR of 10, 15 and 20 dB respectively. We can see that while the greedy method begins to perform poorly at an SNR of 15dB, the LP method still gives plausible partial trajectories for SNRs of 10 and 15 dB. At lower signal to noise ratios, the LP formulation gives some paths that do not correspond to an underlying partial. These could be filtered out by examining the cost of these paths and comparing them to the costs of the others. Those that deviate from a mean cost more than a certain amount should be rejected. This is the strategy used in

[5]The number of paths does not affect the computation time.

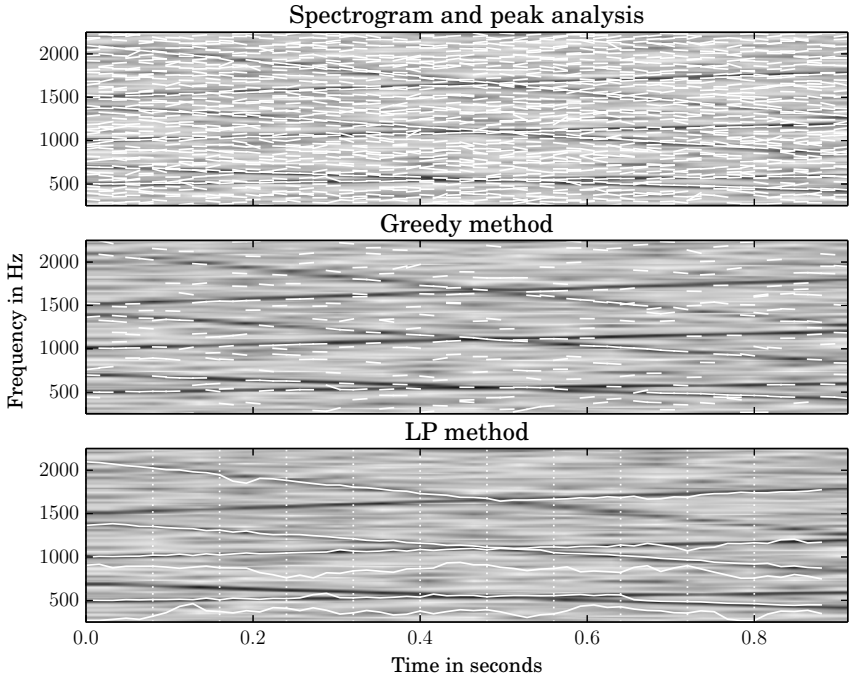

Fig. 4.9: **Compare greedy and LP partial tracking on chirps in noise, SNR 10 dB.** Line-segments representing the discovered partial paths.

Chapter 7 and illustrated in Figure 7.1.

But why did the LP discover a path not present in the underlying signal? This is due to the cost function, which finds a path with minimum prediction error in using the frequency and frequency slope coefficients of one node to predict another node's frequency coefficient. When there are many nodes in the original analysis it is not surprising that some unexpected path exists. An attribute of these erroneous paths is that they are not smooth. To deter the algorithm from finding such paths, regularization could be used like in Section 3.3.1 that minimizes the integral of the squared estimate of the path's second derivative. More on regularization in optimization can be found in [2, ch. 6.3].

4.4 Conclusion

In this chapter we reformulated the classical greedy algorithm of McAulay and Quatieri and showed that can be seen as a greedy algorithm for finding the L shortest paths in a

lattice. An algorithm was then proposed minimizing the sum of the L paths, using a linear programming approach. It was shown on synthetic signals that the new approach finds plausible paths in lattices with a large number of spurious nodes.

There are problems with the proposed approach. As discussed in 4.2, "jagged" paths should be removed using regularization. There are also situations where it is undesireable to have paths extend throughout the entire lattice. Acoustic signals produced by striking media, such as strings or bars, exhibit a spectrum where the upper partials decay more quickly than the lower ones (e.g., see Figure 7.4) — it would be desirable in these situations to have shorter paths for the upper partials, those decaying more quickly. This could be addressed as in [7] where the signal is divided into overlapping sequences of frames and partial paths are connected between sequences.

The proposed algorithm, while faster than algorithms based on the Viterbi algorithm, is still not fast. Assuming the same cost function $\mathcal{D}_{\text{pr.}}$ as in Section 4.3 it would be more efficient to consider narrow bands over which to search for paths when analysing signals with little frequency modulation. However, as we will see in Chapter 6, with different cost functions, the algorithm is useful for solving general L shortest paths problems outside of partial tracking.

Chapter 5

Extended phase model

In Chapter ?? techniques were presented for discovering partials in a signal. Each partial is a set of analysis points indexed by time. The information at each analysis point can be used to synthesize a portion of the partial and these are combined to give a signal representing a partial. The various techniques to synthesize these pieces of the signal discussed here differ in the orders used for analysis and those for synthesis. The first technique presented simply synthesizes signals of short duration using the estimated parameters and blends these segments together. We recognize that having multiple estimations of parameters at discrete times within the partial duration allow us to postulate, via interpolation, functions describing the partial of higher order than those whose parameters were estimated during analysis. It is shown that this strategy is not always to our advantage — interpolants of higher order than the underlying function can suffer from errors due to over-fitting. In the case of functions that are always better approximated by higher order polynomials, we will see that there is an advantage to using high order interpolation. These cases are illustrated through the analysis and synthesis of synthetic signals.

5.1 Partial synthesis

A popular technique for synthesizing partials from a set of analysis points is the *overlap-and-add* procedure [39], [34]. We assume that in the neighbourhood of τ_r the partial's signal is approximately described by the function $x(n) \approx f_{\tau_r}(n - \tau_r)$. To synthesize an approximation of x we sum windowed f_{τ_r} at multiple locations, windowed by a function w with finite support so the resulting signal has finite energy and the piecewise assumption is

maintained. For simplicity we assume the τ_r are equally spaced by H samples, and $\tau_0 = 0$, so we have $\tau_r = rH$. The length of the window function w is $M = VH + 1$ samples, with $V, H \in \mathbb{N}$[1]. The approximate signal at sample n is then

$$\tilde{x}(n) = \sum_{l=L_-}^{L^+} w(n - lH) f_{\tau_l}(n - \tau_l)$$

where

$$L_- = \left\lceil \frac{n}{H} \right\rceil - V$$

and

$$L_+ = \left\lfloor \frac{n}{H} \right\rfloor + V$$

This method has some drawbacks. Usually the function f is an approximation $\tilde{f}$ of the true underlying function. In the case of partial tracking, often partials that are too short are discarded or missed. At amplitude transients, these short partials are important for reproducing sharp attacks that are shorter than the window length. If these partials are missing, the resulting signal takes on a transient similar to the window shape. This could be overcome by choosing a window with a shape similar to the overall amplitude envelope in the attack region when resynthesizing an attack transient.

Another drawback is that no attempt is made to interpolate between the functions estimated at τ_r and τ_{r+1} using the model that the underlying sinusoids are non-stationary[2] From Equation 3.10 we know we can estimate a polynomial of arbitrary order for phase. We will see that using this additional information can give us an interpolating function closer to the underlying model.

[1] M is always odd so one may wonder how the Fast Fourier Transform can be used to invert F_{τ_r}, the frequency domain representaion of f_{τ_r}. Recall that the DTFT can be interpreted as the coefficients of a Fourier series that give the periodic version of the analysed signal. Also recall that we use window functions that are real and even. In practice the edges of the window are often equal to zero so that the length of the non-zero part is equal to the length of the DTFT N. In the case they are not, simply ensuring that values in the window indexed by integer multiples of N are 0, and that the value at the centre of the window is 1 will ensure proper synthesis [39, p. 244]. In that case, the values outside of the part of the window presented to the DTFT are folded into this region using the window indices modulo-N. See [39] and [34] for more details on this procedure.

[2] This is one of the causes of "pre-echo" when time-stretching using the STFT [43].

5.2 The interpolating analysis-synthesis system

In the following, we investigate the synthesis quality of three interpolating analysis-synthesis systems. The qualifier "interpolating" is used because each system takes the multiple sets of estimated parameters of a smaller order model and interpolates them with a higher oder model, which is then used for synthesis. The systems will be denoted $\mathscr{S}_{p,q}$ where p is the order of the analysis system and q the order of the synthesis system, e.g., a linear analysis system has $p = 1$, etc.

5.3 $\mathscr{S}_{1,3}$: the McAulay-Quatieri method

5.3.1 Analysis: linear phase

For the McAulay-Quatieri method, the model is a sinusoid of constant frequency (linear phase) in each analysis frame. To estimate the frequency of this sinusoid we find the bin with the most energy and find a refined estimate of the frequency as the maximum of a quadratic interpolating polynomial fit to this bin and its two neighbouring bins. This is a procedure documented in [46, p. 45]. The interpolation is best performed in the log-spectrum and on a spectrum produced using a window, such as a zero-padded Hann window, giving a wide enough main-lobe so that the three points lie on this lobe and not on side-lobes. A refined estimate of the amplitude of the sinusoid is obtained with this procedure as well.

In the original paper by McAulay and Quatieri, they do not use this technique but, as they show an example analysis of a speech signal, instead adjust the analysis window to be a multiple of the period of the glottal pulse. The bins of the DTFT used in the analysis will be integer multiples of the frequency given as the reciprocal of this period. Under the model of the speech signal as harmonically related sinusoids, the best estimate for the frequency is the bin of a local maximum, its amplitude the modulus of the spectrum at this maximum, and the phase the argument.

In our system, we use a fixed window size. To estimate the phase then we use Equation 3.11 with

$$\gamma_{\mathrm{MQ}}(n) = \exp(2\pi \frac{k^*}{M} n)$$

where k^* is the bin we have determined to correspond to the frequency of the sinusoid. The inital phase is then $\Im\{c_0\}$.

5.3.2 Synthesis: cubic phase

Given two local maxima of the DTSTFT $X(\tau_0, \omega_0)$ and $X(\tau_1, \omega_1)$, where $H = \tau_1 - \tau_0$ we can conjecture a cubic polynomial phase function for the imaginary part of the phase argument

$$\tilde{\phi}(n) = \Im\{c_3\}(n-\tau_0)^3 + \Im\{c_2\}(n-\tau_0)^2 + \Im\{c_1\}(n-\tau_0) + \Im\{c_0\} \tag{5.1}$$

By noting that we have 2 measurements of the phase and frequency, $\angle\{X(\tau_0, \omega_0)\}$ and $\angle\{X(\tau_1, \omega_1)\}$, and the frequency is the derivative of the phase, we can solve for the coefficients of the polynomial phase function using the following linear system of equations, assuming the DTSTFT was computed using a real and even window

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ H^3 & H^2 & H & 1 \\ 0 & 0 & 1 & 0 \\ 3H^2 & 2H & 1 & 0 \end{pmatrix} \begin{pmatrix} \Im\{c_3\} \\ \Im\{c_2\} \\ \Im\{c_1\} \\ \Im\{c_0\} \end{pmatrix} = \begin{pmatrix} \angle\{X(\tau_0, \omega_0)\} \\ \angle\{X(\tau_1, \omega_1)\} + 2\pi M \\ \omega_0 \\ \omega_1 \end{pmatrix} \tag{5.2}$$

We choose M so that

$$\int_0^H (\frac{d^2\tilde{\phi}}{dt^2}(t))^2 dt \tag{5.3}$$

is minimized in order to have a smooth evolution of phase in the interpolated region. M is necessary because some integer number of periods of a sinusoid will have passed from times τ_0 to τ_1. Informally we choose M so that a polynomial describing the phase evolution between these two times takes a direct route, which is a plausible criterion because a signal with more radical phase variation would unlikely exhibit a spectrum that could be well described by two points in the time-frequency plane, i.e., the signal would exhibit a large bandwidth. See [31, p. 751] for further clarification.

As only two measurements of the amplitude of the sinusoid are available, $|X(\tau_0, \omega_0)|$ and $|X(\tau_1, \omega_1)|$, the coefficients c_3 and c_2 are purely imaginary and the real parts of c_1 and c_0 are determined as

$$\begin{pmatrix} 0 & 1 \\ H & 1 \end{pmatrix} \begin{pmatrix} \Re\{c_1\} \\ \Re\{c_0\} \end{pmatrix} = \begin{pmatrix} \log(|X(\tau_0, \omega_0)|) \\ \log(|X(\tau_1, \omega_1)|) \end{pmatrix} \tag{5.4}$$

5.4 $\mathscr{S}_{2,3}$ and $\mathscr{S}_{2,5}$: the DDM-based methods

Here we extend the $\mathscr{S}_{1,3}$ model of McAulay-Quatieri to account for the additional parameters estimated via the DDM. For the $\mathscr{S}_{2,3}$ model, we must introduce additional constraints into the system as we have more estimated parameters than are available in the synthesis model. It would be possible to solve this system via least-squares, but the proposed constraints simplify analytically the expression maximizing the smoothness of the phase function, and give satisfactory results. For the $\mathscr{S}_{2,5}$ model the derivation is straightforward as in the $\mathscr{S}_{1,3}$ case — there are the same number of estimated parameters as there are parameters in the model.

5.4.1 Analysis: quadratic phase

The DDM is used on segments of the signal to estimate the parameters of sinusoid with a complex quadratic phase polynomial. This sinusoid has the form

$$x_a(n) = \exp\left(a_2 n^2 + a_1 n + a_0\right) \tag{5.5}$$

with $a_i \in \mathbb{C}$.

We can estimate the coefficients of Equation 5.5 using the DDM. We write Equation 3.10 in matrix form with $Q = 2$

$$\begin{pmatrix} \langle \mathcal{T}^0 x, \overline{\psi}_1 \rangle & 2\langle \mathcal{T}^1 x, \overline{\psi}_1 \rangle \\ \vdots & \vdots \\ \langle \mathcal{T}^0 x, \overline{\psi}_R \rangle & 2\langle \mathcal{T}^1 x, \overline{\psi}_R \rangle \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} -\left\langle x, \frac{d\overline{\psi}_1}{dn} \right\rangle \\ \vdots \\ -\left\langle x, \frac{d\overline{\psi}_R}{dn} \right\rangle \end{pmatrix} \tag{5.6}$$

From this we recognize we need to define three functions

$$\langle \mathcal{T}^0 x, \overline{\psi}_k \rangle = \sum_{m=0}^{M-1} w(m) x(m+\tau) \exp(-j2\pi \frac{km}{M}) \tag{5.7}$$

$$\langle \mathcal{T}^1 x, \overline{\psi}_k \rangle = \sum_{m=0}^{M-1} m w(m) x(m+\tau) \exp(-j2\pi \frac{km}{M}) \tag{5.8}$$

$$\left\langle x, \frac{d\overline{\psi}_k}{dn} \right\rangle = -j2\pi \frac{k}{M} X_{p_1}(\tau, k) + \sum_{m=0}^{M-1} \frac{dw}{dn}(m) x(m+\tau) \exp(-j2\pi \frac{km}{M}) \tag{5.9}$$

Where M is the length of the window and k is the frequency "bin"[3]. We also only consider x at the samples $m = [0, \ldots, M-1]$ because we can always shift the time reference to view an arbitrary contiguous segment of signal with these indices.

We then find k^* such that X is maximum. If multiple components are present in the signal and are sufficiently separated in frequency, we can split the signal up into frequency bands and find local maxima. A technique for doing so is described in [46, p. 42]. To have a system of equations with a unique solution, we take the two adjacent bins $k-1$ and $k+1$ to have enough unique atoms for Equation 3.10. These bins should only contain energy from the component whose parameters we are interested in measuring — this is true if the components are adequately separated in time and frequency. We could choose only two bins to have a non-singular system, and there are many possibilities for choosing different atoms [1, p. 4639]. We choose three from the same frame to have improved estimation accuracy in situations where components are adaquately separated in frequency, while avoiding a more sophisticated local peak selection procedure. Then a_2 and a_1 can be determined by solving the linear system

$$\begin{pmatrix} \langle \mathcal{T}^0 x, \overline{\psi}_{k-1} \rangle & \langle \mathcal{T}^1 x, \overline{\psi}_{k-1} \rangle \\ \langle \mathcal{T}^0 x, \overline{\psi}_{k} \rangle & \langle \mathcal{T}^1 x, \overline{\psi}_{k} \rangle \\ \langle \mathcal{T}^0 x, \overline{\psi}_{k+1} \rangle & \langle \mathcal{T}^1 x, \overline{\psi}_{k+1} \rangle \end{pmatrix} \begin{pmatrix} a_1 \\ 2a_2 \end{pmatrix} = \begin{pmatrix} -\left\langle x, \frac{d\overline{\psi}_{k-1}}{dn} \right\rangle \\ -\left\langle x, \frac{d\overline{\psi}_{k}}{dn} \right\rangle \\ -\left\langle x, \frac{d\overline{\psi}_{k+1}}{dn} \right\rangle \end{pmatrix} \tag{5.10}$$

With a_1 and a_2 determined, we can use Equation 3.11 to estimate a_0. We will write a_i^τ to refer to coefficient i determined at time τ.

5.4.2 Synthesis: cubic order ($\mathscr{S}_{2,3}$)

In this section we describe how to obtain a cubic phase polynomial from local estimations of the coefficients of a quadratic phase polynomial.

[3] For tractability, the functions $X(\tau, k)$ are only evaluated at a finite number of frequencies, which are often called "bins" in the signal processing literature.

The phase part

A complex sinusoid with cubic phase has the following form:

$$\beta(n) = \exp\left(j\left(b_3 n^3 + b_2 n^2 + b_1 n + b_0\right)\right) \tag{5.11}$$

with $b_i \in \mathbb{R}$. This sinusoid has magnitude 1 everywhere, only its phase is changing.

Once the $\boldsymbol{a}^\tau$ have been determined at two times τ_0 and τ_1, with $H = \tau_1 - \tau_0$, and these times have between determined as connected (see Chapter 4), we can write a system of equations to determine an interpolating cubic phase polynomial. To avoid numerical instabilities and for simplicity, we shift the time origin so that $\tau_0 = 0$. This means $b_0 = \Im\{a_0^{\tau_0}\}$. To reduce the size of the system, we require that

$$\frac{d\phi}{dn}\left(\frac{H}{2}\right) = \frac{1}{2}\left(\Im\{a_1^{\tau_0}\} + \Im\{a_1^{\tau_1}\}\right)$$

and

$$\frac{d^2\phi}{dn^2}\left(\frac{H}{2}\right) = \frac{1}{2}\left(\Im\{a_2^{\tau_0}\} + \Im\{a_2^{\tau_1}\}\right)$$

i.e., the frequency and first-order frequency modulation in the middle of the segment are the average of the two measured coefficients. Finally we require the that the change in phase from time 0 to H correspond to that what was observed, but account for the cycles that were not observed by adding an integer number of 2π radians. If $\phi(n) = j(b_3 n^3 + b_2 n^2 + b_1 n + b_0)$, then

$$\phi(H) = \Im\{a_0^{\tau_1}\} - \Im\{a_0^{\tau_0}\} + 2\pi U^*$$

where $U^* \in \mathbb{Z}$ is determined to minimize Equation 5.3, in this case:

$$\tilde{U} = \arg\min_U \int_0^H \left(6b_3 t + 2b_2\right)^2 dt \tag{5.12}$$

which is then rounded to the nearest integer to give U^*. To summarize we have

$$\begin{pmatrix} H^3 & H^2 & H \\ \frac{3}{4}H^2 & H & 1 \\ 3H & 2 & 0 \end{pmatrix} \begin{pmatrix} b_3 \\ b_2 \\ b_1 \end{pmatrix} = \begin{pmatrix} \Im\{a_0^{\tau_1}\} - \Im\{a_0^{\tau_0}\} + 2\pi U^* \\ \frac{1}{2}\left(\Im\{a_1^{\tau_0}\} + \Im\{a_1^{\tau_1}\}\right) \\ \frac{1}{2}\left(\Im\{a_2^{\tau_0}\} + \Im\{a_2^{\tau_1}\}\right) \end{pmatrix} \tag{5.13}$$

Solving for $b_1, \ldots, b_3$, we have

$$\begin{pmatrix} b_3 \\ b_2 \\ b_1 \end{pmatrix} = \begin{pmatrix} \frac{4}{H^3}\left(\Im\left\{a_0^{\tau_1}\right\} - \Im\left\{a_0^{\tau_0}\right\} + 2\pi U^*\right) - \frac{2}{H^2}\left(\Im\left\{a_1^{\tau_1}\right\} + \Im\left\{a_1^{\tau_0}\right\}\right) \\ \frac{-6}{H^2}\left(\Im\left\{a_0^{\tau_1}\right\} - \Im\left\{a_0^{\tau_0}\right\} + 2\pi U^*\right) - \frac{3}{H}\left(\Im\left\{a_1^{\tau_1}\right\} + \Im\left\{a_1^{\tau_0}\right\}\right) + \frac{1}{4}\left(\Im\left\{a_2^{\tau_0}\right\} + \Im\left\{a_2^{\tau_1}\right\}\right) \\ \frac{-H}{4}\left(\Im\left\{a_2^{\tau_0}\right\} + \Im\left\{a_2^{\tau_1}\right\}\right) + \frac{3}{H}\left(\Im\left\{a_0^{\tau_1}\right\} - \Im\left\{a_0^{\tau_0}\right\} + 2\pi U^*\right) - \Im\left\{a_1^{\tau_1}\right\} - \Im\left\{a_1^{\tau_0}\right\} \end{pmatrix}$$

and then $\tilde{U}$ is determined using Equation 5.12 to be

$$\tilde{U} = \frac{1}{4\pi}\left[H\left(\Im\left\{a_1^{\tau_1}\right\} + \Im\left\{a_1^{\tau_0}\right\}\right) - 2\left(\Im\left\{a_0^{\tau_1}\right\} - \Im\left\{a_0^{\tau_0}\right\}\right)\right]$$

and then rounded to obtain U^*.

The amplitude part

Solving for the cubic polynomial describing the local amplitude function

$$\mu(n) = \exp\left(c_3 n^3 + c_2 n^2 + c_1 n + c_0\right) \tag{5.14}$$

with $c_i \in \mathbb{R}$, is more straightforward analytically as it does not require solving to maximize the smoothness of resulting polynomial. To require continuity at the end-points of our polynomial, we require

$$\mu(0) = \Re\left\{a_0^{\tau_0}\right\}$$

and

$$\mu(H) = \Re\left\{a_0^{\tau_1}\right\}$$

The first constraint is satisfied simply by setting $c_0 = \Re\left\{a_0^{\tau_0}\right\}$. The second will be accounted for in a constrained least-squares solution for the other coefficients. The other observations are

$$\frac{d\mu}{dn}(0) = \Re\left\{a_1^{\tau_0}\right\}$$

$$\frac{d\mu}{dn}(H) = \Re\left\{a_1^{\tau_1}\right\}$$

$$\frac{d^2\mu}{dn^2}(0) = \Re\left\{a_2^{\tau_0}\right\}$$

$$\frac{d^2\mu}{dn^2}(H) = \Re\left\{a_2^{\tau_1}\right\}$$

The constrained least-squares problem to be solved is then

$$\begin{pmatrix} 0 & 0 & 1 \\ 3H^2 & 2H & 1 \\ 0 & 2 & 0 \\ 6H & 2 & 0 \end{pmatrix} \begin{pmatrix} c_3 \\ c_2 \\ c_1 \end{pmatrix} = \begin{pmatrix} \Re\{a_1^{\tau_0}\} \\ \Re\{a_1^{\tau_1}\} \\ \Re\{a_2^{\tau_0}\} \\ \Re\{a_2^{\tau_1}\} \end{pmatrix} \tag{5.15}$$

subject to

$$\begin{pmatrix} H^3 & H^2 & H \end{pmatrix} \begin{pmatrix} c_3 \\ c_2 \\ c_1 \end{pmatrix} = \begin{pmatrix} \Re\{a_0^{\tau_1}\} \end{pmatrix}$$

This can be solved using numerical methods, in particular, using a specific interpretation of weighted least-squares [13, p. 266].

5.4.3 Synthesis: quintic order ($\mathscr{S}_{2,5}$)

The phase part

Solving for the coefficients of a quintic phase polynomial is done very similarly to Section 5.4.2. As we have the same number of analysed parameters and synthesis parameters, no constraints have to be introduced to solve the system apart from the value U that maximizes smoothness of the phase function. The quintic phase polynomial is[4]

$$\lambda(n) = \exp\left(j\left(u_5 n^5 + u_4 n^4 + u_3 n^3 + u_2 n^2 + u_1 n + u_0\right)\right) \tag{5.16}$$

with $u_i \in \mathbb{R}$. We have

$$\begin{aligned} \phi(0) &= \Im\{a_0^{\tau_0}\} \\ \frac{d\phi}{dn}(0) &= \Im\{a_1^{\tau_0}\} \\ \frac{d^2\phi}{dn^2}(0) &= \Im\{a_2^{\tau_0}\} \end{aligned}$$

[4] Remember, this sinusoid has constant amplitude of 1 and this function only describes its change of phase.

and solving for the remaining coefficients is done using the linear system of equations:

$$\begin{pmatrix} H^5 & H^4 & H^3 \\ 5H^4 & 4H^3 & 3H^2 \\ 20H^3 & 12H^2 & 6H \end{pmatrix} \begin{pmatrix} u_5 \\ u_4 \\ u_3 \end{pmatrix} = \begin{pmatrix} -\frac{H^2}{2}\Im\left\{a_2^{\tau_0}\right\} - H\Im\left\{a_1^{\tau_0}\right\} + \Im\left\{a_0^{\tau_1}\right\} - \Im\left\{a_1^{\tau_1}\right\} + 2\pi U^* \\ -H\Im\left\{a_2^{\tau_0}\right\} + \Im\left\{a_1^{\tau_1}\right\} - \Im\left\{a_1^{\tau_0}\right\} \\ \Im\left\{a_2^{\tau_1}\right\} - \Im\left\{a_2^{\tau_0}\right\} \end{pmatrix} \tag{5.17}$$

The smoothness maximizing $\tilde{U}$ is found as

$$\tilde{U} = \frac{1}{80\pi}\left[20H(\Im\left\{a_1^{\tau_0}\right\} + \Im\left\{a_1^{\tau_1}\right\}) + H^2(\Im\left\{a_2^{\tau_0}\right\} - \Im\left\{a_2^{\tau_1}\right\}) + 40(\Im\left\{a_0^{\tau_0}\right\} - \Im\left\{a_0^{\tau_1}\right\})\right]$$

and then rounded to produce U^* as above.

The quintic interpolating phase polynomial has been proposed in a previous paper [12] although they do not directly estimate the frequency slope, choosing instead to derive it using the difference in frequency between two analysis frames.

The amplitude part

Solving for the quintic amplitude polynomial

$$\rho(n) = \exp\left(v_5 n^5 + v_4 n^4 + v_3 n^3 + v_2 n^2 + v_1 n + v_0\right) \tag{5.18}$$

with $v_i \in \mathbb{R}$, is as follows:

$$\mu(0) = \Re\left\{a_0^{\tau_0}\right\}$$
$$\frac{d\mu}{dn}(0) = \Re\left\{a_1^{\tau_0}\right\}$$
$$\frac{d^2\mu}{dn^2}(0) = \Re\left\{a_2^{\tau_0}\right\}$$

and solving for the remaining coefficients is done using the linear system of equations:

$$\begin{pmatrix} H^5 & H^4 & H^3 \\ 5H^4 & 4H^3 & 3H^2 \\ 20H^3 & 12H^2 & 6H \end{pmatrix} \begin{pmatrix} v_5 \\ v_4 \\ v_3 \end{pmatrix} = \begin{pmatrix} -\frac{H^2}{2}\Re\left\{a_2^{\tau_0}\right\} - H\Re\left\{a_1^{\tau_0}\right\} + \Re\left\{a_0^{\tau_1}\right\} - \Re\left\{a_1^{\tau_1}\right\} \\ -H\Re\left\{a_2^{\tau_0}\right\} + \Re\left\{a_1^{\tau_1}\right\} - \Re\left\{a_1^{\tau_0}\right\} \\ \Re\left\{a_2^{\tau_1}\right\} - \Re\left\{a_2^{\tau_0}\right\} \end{pmatrix} \tag{5.19}$$

5.5 Evaluation

We compared the quality of an analysis-synthesis system using the original $\mathscr{S}_{1,3}$ method, $\mathscr{S}_{2,3}$ method, and the $\mathscr{S}_{2,5}$. Frequency- and amplitude-modulated sinusoids were synthesized and then analysed frame-by-frame using the DDM to estimate their initial phase (amplitude), frequency (amplitude slope), and frequency-modulation (amplitude-modulation). Afterwards, the signals were resynthesized using the estimated parameters and compared to the original. We are interested in seeing in what cases higher order phase polynomials will improve the accuracy of synthesis.

The extended methods

Two extended methods are evaluated: the cubic and quartic phase and amplitude polynomials derived from DDM-estimated coefficients. For the polynomials of both orders, the DDM is used exactly as described in Section **??** to estimate the parameters. The difference is in how the interpolating polynomials are formed between parameters estimated at two time points. For the cubic polynomials, the interpolation method described in Section **??** is used for phase, while that in Section **??** is used for amplitude. For the quintic polynomials the interpolation methods described in Sections **??** and **??** are used for phase and amplitude, respectively.

5.5.1 Evaluation on sinusoid of cubic phase

Table 5.1

Time (seconds)	0	0.25	0.5
Frequency (Hz)	100	200	100

Table 5.2

Time (seconds)	0	0.1	0.3	0.5
Amplitude (dB)	-10	0	0	-10

The initial evalutation illustrates how higher order phase polynomials will not necessarily improve the quality of synthesis if the underlying phase function is a polynomial of lower order than the polynomials used for synthesis. As we will see, the estimated phase functions suffer from "overfitting".

The synthesized signal has 3 frequency break-points and an initial phase, therefore its phase function can be interpolated by a cubic polynomial

$$x_\phi(n) = \exp\left(g_3 n^3 + g_2 n^2 + g_1 n + g_0\right)$$

the frequency break-points are summarized in Table 5.1. The initial phase is 0 radians.

A quartic polynomial is used for the amplitude function

$$x_\mu(n) = \exp\left(h_4 n^4 + h_3 n^3 + h_2 n^2 + h_1 n + h_0\right)$$

and its amplitude break-points are summarized in Table 5.2.

These polynomials are chosen because their orders are greater than or equal to the order of the synthesis model in the $\mathscr{S}_{2,3}$ system and less than the order of the synthesis model in the $\mathscr{S}_{2,5}$ system.

The signal was sampled with a sampling rate of 16000 Hz and was analysed every 256 samples with an analysis window of length 1024 samples. For the DDM method, the $\mathcal{C}^1$ 4-Term Blackman-Harris was used (see Section 3.5).

The results of the evalution are presented in Figures **??** through **??**.

5.5.2 Evaluation on sinusoid of exponential phase

The previous evaluation of this analysis-synthesis system was on a sinusoid with small order polynomial phase. In the cases where we observed overfitting, the polynomial used for synthesis was of higher order than the true underlying one — the interpolating polynomials were more times differentiable than the true polynomial. We propose evaluating the system on an infinitely differentiable and analytic phase function. The rationale behind this stems from the definition of an analytic function: one whose power series representation (a polynomial) converges to the function as the number of terms approches infinity. What this means is, in the region of convergence, the larger the number of terms in the approximating polynomial, the better the approximation to the true underlying function. The exponential

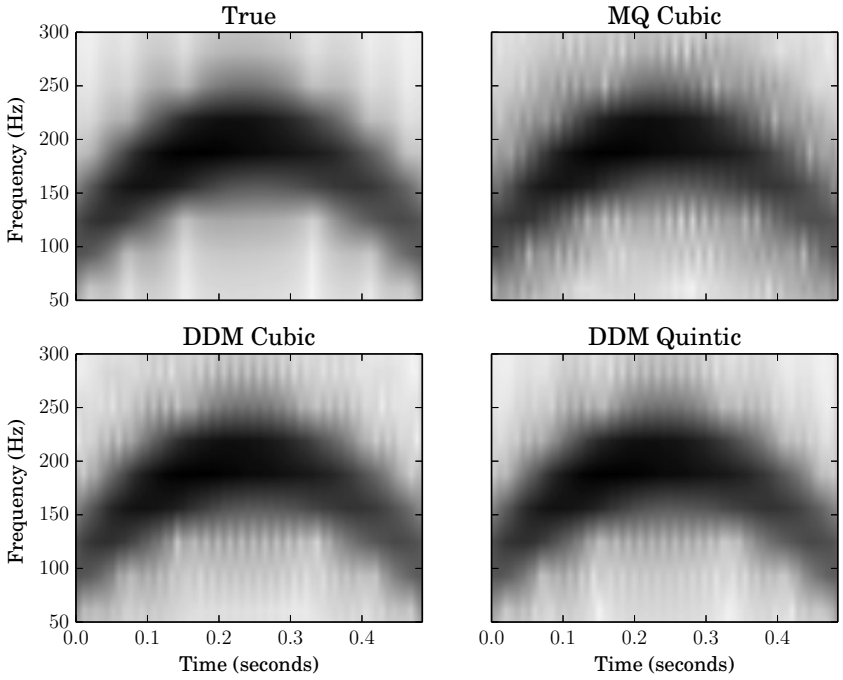

Fig. 5.1: **Spectrogram of original and resynthesized signals.** Spectrograms of the true signal and estimated signals for the polynomial phase signal.

function

$$y = \exp(x), x, y \in \mathbb{R}$$

is one such function whose power series is

$$\exp(x) = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

To find the radius of convergence, we use the ratio test

$$\lim_{n\to\infty} \frac{|1/n!|}{|1/(n+1)!|} = \lim_{n\to\infty} n = \infty$$

i.e., the power series of the exponential function converges everywhere and so using more terms of its power series will improve its approximation for any $x \in \mathbb{R}$.

The exponential function arises in music. In 12-tone equal temperament tuning, to find

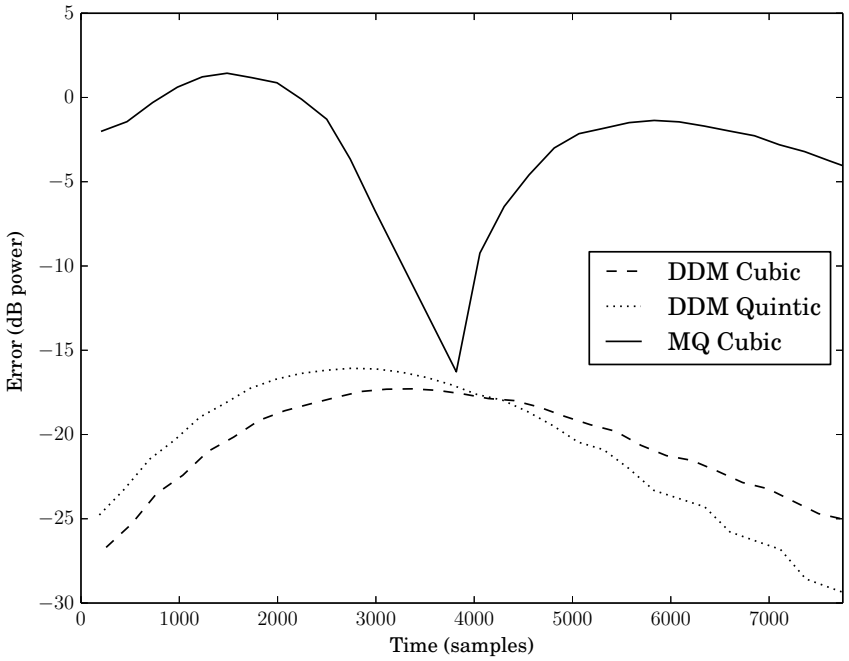

Fig. 5.2: **Original vs. estimated signals: upper error bound.** The power of the error when subtracting the original signal from the estimated signal. The local upper bound on the error was produced by connecting the local maxima in the error data.

the frequency f_1 of a pitch b-semitones away from the frequency f_0 we compute

$$f_1 = f_0 2^{\frac{b}{12}} = f_0 \exp(\log(2)\frac{b}{12})$$

So a linear transition from pitch b_0 to b_1 is an exponential change in frequency. This could be observed in recordings of performances of the *portamento* gesture.

We synthesize a sinusoid of length N samples with exponential phase and use the same analysis system as in Section 5.5.1 to evaluate the synthesis accuracy for piece-wise interpolating polynomials of cubic and quartic order for phase. The signal x is defined

$$x(n) = \exp(\frac{2\pi f_0}{\log(2)c_1} \exp(j(c_1 n + c_0)))$$

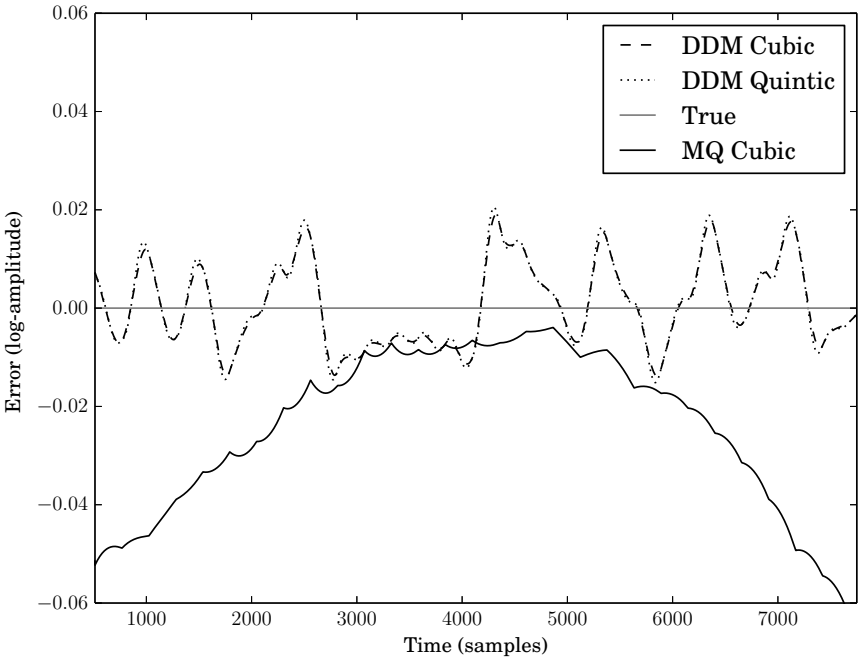

Fig. 5.3: **Log-amplitude function error.** This shows the error of the interpolated signals when compared with the original signal for the three proposed methods.

with

$$c_0 = \log(2)\frac{b_0}{12}$$

and

$$c_1 = \log(2)\frac{b_1}{12N}$$

i.e., a signal starting at pitch b_0 with frequency f_0 and arriving at pitch b_1 in N samples. We keep the amplitude of the signal constant in this evalutation as we are interested in the accuracy of the phase reconstruction. The same proceduare as in Section 5.5.1 is used to estimate the parameters of piece-wise interpolating phase polynomials. We can see in Figure 5.6 that the resynthesis accuracy is greater for higher order polynomials. Spectrograms of the original and estimated signals are plotted in Figure 5.7.

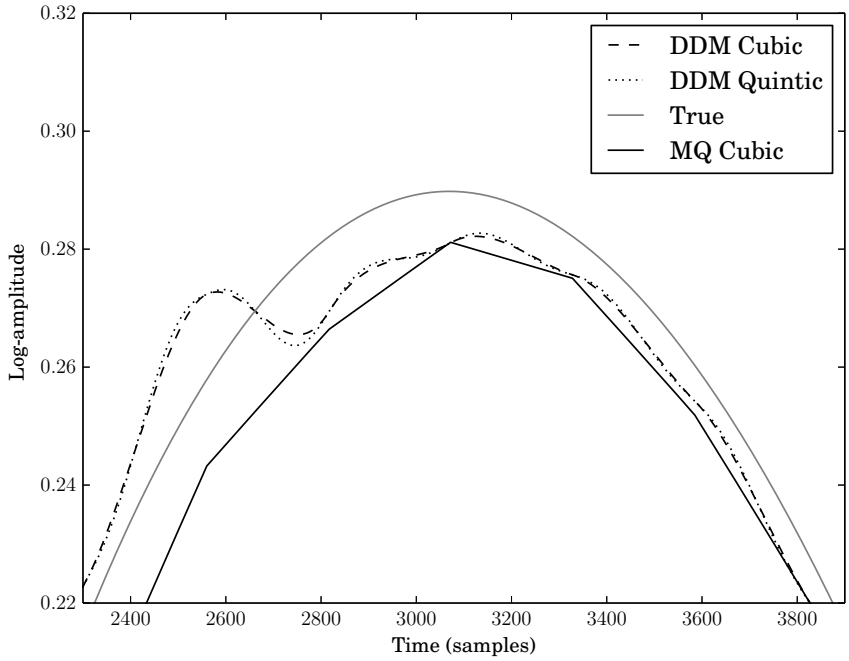

Fig. 5.4: **Log-amplitude function.** This compares the original log-amplitude function with the interpolated log-amplitude functions. The log-amplitude functions are considered because these are the real part of the polynomial exponents in the complex sinusoid model.

5.6 Conclusion

5.6.1 Polynomial phase function

Out of the three proposed methods it appears that the modified cubic interpolation method works superiorly for the signal model considered. We obseve overfitting by the higher-order quintic model in Figure 5.4, compromising the accuracy of resynthesis. Even the proposed cubic model shows some overfitting in this case. This is consistent with the results of [12]. From Figure 5.5 it is clear that the DDM based methods provide superior estimation of the phase function — this is not the case for the log-amplitude function. Depending on the underlying signal, perhaps better results can be obtained by postulating a lower-order amplitude function and higher-order phase function. The possibility of errors arising from numerical accuracy when evaluating the quintic polynomials has been ruled out. We

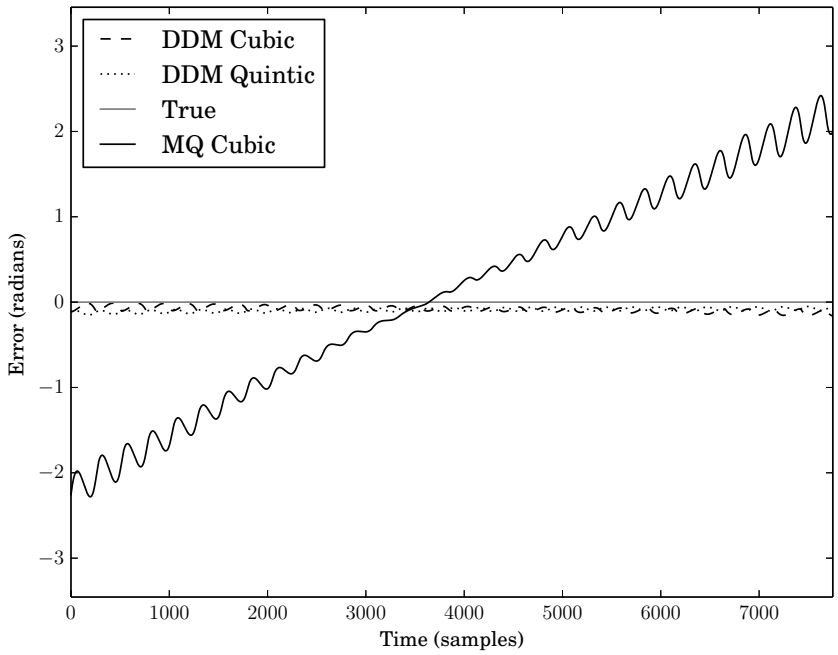

Fig. 5.5: **Phase function error.** This compares the theoretical phase function with the interpolated phase functions. The phase functions are considered because these are the imaginary part of the polynomial exponents in the complex sinusoidal model. The errors are "wrapped" to lie between $-\pi$ and π. The errors stem from both the estimation of the phase and the interpolation of phase between analysis points. As a frequency modulated sinusoid is considered, it is not surprising that the stationary frequency assumption of the McAulay-Quatieri model exhibits the most errors.

evalutated these polynomials using an implementation of Horner's method that keeps track of the error bound [19, p. 95]: the errors are negligible, see Figure 5.8 for the results.

5.6.2 Exponential phase function

The quintic interpolation, the polynomial of highest order, performs the most accurate resynthesis. This is consistent with the analytic property of the exponential function and an encouraging result as it suggests arbitrary analytic phase functions can be approximated with arbitrary accuracy simply by increasing the order of the interpolating polynomials. Many models of musical gestures involve such functions, apart from the portamento gesture modeled by an exponential phase function, vibrato can be modeled as a sinusoid with

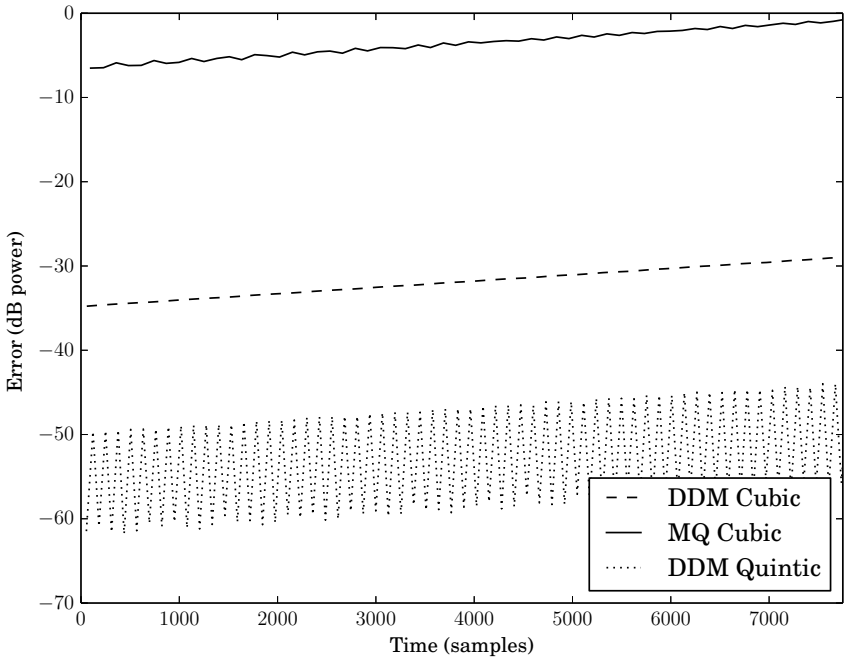

Fig. 5.6: **Original vs. estimated signals: upper error bound.** The power of the error when subtracting the original signal from the estimated signal for the signals of exponential phase. The local upper bound on the error was produced by connecting the local maxima in the error data.

sinusoidal phase [27]. The DDM-based analysis system combined with the higher order polynomial phase synthesis system presented here allow for accurate modeling of these gestures.

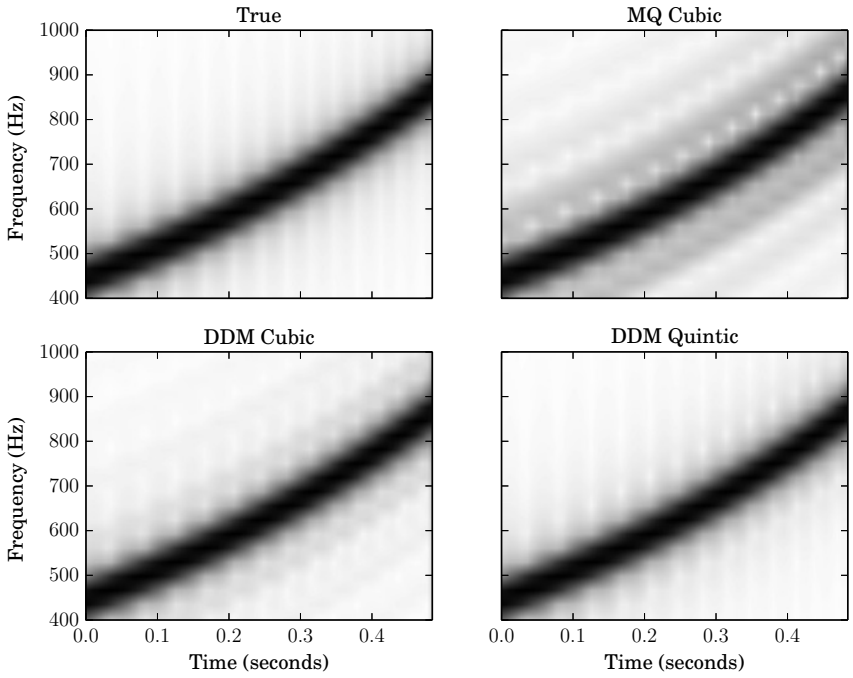

Fig. 5.7: **Spectrogram of original and resynthesized signals.** Spectrograms of the true signal and estimated signals for the exponential phase signal.

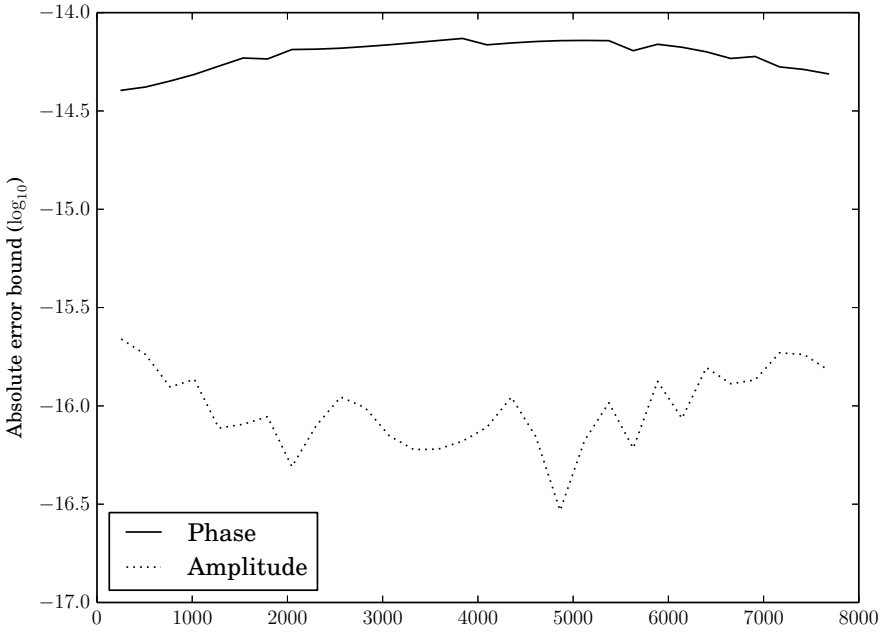

Fig. 5.8: **Polynomial evaluation error bound.** The error bound in evaluating the quintic amplitude and phase polynomials using Horner's method. This plot was produced by plotting only the local maxima of the error bound data in order to reduce the plot's range.

Chapter 6

Experiment: Partial grouping by amplitude- and frequency-modulation

6.1 Introduction

To evaluate whether the grouping of partials with common AM and FM parameters is plausible, we synthesize a set of parameters and test by corrupting the parameters with noise and adding spurious sets of parameters that should not belong to any sources.

6.2 Methodology

We assume parameters have been estimated already so we start from theoretical values for the amplitude, frequency, frequency modulation and amplitude modulation. On each frame of analysis data, i.e., for parameters belonging to the same time instant, we consider each data-point as a multi-dimensional random variable. With these random variables, we compute principal components in order to produce a variable with maximum variance. This variable is classified using a clustering algorithm and we evalutate the results. A summary follows:

- Parameters are synthesized from a theoretical mixture of AM and FM sinusoids. Spurious data are added to these parameters.
- Principal components analysis is carried out on the parameters happening at one time instance.

- A histogram is made of the first principal components. Values sharing a bin with too few other values are discarded to remove spurious data points.

- Initial means and standard deviations for the Gaussian mixture models are made by dividing the histogram into equal parts by area and choosing the centres of these parts.

- The EM algorithm for Gaussian mixture models is carried out to classify the sources.

6.3 Evaluation

The algorithm is run on a typical source separation problem to evaluate its plausibility.

6.4 Synthesis

Our model makes available the parameters summarized in Table 6.1. Time values are in seconds, frequency values are in Hz and phase values are in radians.

H	duration between data-point calculations in samples (i.e., the hop size).
N_p	number of sources.
p	which source.
$f_{0,p}$	fundamental frequency.
K_p	number of harmonics.
$k_{60,p}$	harmonic number 60 dB lower than the first.
B_p	the inharmonicity coefficient.
$\phi_{0,p}$	initial phase.
$\phi_{0,f,p}$	initial FM phase.
$t_{60,p}$	time until amplitude of partial has dropped 60 dB.
$t_{\text{attack},p}$	time duration of attack portion.
$A_{f,p}$	amplitude of FM.
$f_{f,p}$	frequency of FM.
s_p	the signal representing the pth source.

Table 6.1. Synthesis parameters

To incorporate inharmonicity often observed in real string instruments where the strings

exhibit some stiffness, we define the *stretched* harmonic numbers as follows [49][1]

$$K_B(k) = k(1 + Bk^2)^{\frac{1}{2}} \tag{6.1}$$

Each source is synthesized using the following equation:

$$s_p(t) = \sum_{k=1}^{K_p} A_p(k,t) \exp(j(2\pi f_{0,p} t - \frac{A_{f,p}}{f_{f,p}} \cos(2\pi f_{f,p} t + \phi_{0,f,p}) K_{B_p}(k) + \phi_{0,p})) \tag{6.2}$$

where

$$A_p(k,t) = \begin{cases} \exp(a_{60,p} t + a_{k,60,p} k) \cos^2(\frac{\pi}{2}(\frac{t}{t_{\text{attack},p}} - 1)) & \text{if } t \leq t_{\text{attack},p}, \\ \exp(a_{60,p} t + a_{k,60,p} k) & \text{if } t > t_{\text{attack},p}, \\ 0 & \text{otherwise.} \end{cases} \tag{6.3}$$

$$a_{60,p} = \frac{\log(10^{-3})}{t_{60,p}} \tag{6.4}$$

$$a_{k,60,p} = \frac{\log(10^{-3})}{k_{60,p}} \tag{6.5}$$

The piecewise amplitude function is based on the amplitude function of the *Formant Wave Function (FOF)*[2] described in [42, p. 19].

The estimation of these parameters is a separate problem addressed by the DDM (see Section 3.4). We use theoretical values calculated directly from the model signals. For interpretation, and to make it possible to simply replace the theoretical values with those obtained from an analysis, we compute parameters that correspond to a model whose parameters could be estimated through a technique such as the DDM.

For this experiment we seek signals $s_k \in \mathbb{C}$ of the following form:

$$s_k(n) = \exp(\log(A_k) + \alpha_k n + j(\phi_k + \omega_k n + \frac{1}{2}\psi_k n^2)) \tag{6.6}$$

Here n is the sample number. Typically when performing a short-time analysis, the time corresponding to $n = 0$ is made to be the centre of the window, therefore, t is the time at the centre of the window and N_w, in samples, is the length of the middle (usually non-zero)

[1] `http://ccrma.stanford.edu/~jos/pasp/Dispersion_Filter_Design_I.html`

[2] FOF stands for *Forme d'Onde Formantique*.

portion of the window. The coefficents of the kth harmonic of the pth source from our synthetic model are given by

$$\alpha_{k,p}(t) = \frac{a_{60,p}}{f_s} \tag{6.7}$$

$$A_{k,p}(t) = \exp(a_{60,p}t + a_{k,60,p}k) \tag{6.8}$$

for the part of the signal after the attack portion.

For the attack portion, we estimate the parameters using least-squares on a rectangular-windowed signal. Let

$$\hat{\mathbf{s}}_{k,p}(t) = \begin{pmatrix} \exp\left(a_{60,p}\left(t - \frac{N_w}{2f_s}\right) + a_{k,60,p}k\right)\cos^2\left(\frac{\pi}{2}\left(\frac{t - \frac{N_w}{2f_s}}{t_{\text{attack},p}} - 1\right)\right) \\ \vdots \\ \exp\left(a_{60,p}\left(t + \frac{N_w}{2f_s}\right) + a_{k,60,p}k\right)\cos^2\left(\frac{\pi}{2}\left(\frac{t + \frac{N_w}{2f_s}}{t_{\text{attack},p}} - 1\right)\right) \end{pmatrix} \tag{6.9}$$

then $\log(A_{k,p})$ and $\alpha_{k,p}$ are found as the least-squares solution of

$$\begin{bmatrix} 1 & \frac{-N_w}{2} \\ \vdots & \vdots \\ 1 & \frac{N_w}{2} \end{bmatrix} \begin{pmatrix} \log(A_{k,p}(t)) \\ \alpha_{k,p}(t) \end{pmatrix} = \log \hat{\mathbf{s}}_{k,p}(t) \tag{6.10}$$

for the argument parameters (those multiplied by j in Equation (6.6))

$$\omega_{k,p}(t) = \frac{2\pi}{f_s}\left(f_{0,p} + A_{f,p}\sin\left(2\pi f_{f,p}t + \phi_{0,f,p}\right)\right)K_{B_p}(k) \tag{6.11}$$

$$\psi_{k,p}(t) = \left(\frac{2\pi}{f_s}\right)^2 A_{f,p}f_{f,p}\left(f_{0,p} + A_{f,p}\cos\left(2\pi f_{f,p}t + \phi_{0,f,p}\right)\right)K_{B_p}(k) \tag{6.12}$$

$$\phi_k(t) = \left(2\pi f_{0,p}t - \frac{A_{f,p}}{f_{f,p}}\cos\left(2\pi f_{f,p}t + \phi_{0,f,p}\right)\right)K_{B_p}(k) + \phi_{0,p} \tag{6.13}$$

To simulate the noise that would be present in an estimation of the signal parameters from an arbitrary signal, we create noise corrupted values by substituting the random variables:

- $\tilde{\psi}_{k,p}(t) \sim \mathcal{N}(\psi_{k,p}(t), \psi_{no})$

- $\tilde{\omega}_{k,p}(t) \sim \mathcal{N}(\omega_{k,p}(t), \omega_{no})$
- $\tilde{\alpha}_{k,p}(t) \sim \mathcal{N}(\alpha_{k,p}(t), \alpha_{no})$
- $\tilde{A}_{k,p}(t) \sim \mathcal{N}(A_{k,p}(t), A_{no})$

The θ_{no} (where θ is replaced by ω etc.) specifies the variance of the particular parameter. Most likely in practice these random variables would be correlated but not knowing the estimation method, we cannot at this point say anything about this correlation. Therefore the noisy parameters are uncorrelated random variables for this experiment.

We also add spurious data-points as a fraction r of the number of true data-points. Their values are drawn from uniform distributions with boundaries $\theta_{\min}$ and $\theta_{\max}$, where θ is some parameter above, e.g., $\omega_{\min}$ and $\omega_{\max}$ for the ω parameter. For this experiment $r = 0.25$. The parameters of the uniformly distributed random variables are given in Table 6.2. Data points are computed for the times $t = 0, \frac{H}{f_s}, \frac{2H}{f_s}, \ldots, \frac{\left\lfloor \frac{N}{H} \right\rfloor H}{f_s}$.

Table 6.2. Distribution parameters of uniformly distributed random variables

Parameter	$\theta_{\min}$	$\theta_{\max}$
ω	0	π
ψ	-1×10^{-4}	1×10^{-4}
α	-1×10^{-3}	1×10^{-3}

6.5 Computation of Principal Components

At each time t we have L data-points. As the source of each data-point is now unknown, we replace the k and p indices with index l. We only consider the amplitude and frequency modulation. According to our model, the frequency modulation is greater for harmonics of greater centre frequency. To take this into consideration, we divide the frequency modulation estimate $\psi_l(t)$ by the constant frequency estimate $\omega_l(t)$. This is similar to the approach taken in [4]. The amplitude modulation $\alpha_l(t)$ remains constant for all harmonics of the same source, only its initial value changes according to $k_{60,p}$. We compile the data-points

at one time into a set of observations.

$$\mathbf{x}_l(t) = \begin{pmatrix} \frac{\psi_l(t)}{\omega_l(t)} \\ \alpha_l \end{pmatrix} \tag{6.14}$$

$$\mathbf{X}(t) = \Big[\mathbf{x}_1(t) \dots \mathbf{x}_L(t)\Big] \tag{6.15}$$

From these L observations the correlation matrix $\mathbf{S}$ is computed. We use the correlation matrix because the values in each row of $\mathbf{x}_l(t)$ do not have the same units, see [21, p. 22] for a discussion about this.

Following the standard technique for producing principal components [21, p. 11], we obtain a matrix $\mathbf{V}(t)$ of eigenvectors sorted so that the eigenvector corresponding to the largest eigenvalue is in the first column, etc. The principal components $\mathbf{A}(t)$ are then computed as

$$\mathbf{A}(t) = \mathbf{V}^T(t)\mathbf{X}(t) \tag{6.16}$$

We have found it sufficient to use only the first principal component and therefore only use the values in the first row of $\mathbf{A}(t)$.

If we see the $\mathbf{x}_l(t)$ as realizations of a random variable, the above computation of principal components has the effect of projecting realizations of $\mathbf{x}_l(t)$ to points $a_{1,l}(t)$ on a 1-dimensional subspace. It is a fundamental theory of principal components that the transformation above maximizes the expected euclidean distance between the points $a_{1,l}(t)$. This is desirable for the current problem because it will always produce a variable emphasizing the parameter with the most variance. More specifically, if a scatter plot of the frequency-modulation measurements shows multiple distinct clusters whereas the amplitude-modulation measurements are all close and show only one cluster in a scatter plot, the first PC will emphasize the frequency-modulation measurements, which we desire for ease of clustering. The drawback of this approach is that if one parameter is very noisy and the other is not, the noisy parameter will be emphasized but forming informative clusters will be difficult. In that case it would be better to reject this parameter or use more PCs on which to perform clustering.

6.6 Preparing data for clustering

The EM underlying the Gaussian mixture model parameter estimation will only converge to a local maximum [6], therefore, for the best results, we compute a good initial guess and remove obvious outliers before carrying out the clustering algorithm.

The $a_{1,l}(t)$ are compiled into a histogram of N_b bins. The minimum and maximum bin boundaries are computed from the maximum and minimum values of $a_{1,l}(t)$ respectively. Values in a bin with less than τ_h other values are discarded. We find N_p contiguous sections of equal area in the new histogram omitting the discarded values. We use the centres of these sections as the initial mean guesses and half their width as the distance 3 standard deviations from the mean (roughly 99.7 percent of values drawn from one distribution will lie within this interval if they follow a normal distribution). The initial guesses for the weights are simply $\frac{1}{N_p}$.

6.7 Clustering

GMM parameter estimation is discussed in Section B. After convergence we have an estimated probability $p(a_{1,l}(t)$ from distribution $p)$. We choose the distribution p for each $a_{1,l}(t)$ that gives the highest probability of it having occured. The values $\mathbf{x}_t(t)$ corresponding to the $a_{1,l}(t)$ have this same classification. Those sharing the same classification can be interpreted as coming from the same source. The figure shows the results of the above steps carried out on a mixture of two sources synthesized with the parameters summarized in Table 6.3. The length of the signal N is 8000 samples and the anlaysis hop size H is 256 samples. The figures in 6.8 summarize the results of the source separation experiment.

6.8 Results

Here we show source separation results for the synthesized signals with varying amounts of noise added to the synthesis parameters ψ, ω, α and A. The amount of noise added and the corresponding plot title is summarized in Table 6.4.

[3] These are the fundamental frequencies of a c_4 and $c_4^\sharp$ respectively.

[4] These values are found by computing $f_{0,p}2^{1/24} - f_{0,p}$ giving a quarter-tone of frequency modulation centred around the fundamental frequency.

Parameter	Source 1 value	Source 2 value
$f_{0,p}$	261.63	277.18 [3]
K_p	20	20
$k_{60,p}$	20	20
B_p	0.001	0.001
$\phi_{0,p}$	0	0
$\phi_{0,f,p}$	0	0.8
$t_{60,p}$	0.5	0.75
$t_{\text{attack},p}$	0.1	0.1
$A_{f,p}$	7.666	8.1220 [4]
$f_{f,p}$	3	2

Table 6.3. Synthesis parameters for source separation by frequency and amplitude modulation.

Table number	ψ_{no}	ω_{no}	α_{no}	A_{no}
1	1×10^{-2}	1×10^{-2}	1×10^{-2}	1×10^{-4}
2	1×10^{-2}	1×10^{-2}	1×10^{-2}	1×10^{-5}
3	1×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-4}
4	1×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-5}

Table 6.4

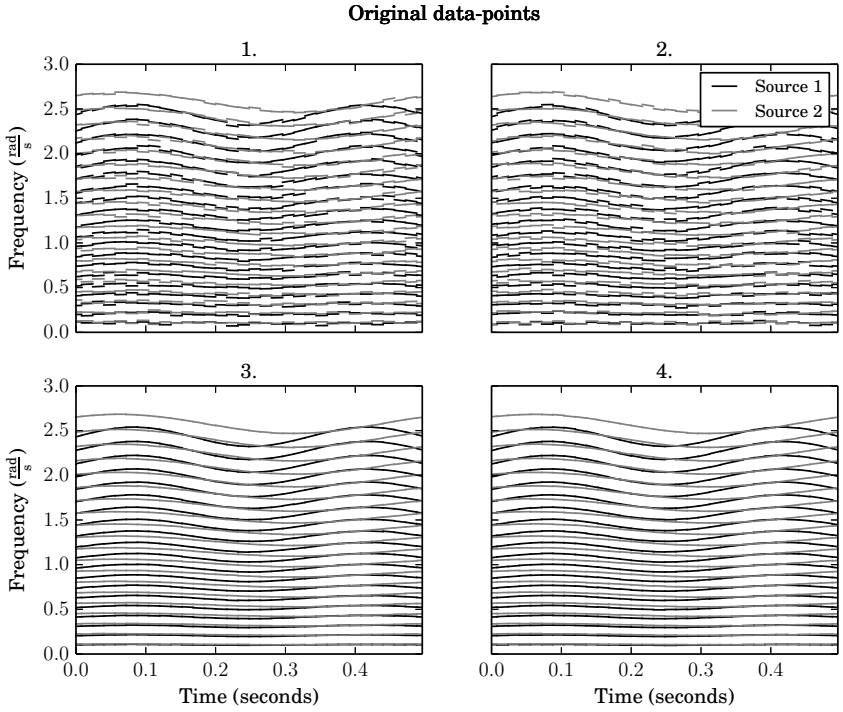

Fig. 6.1: **Original data-points.** Line-segments describing the frequency and frequency modulation of both sources.

As seen in Figure 6.4 and 6.5, while classified well in individual frames, the overall classification does not correspond to a single source. We must find a collection of frames with high plausibility of belonging to one source. We consider the collections of classified data-points corresponding to each source as a node in a lattice. Each frame of the lattice contains two nodes, one for each source. A best path through the lattice should connect together those nodes belonging to a single source. We use the results of Section 4.2 to find the two best paths through this lattice. We compare two distance metrics for the cost function.

The first prefers smoothness in frequency between two frames. For frame h with initial classification $\tilde{p}$ we have frequency measurements $\omega^h_{k,\tilde{p}}$ and frequency slope measurements $\psi^h_{k,\tilde{p}}$. The set of parameters at time h from initially classified source $\tilde{p}$ we will denote $\boldsymbol{\theta}^h_{\tilde{p}}$. Between frame h and frame $h+1$ we use Algorithm 1 on the pairs $\left\{\boldsymbol{\theta}^h_{\tilde{m}}, \boldsymbol{\theta}^{h+1}_{\tilde{n}}\right\}$ with $(m,n) \in \{0,1\} \times \{0,1\}$. For each pair, L is set to $\min(\#\boldsymbol{\theta}^h_{\tilde{m}}, \#\boldsymbol{\theta}^{h+1}_{\tilde{n}})$.[5] The cost function

[5]Here, the threshold parameter $\Delta = \infty$, i.e., a connection of any cost is possible.

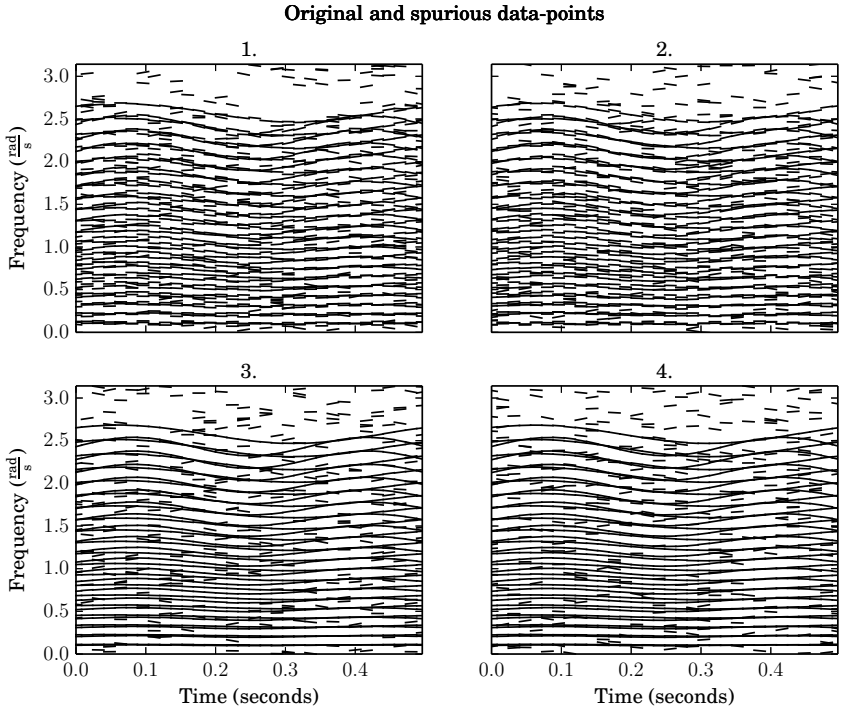

Fig. 6.2: **Original and spurious data-points.** Line-segments describing the frequency and frequency modulation of the original data and the spurious data.

is the absolute error in predicting the frequency in the next frame from parameters in the current frame, i.e.,

$$\mathcal{D}_f\left(\theta^h_{i,\tilde{m}}, \theta^{h+1}_{j,\tilde{n}}\right) = |\omega^h_{i,\tilde{m}} + \psi^h_{i,\tilde{m}} H - \omega^{h+1}_{j,\tilde{n}}|$$

where H is the hop-size in samples between the two frames. The second distance metric measures the smoothness in amplitude between two frames by predicting the next frame's amplitude parameters using the amplitude and amplitude-modulation parameters of the current frame. It is given as

$$\mathcal{D}_a\left(\theta^h_{i,\tilde{m}}, \theta^{h+1}_{j,\tilde{n}}\right) = |\log(A^h_{i,\tilde{m}}) + \psi^h_{i,\tilde{m}} H - \log(A^{h+1}_{j,\tilde{n}})|$$

We have found the absolute error to give better results than the squared error.

The costs of these connections are summed over the index pairs Γ_h to give the entries

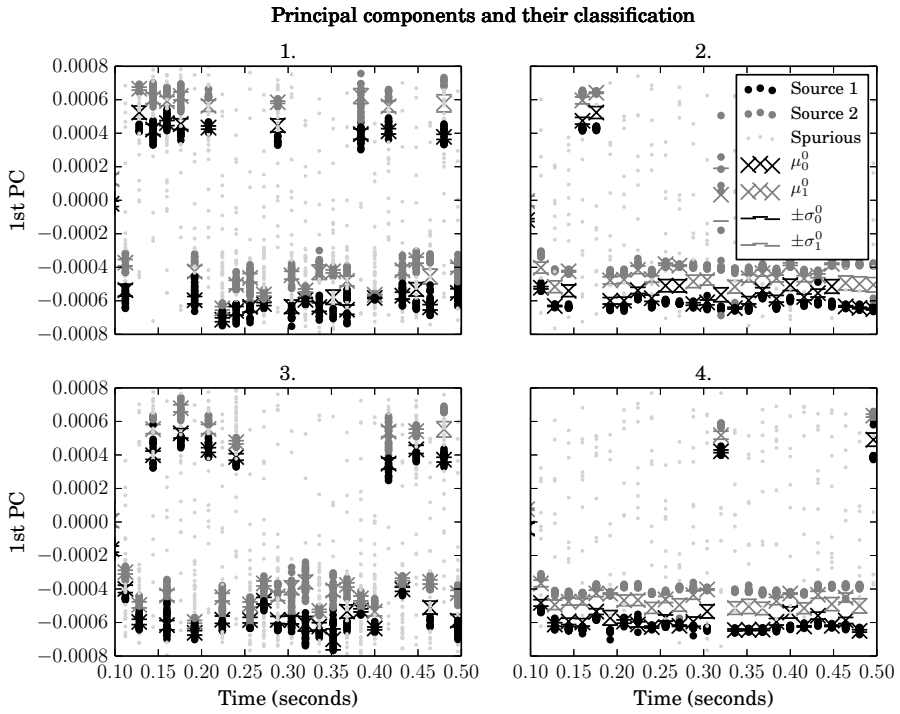

Fig. 6.3: **Principal components and their classification.** The PCs for each theoretical analysis time-point. μ_i^0 and σ_i^0 are respectively the initial mean and standard deviation guesses for the EM algorithm fitting the GMM parameters to the ith source. These values are also visible on the plot. The spurious points rejected using the process described in Section 6.6 are included for comparison. The plot is of a subsection of the analysis to better show the process.

of the cost vector $\boldsymbol{c}$ in the LP

$$\boldsymbol{c}_{4t+2m+n} = \sum_{i^*,j^* \in \Gamma_h} \mathcal{D}\left(\theta^h_{i^*,\tilde{m}}, \theta^{h+1}_{j^*,\tilde{n}}\right)$$

The specification of the constraint matrices is done according to the topology of the lattice and the requirement that we find 2 non-overlapping paths (see Section 4.2). An example of discovered paths is given in Figure 6.6. The estimated sources after smoothing in frequency are shown in Figures 6.7 and 6.8. The estimated sources after smoothing in amplitude are shown in Figures 6.9 and 6.10. We see that when smoothed in frequency, the results are acceptable. However, when both sets of parameters are close and give close costs, the spurious data-points can influence the cost function causing a false classification.

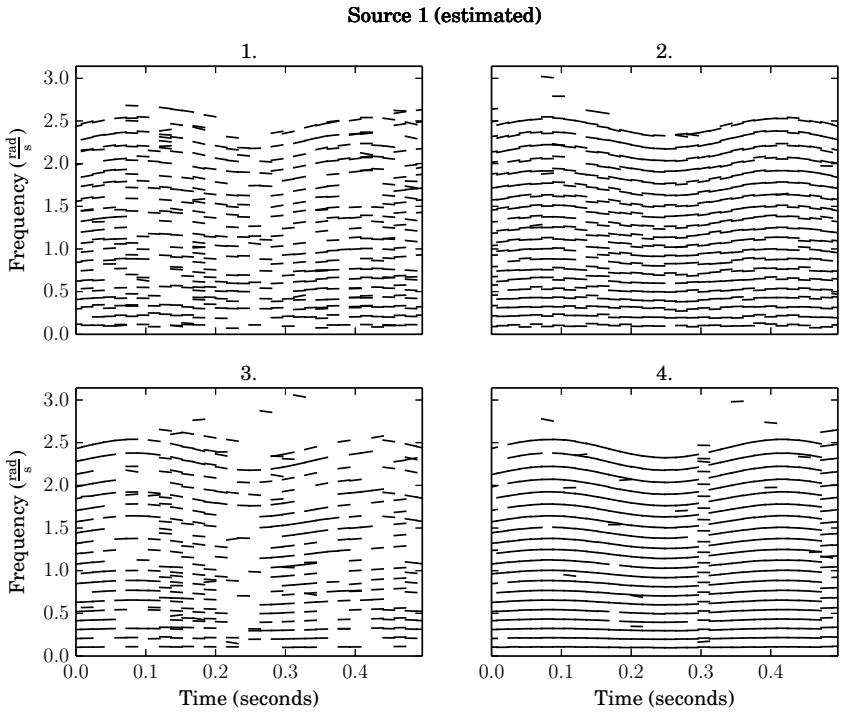

Fig. 6.4: **Source 1 (estimated).** Line-segments classified as belonging to Source 1. The classification is done on each frame so classifications in consecutive frames may not belong to the same true source. This is because the ordering of the clusters in each frame in Figure 6.3 is not predictible.

This difficulty is not surprising, looking at Figure 6.1 we see that there are some segments where the frequency slopes are close.

When smoothed in amplitude, the results are less convincing. This is not surprsing as smoothness in amplitude is not the best criterion at all time points. In Figure 6.11 we see that the amplitudes of both sources are similar at many points, e.g., at around 0.05 and 0.15 seconds.

6.9 Conclusion

In this chapter we evaluated the plausibility of separating two mixed sources by classifying based on their theoretical frequency- and amplitude-modulation. We obtained acceptable results for signals with small measurement errors. The method is also robust in the presence of spurious data points. A shortcoming of the method is the requirement that the frequency

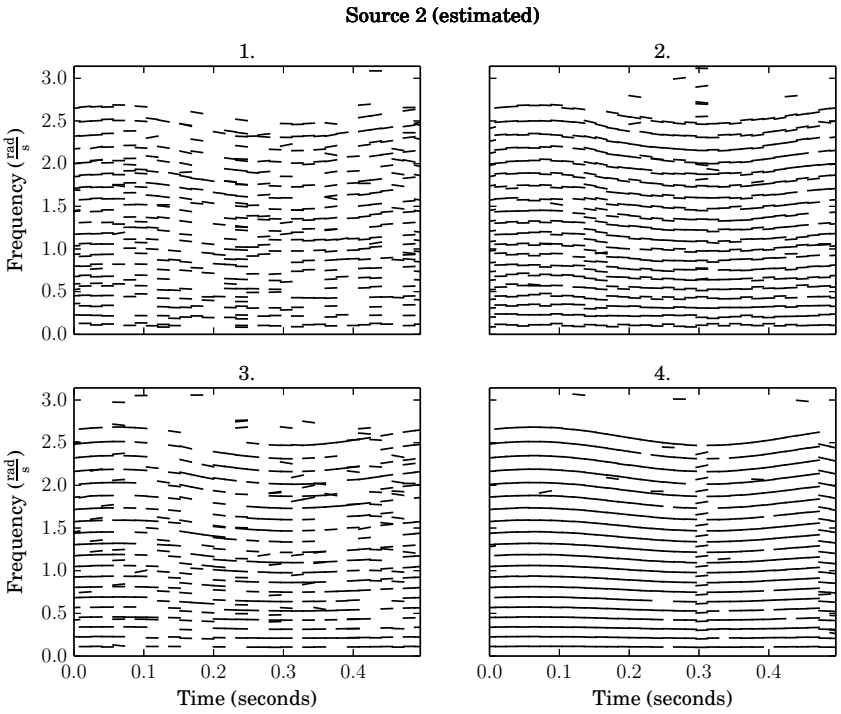

Fig. 6.5: **Source 2 (estimated).** Line-segments classified as belonging to Source 2. See Figure 6.5 for more information.

and frequency-modulation of the signals be known. If the signals are sufficiently separated in frequency and have small bandwidth, as shown in Section 3.4 the DDM can be used to estimate these parameters. If signals are close in frequency, the number of sinusoids is known, and these exhibit slow modulations, signal subspace methods could be used [44] where the estimations at different time points are connected as in Section **??** and the modulation parameters postulated via interpolation similarly to Section 3.3.1. Another strategy might be to use two uncorrupted measurements of one source and extrapolate the parameters of the signal in the part corrupted by the other. Another shortcoming of the technique presented here is the use of the costly EM algorithm to classify data points using GMM. A more ad hoc approach could be taken to save on these computations, perhaps partitioning the data sets using local minima as illustrated in Figure 6.12. In any case, the source separation technique presented here, being interative, is of a complexity similar to NMF or PLCA but can also resolve the phases of the sinusoids which are discarded in most

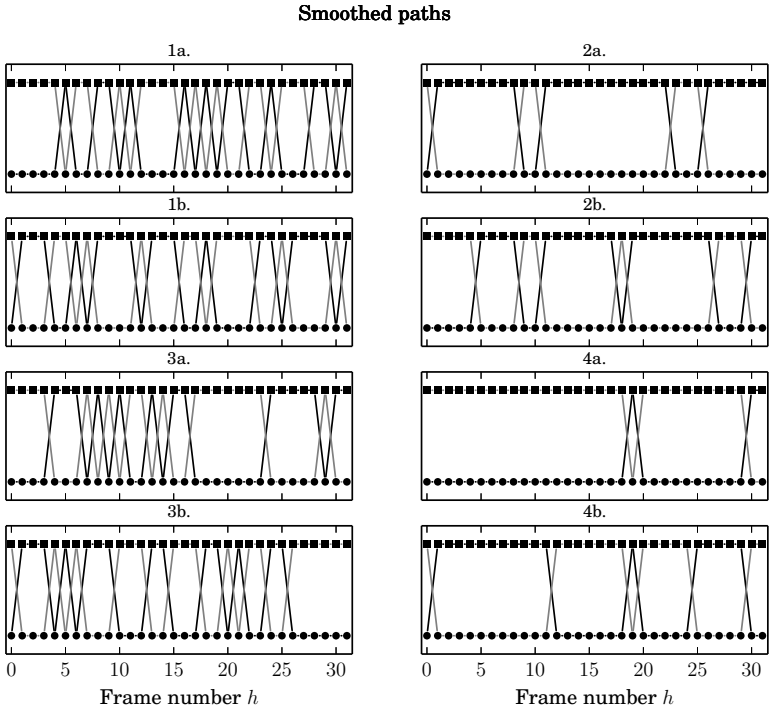

Fig. 6.6: **Smoothed paths.** The points originally classified as source 1 are marked with circles and those originally classified as source 2 squares. The paths in black or grey connect the points for source 1 or source 2 respectively with optimal smoothness. Plots indicated with an a are the paths with frequency smoothness as the criterion and those indicated with b are with amplitude smoothness as the citerion.

NMF or PLCA implementations[6].

[6]See [3] for an approach that does take into consideration the phase information in the spectrogram.

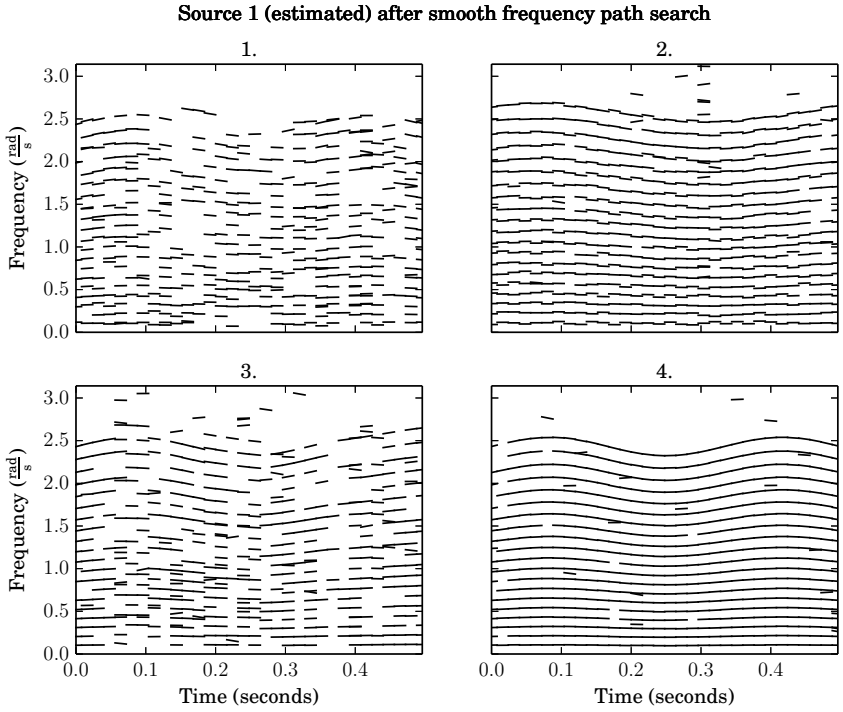

Fig. 6.7: **Source 1 (estimated) after smooth frequency path search.** Line-segments classified as belonging to source 1 after smoothing in frequency.

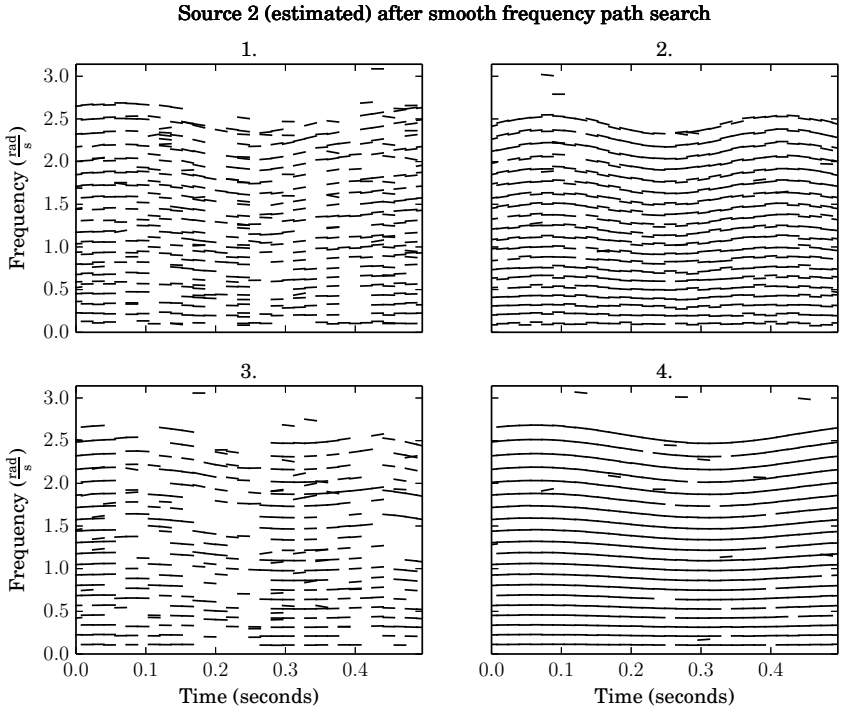

Fig. 6.8: **Source 2 (estimated) after smooth frequency path search.** Line-segments classified as belonging to source 2 after smoothing in frequency.

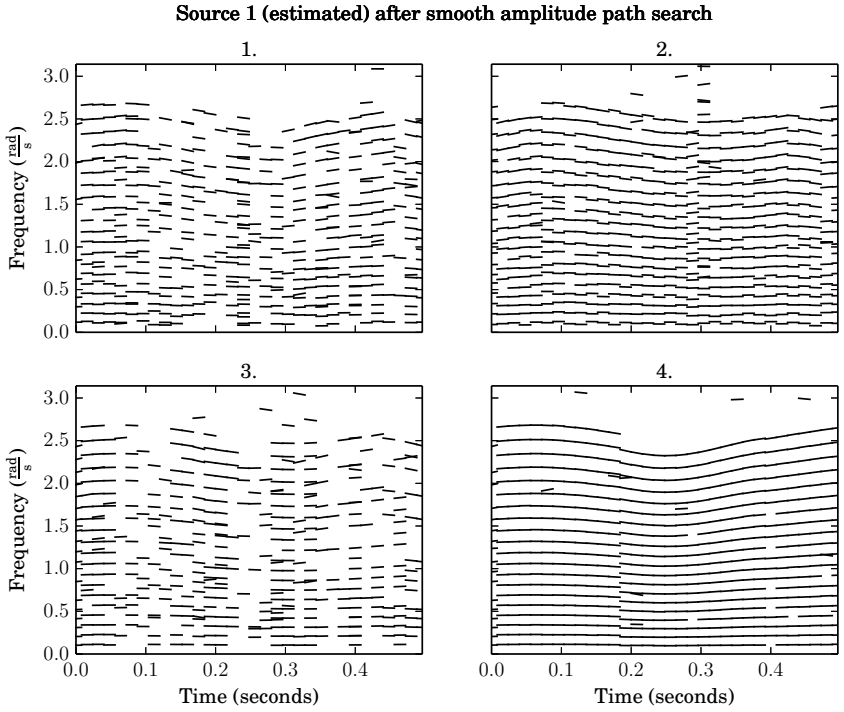

Fig. 6.9: **Source 1 (estimated) after smooth amplitude path search.** Line-segments classified as belonging to source 1 after smoothing in amplitude.

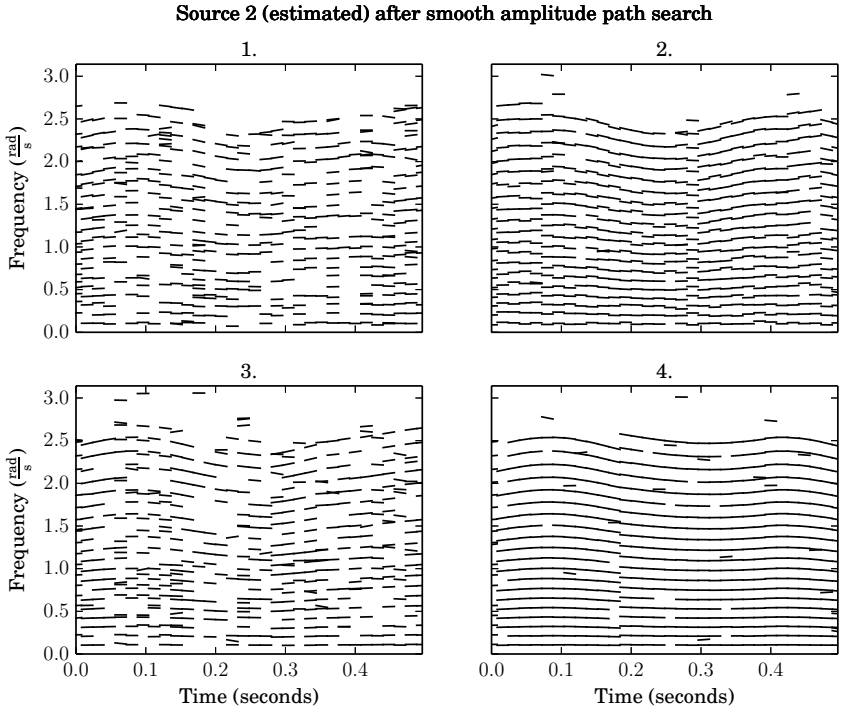

Fig. 6.10: **Source 2 (estimated) after smooth amplitude path search.** Line-segments classified as belonging to source 2 after smoothing in amplitude.

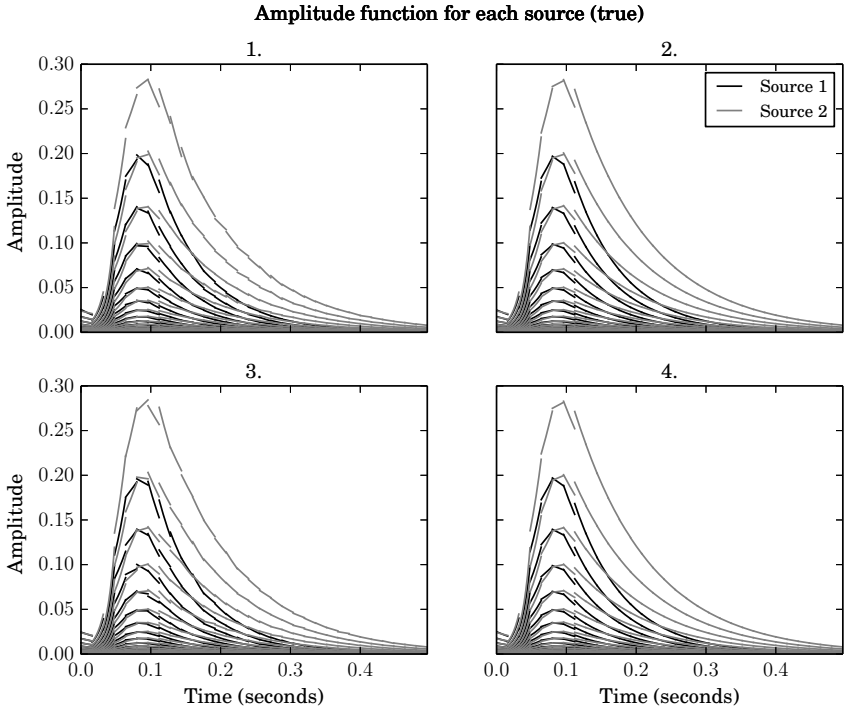

Fig. 6.11: **Amplitude function for each source (true).** Line-segments representing the instantaneous amplitude and amplitude slope at analysis time points.

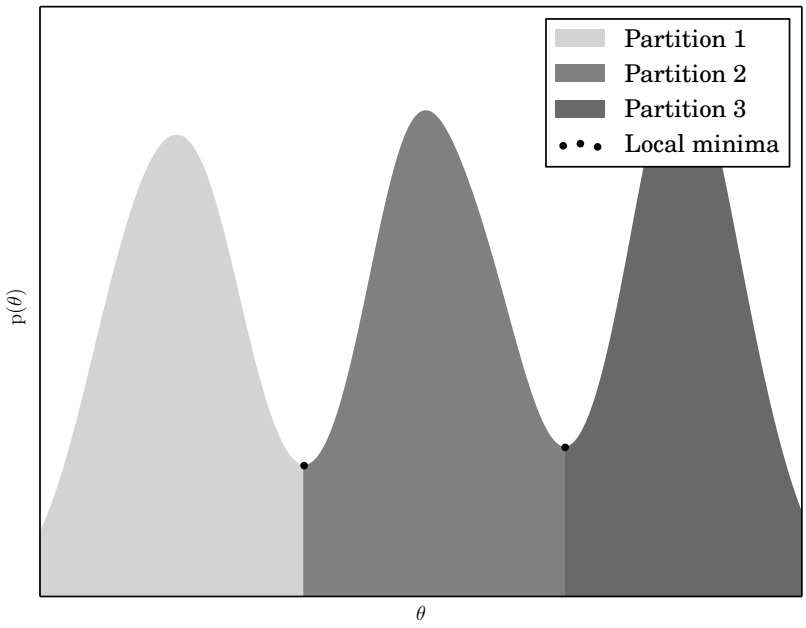

Fig. 6.12: **Paritioning example.** The data points are convolved with some smoothing kernels giving a function with a small number of extrema. The minima, indicated by circles, are used as boundaries between the partitions which are illustrated with different shades of grey.

Chapter 7

Experiment: Separation of two sources using partial decay rate

In this section we demonstrate how the techniques described above can be used to perform audio source separation on signals obtained from recordings of acoustic instruments. We start with a recording of an acoustic guitar playing a_3 and a xylophone playing $f_4^\sharp$. The recordings are from [36] and have been mixed down to one channel (by simply adding the two signals together) and resampled at 16 KHz, coded simply as a stream of 64-bit floating-point numbers. Spectrograms of the original signals are shown in Figure 7.4 and Figure 7.5. The spectrograms were produced with a Hann window, DFT size of 4096 samples and a hop size of 512 samples.

The mixture of the two signals was analysed using the DDM for finding the coefficients of a cubic complex phase polynomial. Local maxima in each frame were found using the technique described in [46, p. 42]. For each of these local maxima, the polynomial coefficients were estimated. The analysis used the $\mathcal{C}^1$ 4-Term Blackman-Harris window. To obtain partials it was then necessary to connect the local maxima. As the partials of these two sound sources are quite stable in frequency it sufficed to use the Viterbi algorithm [10] to connect local maxima in sub-bands of the spectrum. The cost function is simply the Euclidean distance between the frequencies of two local maxima. Partial starting points are considered in sub-bands of width 15 Hz and these sub-bands overlap by 7.5 Hz. A partial path starts on the first local maximum in the band exceeding -100 dB and ends at the last maximum exceeding -100 dB. The path search algorithm will also look ahead to

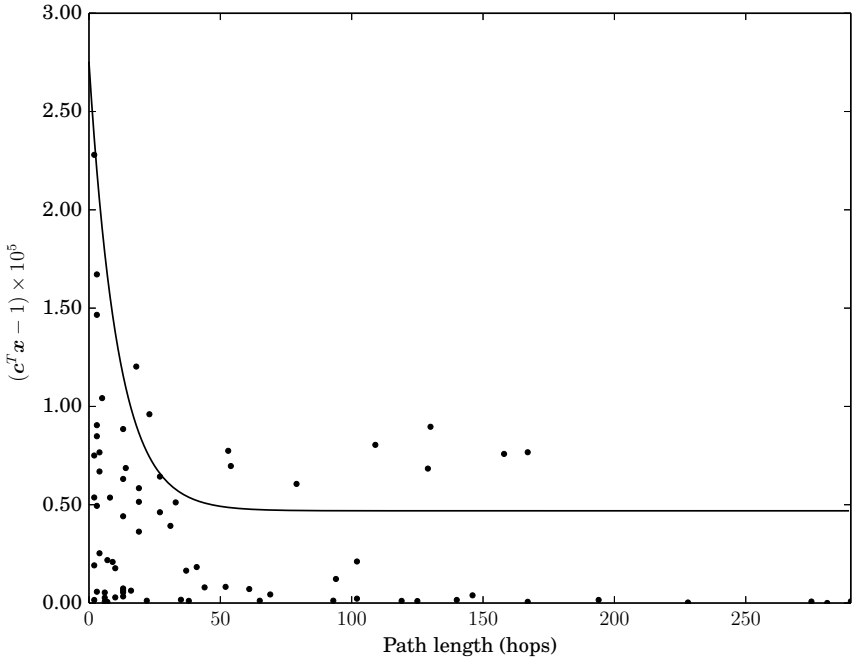

Fig. 7.1: **Path cost vs. length and thresholding boundary.** Paths (represented as circles) are considered only if their path cost is smaller than the threshold function (represented in black).

further frames if no maximum is present in the next frame. Because of this, sometimes unrealistic paths are discovered that jump between spurious maxima. These are filtered out by discarding paths whose cost-length ratio is excessive. See Figure 7.1 to see a plot of these values and the thresholding function. The spectrogram of the mixture can be seen in Figure 7.2 and the partial paths in Figure 7.3.

A line function is fit via least-squares to each partial, as shown in Figure 7.6. Our goal is to classify based on the amplitude modulation of each partial, or to an approximation, the slope of these line-functions. We found that examining the log-length of the partials gives better results than examining the slope directly. This is perhaps because the log-length encodes both the starting amplitude and the slope—recall that the partials start on the first local maxium exceeding an amplitude threshold. To motivate the use of this metric, we plot the log-length vs. the starting frequency of partials from separate analyses of the guitar and xlyophone signals and overlay the plots (Figure 7.7). The data-points have the

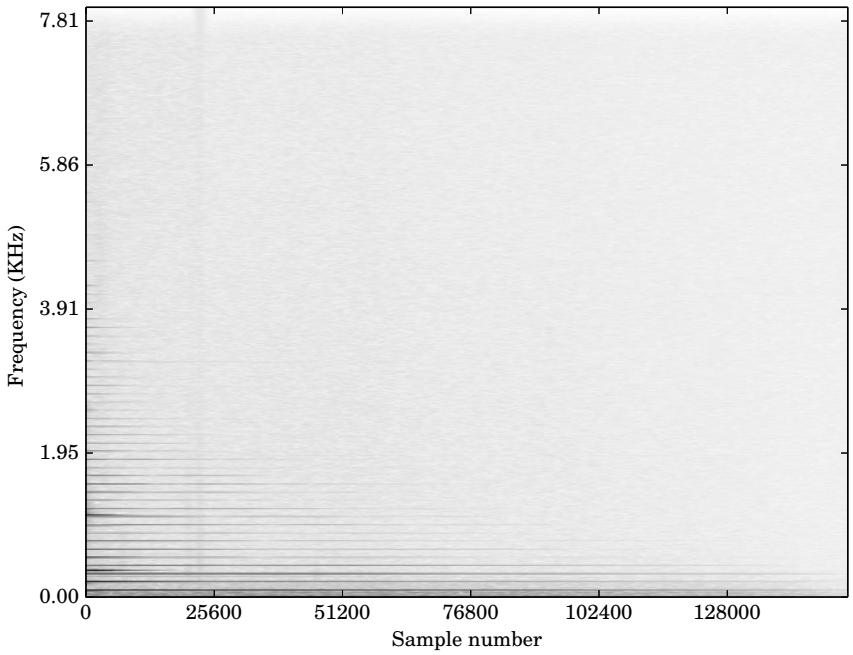

Fig. 7.2: **Spectrogram of mixture**

form

$$\boldsymbol{a}_i = \begin{pmatrix} a_{i,0} \\ a_{i,1} \end{pmatrix}$$

where $a_{i,0}$ is the first PC and $a_{i,1}$ the second. The set of PC data points will be denoted $\{\boldsymbol{a}\}$. We see that, for the most part, the partials belonging to the two sources are separated appropriately into two clusters. The partials from the xylophone present in the guitar cluster belong to higher partials, whose omission in the final rendering of the xylophone source would not be detrimental to its perceptual quality. Similarly, partials belonging to the guitar present in the xylophone cluster are short and most likely belong to briefly excited modes of the guitar body. Our intention is now to use GMM on a set of unclassified partials to yield a plausible source separation.

GMM fitting is sensitive to its initial guess of the parameters as the algorithm only finds a local maximum of the likelihood function [23, p. 187]. To find an initial guess we convolve the scatter plot with Gaussian kernels, giving a continuous function. We use the two local maxima of this function as the initial means for the two sought classifying

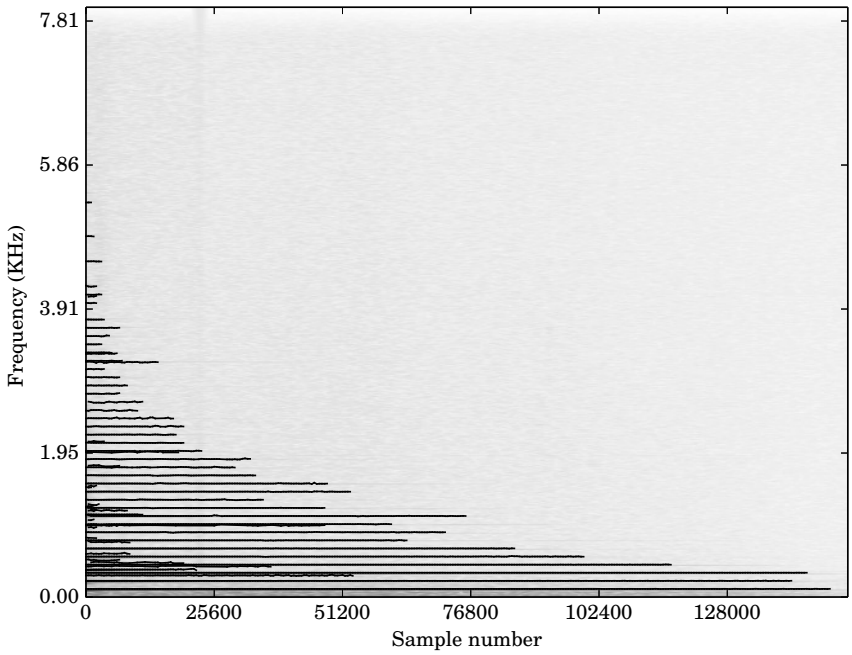

Fig. 7.3: **Spectrogram and partial trajectories of mixture**

Gaussian distributions. The convolution function evalutated at $\hat{\boldsymbol{a}}$ is

$$f(\hat{\boldsymbol{a}}) = \sum_{\boldsymbol{a}_i \in \{\boldsymbol{a}\}} \mathcal{N}\left(\hat{\boldsymbol{a}}; \boldsymbol{a}_i, \boldsymbol{\Sigma}_{\boldsymbol{a}}\right)$$

(see Appendix **??** for the definition of $\mathcal{N}$). To make the variance proportial to the extent of each dimension, $\boldsymbol{\Sigma}_{\boldsymbol{a}}$ is defined as

$$\boldsymbol{\Sigma}_{\boldsymbol{a}} = \begin{pmatrix} \frac{\Delta_{\boldsymbol{a}_1}}{\Delta_{\boldsymbol{a}_0 + \Delta_{\boldsymbol{a}_1}}} \theta_{\boldsymbol{\Sigma}_{\boldsymbol{a}}} & 0 \\ 0 & \frac{\Delta_{\boldsymbol{a}_0}}{\Delta_{\boldsymbol{a}_0 + \Delta_{\boldsymbol{a}_1}}} \theta_{\boldsymbol{\Sigma}_{\boldsymbol{a}}} \end{pmatrix}$$

where

$$\Delta_{\boldsymbol{a}_0} = \max\left(\boldsymbol{a}_0\right) - \min\left(\boldsymbol{a}_0\right)$$

$$\Delta_{\boldsymbol{a}_1} = \max\left(\boldsymbol{a}_1\right) - \min\left(\boldsymbol{a}_1\right)$$

and $\theta_{\boldsymbol{\Sigma}_{\boldsymbol{a}}}$ is a parameter to control the smoothness of the resulting function, here $\theta_{\boldsymbol{\Sigma}_{\boldsymbol{a}}} = 1.2$.

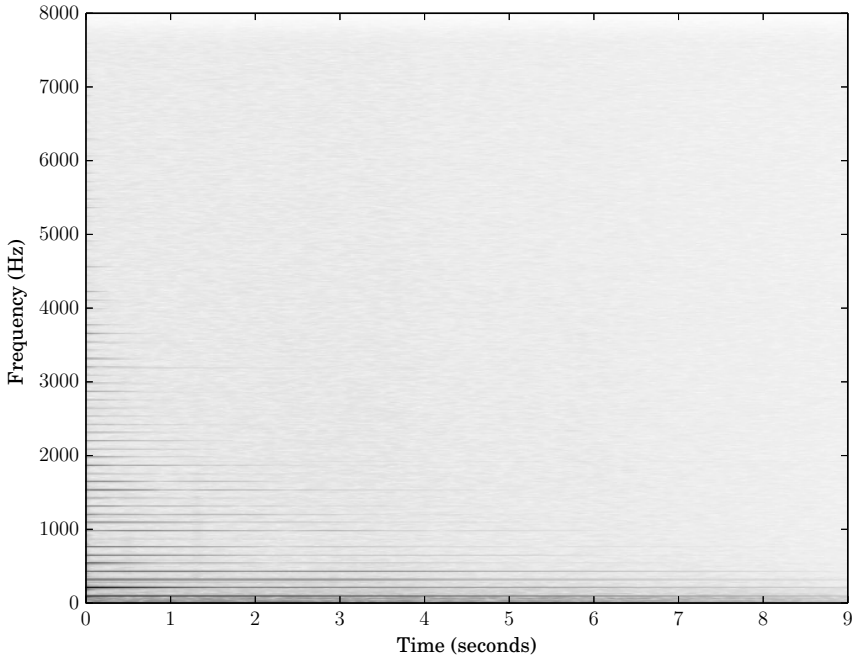

Fig. 7.4: **Spectrogram of acoustic guitar**

A contour plot of the resulting function is shown in Figure 7.8.

To initialize GMM the initial means $\boldsymbol{\mu}^0$ are chosen to be the points corresponding to the local maxima of the smoothed scatter plot. To determine initial weights $\boldsymbol{w}^0$ we first determine the value of the function at the two local maxima, $f(\boldsymbol{a}_0^*)$ and $f(\boldsymbol{a}_1^*)$. To weight relative to these two values, we compute

$$w_i^0 = \frac{\Theta_w \left\{ f(\boldsymbol{a}_i^*) \right\}}{\sum_{p=0}^{P-1} \Theta_w \left\{ f(\boldsymbol{a}_p^* \right\}}$$

where Θ_w is some kind of weighting operator to have parametric control over the influence of each function value. Here

$$\Theta_w \left\{ f(\boldsymbol{a}_i^*) \right\} = \begin{cases} f(\boldsymbol{a}_i^*)\theta_w & i = 0 \\ f(\boldsymbol{a}_i^*) & \text{otherwise} \end{cases}$$

and $P = 2$. For this experiment the parameter set as $\theta_w = 1.1$ gave the best results. The

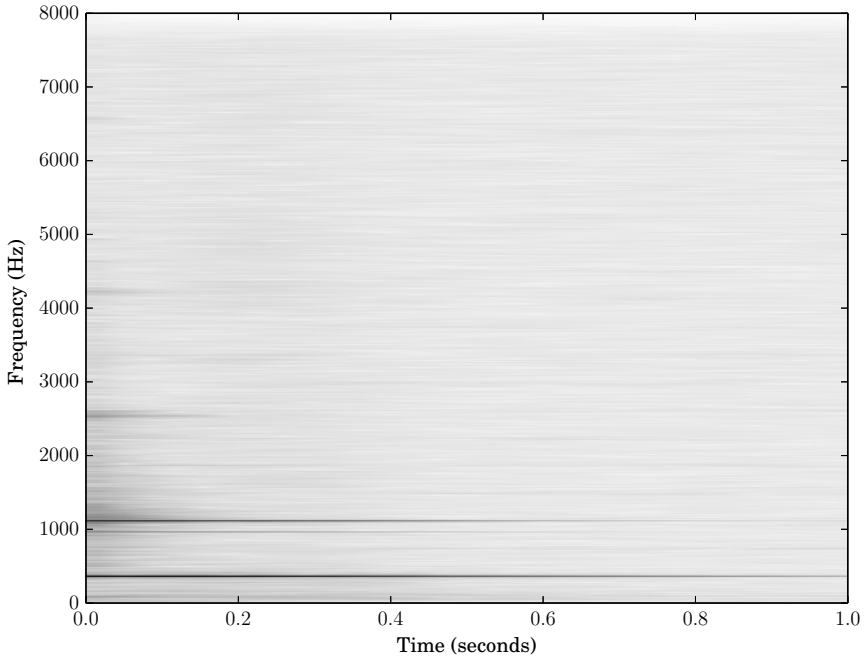

Fig. 7.5: **Spectrogram of xylophone**

covariance matrix $\mathbf{\Sigma^0}$ is computed as

$$\mathbf{\Sigma^0} = \boldsymbol{S}(\{\boldsymbol{a}\}) + \boldsymbol{\epsilon}$$

where $\boldsymbol{S}$ computes the sample covariance and $\boldsymbol{\epsilon}$ is some small constant to avoid a singular initial covariance matrix. 100 iterations of the EM algorithm are performed to compute classifications. Each point is assigned to its most likely cluster using the final estimated Gaussian distributions. The final classifications for this classification task can be seen in Figure 7.8.

After the classifications have been made, synthesizing the separated sources simply involves only synthesizing the partials classified as belonging to the same source. For the synthesis, we use the technique described in Section **??**. Spectrograms of the source separated signals are shown in Figure 7.9 and Figure 7.10.

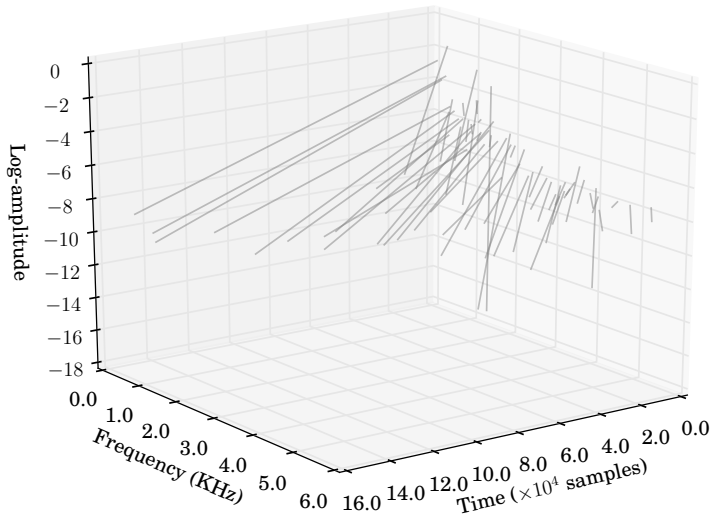

Fig. 7.6

7.0.1 Conclusion

After an informal listening, the source separation is perceptually convincing.[1] At least one partial from the guitar can be heard in the xylophone recording, however — it is difficult to separate partials that do not have sufficient spacial separation in Figure 7.8. Another drawback of the current technique is that it requires some tuning of the parameters θ_w and $\theta_{\mathbf{\Sigma}_a}$. From the spectrograms of the resynthesized sources, we see that some of the partials from both sounds were lost in the analysis. Although a shortcoming of the analysis rather than the classification, if partials are not sufficiently separated in time or frequency, they cannot be separated as their analysis will yield simply one partial when there in fact many. In any case, it is important to see that source separation can be carried out by only considering the amplitude modulation (in this case, the decay rate) in relation to the partial frequency.

[1] Soundfiles can be downloaded from
https://drive.google.com/file/d/0B8B4c04j8tBwZDFraEZ1dFZHRFU/view?usp=sharing

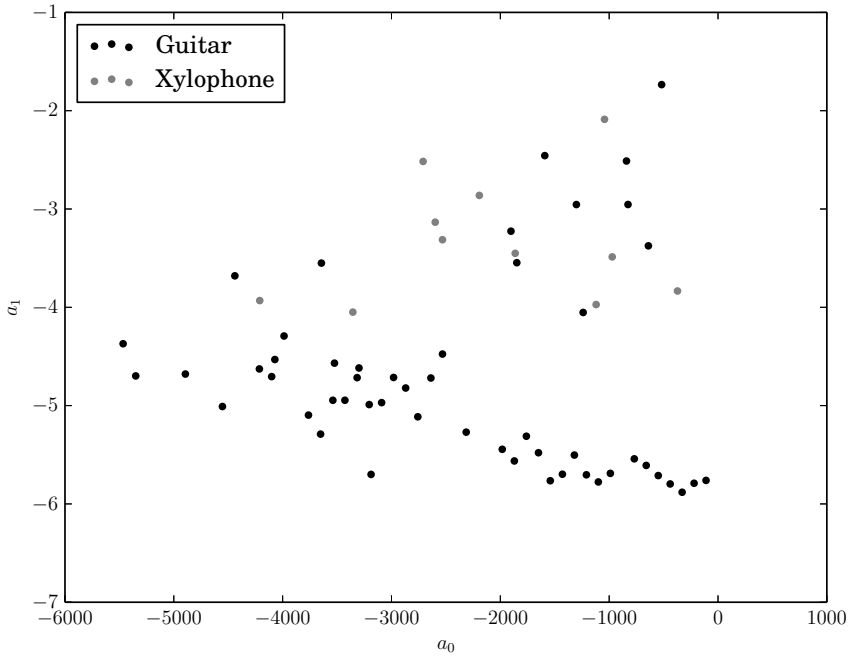

Fig. 7.7: **Log-partial-length vs. frequency: principal components.** This shows the distribution of partials when plotting their two PCs derived from their mean frequency and log-length. In this case, the source memberships of the partials are known. We see that there is generally a separation of the partials into two clusters corresponding to the two sources.

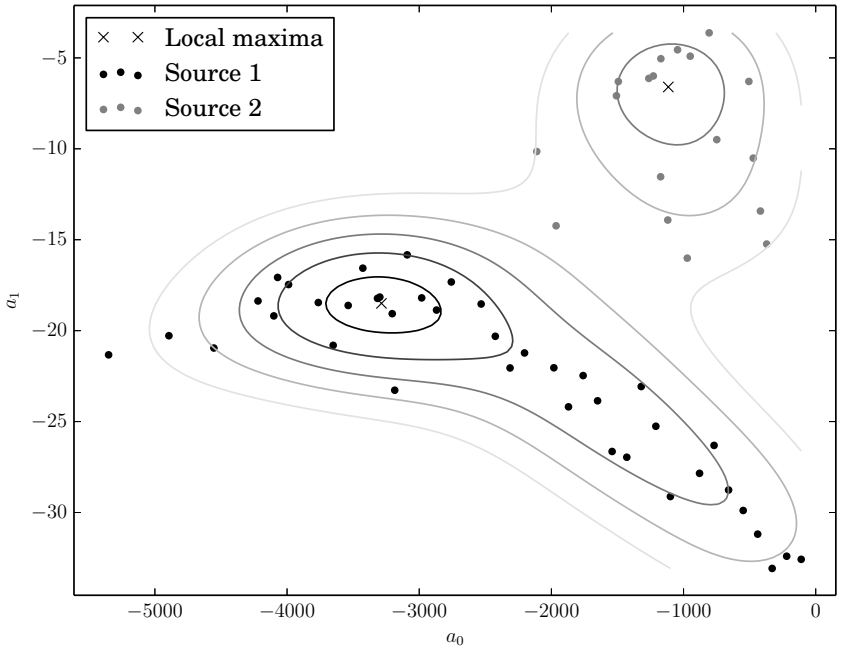

Fig. 7.8: **Estimated memberships**

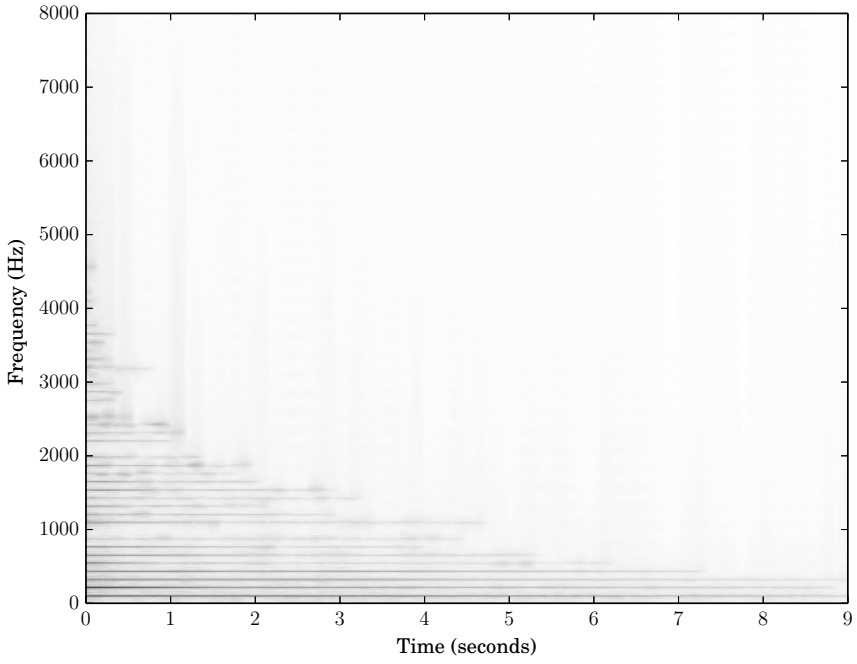

Fig. 7.9: **Spectrogram of source separated acoustic guitar**

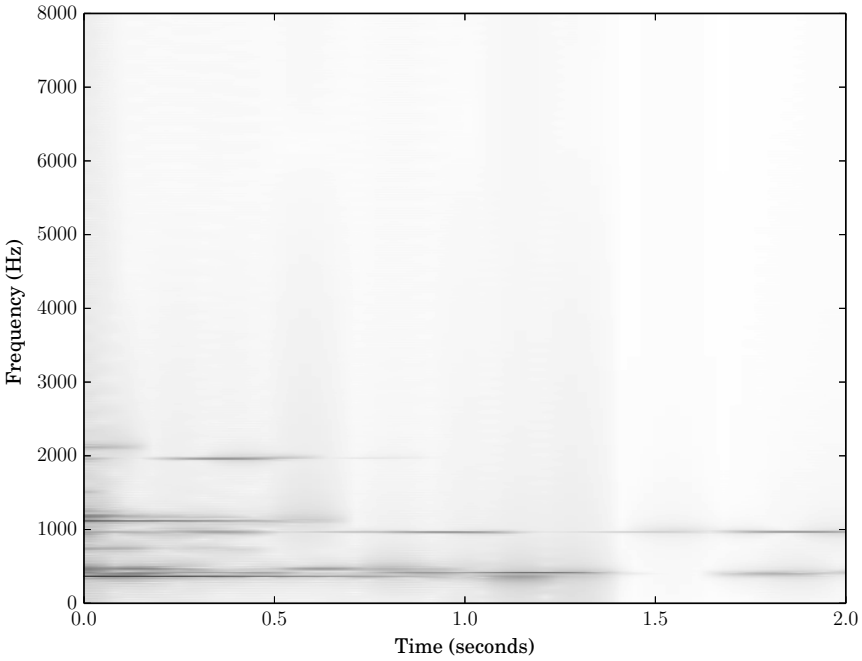

Fig. 7.10: **Spectrogram of source separated xylophone**

Appendices

Appendix A

Principal components analysis (PCA)

A.1 Motivation

If data-points consist of more than two dimensions (say p), it becomes burdensome to try and find the single best or two best dimensions on which to examine for grouping. If we consider variables on each of the dimensions that take on the data-point's corresponding values, we are interested in the variables that capture most of the data-points's variance. In turns out we can determine a linear transformation of our original dataset giving p variables and their p variances such that the resulting variable with the highest variance will have the maximum variance acheivable, under some constraints that will be explained shortly.

A.2 Computation of principal components

The following development is based on [21]. Say we have a set $\{\boldsymbol{x}\}$ of data-points and their covariance matrix $\boldsymbol{S}$. A linear function of $\boldsymbol{x}$, $f_1(\boldsymbol{x}) = \boldsymbol{a}_1^T\boldsymbol{x}$ has variance $\sigma_{\boldsymbol{a}_1^T\boldsymbol{x}} = \boldsymbol{a}_1^T\boldsymbol{S}\boldsymbol{a}_1$. Therefore, we desire a vector a that maximizes $\sigma_{\boldsymbol{a}_1^T\boldsymbol{x}}$. We can find this via the program

$$\max \boldsymbol{a}_1^T\boldsymbol{S}\boldsymbol{a}_1$$

subject to

$$\boldsymbol{a}_1^T\boldsymbol{S}\boldsymbol{a}_1 = 1$$

(to obtain a bounded solution).

The resulting function of $\boldsymbol{x}$, $f_1(\boldsymbol{x}) = \boldsymbol{a}_1^{*T}\boldsymbol{x}$ is called the first *principal component*.

The second principal component $f_2(\boldsymbol{x}) = \boldsymbol{a}_2^{*T}\boldsymbol{x}$ is found similary to the first, except with the additional constraint that it be uncorrelated (orthogonal) to the first component, i.e., $\boldsymbol{a}_2^{*T}\boldsymbol{a}_1 = 0$, and the third is found by requiring orthognality with the first two principal components, etc.

The principal components (PCs) now allow us to examine for grouping more easily as the total variance of the dataset has been captured in the first few principal component variables. These transformed data-points can now be classified using a classification algorithm.

Before we continue describing classification techniques we will briefly discuss the nature of our classification problem. If accurate and consistent measurements of data-points can be made, and a large enough sample of data-points is available to train a model, then to predict the classification of new points, a function $g = \hat{h}(\boldsymbol{x})$ is postulated, giving the classification g of a data-point $\boldsymbol{x}$. For example, if there are only two classes, we might postulate a linear classifier

$$\hat{h}(\boldsymbol{x}) = \begin{cases} 0 \text{ if } \boldsymbol{\beta}^T\boldsymbol{x} > c \\ 1 \text{ if } \boldsymbol{\beta}^T\boldsymbol{x} < c \end{cases}$$

where 0 and 1 indicate membership in class 0 or 1. Techniques for choosing $\hat{h}$ are discussed in **??** and often require data on which to "train" a classifier, i.e., determine a function $\hat{h}$ that classifies well a dataset with known classifications. Here we do not have a sample dataset because of the large number of possible situations and the difficulty of consistently estimating underlying model parameters (see, for example, Section 4.3). As training to find a suitable $\hat{h}$ is not possible, we propose *kernels* that reflect the hypothesized underlying structure of the distribution of $\boldsymbol{x}$. To elaborate, we may observe that the classification of a point is indicated by its nearness to other points in the same class, i.e., its membership to a cluster. The proposed kernel is a parameterized model of a cluster, whose parameters we can adjust until they fit the observed data well. We then choose the kernel that best explains the data to indicate what class these data are in. The following section describes a technique using normal distributions as kernels, which is the technique used in later experiments.

Appendix B

Gaussian Mixture Models (GMM)

Consider the data-points $\boldsymbol{X}$ as realizations of the vector Gaussian distributed random variable X. With a large enough sample and a small enough covariance, we will observe realizations of X as a cluster with some mean (centre point) $\boldsymbol{\mu}$ and a shape described by the covariance matrix $\boldsymbol{\Sigma}$. If we observe multiple clusters this might imply that there are P different distributions each with mean $\boldsymbol{\mu}_p$ and covariance matrix $\boldsymbol{\Sigma}_p$ and on each iteration one is chosen with probability w_p. With N observations $\boldsymbol{x}_n$ we can estimate, via maximum likelihood, the P sets of parameters using a form of the *expectation maximization (EM)* algorithm [33], [6], which is an algorithm suitable for estimating missing data from known ones. First, define

$$\mathrm{p}\left(\boldsymbol{x}_n|p\right) = \frac{\mathcal{N}\left(\boldsymbol{x}_n; \boldsymbol{\mu}_p^k, \boldsymbol{\Sigma}_p^k\right) w_p^k}{\sum_{l=1}^{P} \mathcal{N}\left(\boldsymbol{x}_n; \boldsymbol{\mu}_l^k, \boldsymbol{\Sigma}_l^k\right) w_l^k}$$

the probability that $\boldsymbol{x}_n$ given distribution p (see Appendix **??** for the definition of $\mathcal{N}\left(\boldsymbol{x}; \boldsymbol{\mu}^k, \boldsymbol{\Sigma}^k\right)$). The superscript k indicates the value of this parameter on iteration k. To update w_p^k:

$$w_p^{k+1} = \frac{1}{N} \sum_{n=1}^{N} \mathrm{p}\left(\boldsymbol{x}_n|p\right)$$

which means intuitively that the probability of a data-point having been generated by distribution p is the average probability of observing any $\boldsymbol{x}_n$ given p. To update $\boldsymbol{\mu}_p^{k+1}$:

$$\boldsymbol{\mu}_p^{k+1} = \frac{\sum_{n=1}^{N} \mathrm{p}\left(\boldsymbol{x}_n|p\right) \boldsymbol{x}_n}{\sum_{n=1}^{N} \mathrm{p}\left(\boldsymbol{x}_n|p\right)}$$

which is a weighted mean of all the data-points. Those less likely for a given p will weight the mean less and vice versa. A similar computation is made for $\boldsymbol{\Sigma}_p^{k+1}$:

$$\boldsymbol{\Sigma}_p^{k+1} = \frac{\sum_{n=1}^{N} \mathrm{p}\left(\boldsymbol{x}_n|p\right) \left(\boldsymbol{x}_n - \boldsymbol{\mu}_p^{k+1}\right) \left(\boldsymbol{x}_n - \boldsymbol{\mu}_p^{k+1}\right)^T}{\sum_{n=1}^{N} \mathrm{p}\left(\boldsymbol{x}_n|p\right)}$$

The algorithm is halted after some number of iterations or when convergence is reached, i.e., the parameters change little each iteration. After convergence, the classification p^* of the data-point $\boldsymbol{x}$ is simply

$$p^* = \arg\min_p \mathrm{p}\left(\boldsymbol{x}_n|p\right)$$

References

[1] Michaël Betser. Sinusoidal polynomial parameter estimation using the distribution derivative. *Signal Processing, IEEE Transactions on*, 57(12):4633–4645, 2009.

[2] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[3] James Bronson. *An exploration of complex matrix factorization as a tool for single-channel musical source separation*. McGill University Libraries, 2014.

[4] Elliot Creager. *Musical source separation by coherent frequency modulation cues*. McGill University Libraries, 2016.

[5] RE Crochiere and Lawrence R Rabiner. *Multi-Rate Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1983.

[6] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[7] Ph Depalle, Guillermo Garcia, and Xavier Rodet. Tracking of partials for additive sound synthesis using hidden markov models. In *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, volume 1, pages 225–228. IEEE, 1993.

[8] Cédric Févotte, Nancy Bertin, and Jean-Louis Durrieu. Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis. *Neural computation*, 21(3):793–830, 2009.

[9] Neville H Fletcher and Thomas Rossing. *The physics of musical instruments*. Springer Science & Business Media, 2012.

[10] G David Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[11] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[12] Laurent Girin, Sylvain Marchand, Joseph Di Martino, Axel Robel, and Geoffroy Peeters. Comparing the order of a polynomial phase model for the synthesis of quasi-harmonic audio signals. In *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, pages 193–196. IEEE, 2003.

[13] Gene H Golub and Charles F Van Loan. *Matrix computations*. The John Hopkins University Press, 3rd edition, 1996.

[14] Daniel Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243, 1984.

[15] Brian Hamilton. *Non-stationary sinusoidal parameter estimation*. McGill University Libraries, 2011.

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

[17] David Havelock, Sonoko Kuwano, and Michael Vorländer. *Handbook of signal processing in acoustics*. Springer Science & Business Media, 2008.

[18] Monson H Hayes. *Statistical digital signal processing and modeling*. John Wiley & Sons, 2009.

[19] Nicholas J Higham. *Accuracy and stability of numerical algorithms*. Siam, 2002.

[20] Hao Jiang, Sidney Fels, and James J Little. A linear programming approach for multiple object tracking. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[21] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[22] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311. ACM, 1984.

[23] Steven M Kay. *Fundamentals of statistical signal processing, volume I: estimation theory*. Prentice Hall, 1993.

[24] Corey Kereliuk and Philippe Depalle. Sparse atomic modeling of audio: A review. In *Proc. of the 14th Int. Conference on Digital Audio Effects (DAFx-11), Paris, France*, 2011.

[25] Jonathan Le Roux and Emmanuel Vincent. Consistent wiener filtering for audio source separation. *IEEE signal processing letters*, 20(3):217–220, 2013.

[26] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[27] Robert Maher and James Beauchamp. An investigation of vocal vibrato for synthesis. *Applied Acoustics*, 30(2-3):219–245, 1990.

[28] John Makhoul. Linear prediction: A tutorial review. *Proceedings of the IEEE*, 63(4):561–580, 1975.

[29] Cécile MH Marin and Stephen McAdams. Segregation of concurrent sounds. ii: Effects of spectral envelope tracing, frequency modulation coherence, and frequency modulation width. *The Journal of the Acoustical Society of America*, 89(1):341–351, 1991.

[30] Stephen McAdams. Segregation of concurrent sounds. i: Effects of frequency modulation coherence. *The Journal of the Acoustical Society of America*, 86(6):2148–2159, 1989.

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

[32] Rodger J McNab, Lloyd A Smith, Ian H Witten, Clare L Henderson, and Sally Jo Cunningham. Towards the digital music library: Tune retrieval from acoustic input. In *Proceedings of the first ACM international conference on Digital libraries*, pages 11–18. ACM, 1996.

[33] Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.

[34] F Richard Moore. *Elements of computer music*. Prentice-Hall, Inc., 1990.

[35] James A Moorer. On the transcription of musical sound by computer. *Computer Music Journal*, pages 32–38, 1977.

[36] F Opolko and J Wapnick. Mcgill university master samples [cd-rom]. *Montreal, Quebec, Canada: jwapnick@ music. mcgill. ca*, 1987.

[37] R Gary Parker and Ronald L Rardin. *Discrete optimization*. Academic Press, 1988.

[38] H Vincent Poor. *An introduction to signal detection and estimation*. Springer Science & Business Media, 2013.

[39] Michael R Portnoff. Implementation of the digital phase vocoder using the fast fourier transform. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 24(3):243–248, 1976.

[40] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[41] Lawrence R Rabiner, Bernard Gold, and CA McGonegal. An approach to the approximation problem for nonrecursive digital filters. *Audio and Electroacoustics, IEEE Transactions on*, 18(2):83–106, 1970.

[42] Xavier Rodet, Yves Potard, and Jean-Baptiste Barriere. The chant project: From the synthesis of the singing voice to synthesis in general. *Computer Music Journal*, 8(3):15–31, 1984.

[43] Axel Roebel. Transient detection and preservation in the phase vocoder. In *International Computer Music Conference (ICMC)*, pages 247–250, 2003.

[44] Robert Roy, Arogyaswami Paulraj, and Thomas Kailath. Esprit–a subspace rotation approach to estimation of parameters of cisoids in noise. *IEEE transactions on acoustics, speech, and signal processing*, 34(5):1340–1342, 1986.

[45] Matti P Ryynänen and Anssi P Klapuri. Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, 32(3):72–86, 2008.

[46] Xavier Serra and Xavier Serra. A system for sound analysis/transformation/synthesis based on a deterministic plus stochastic decomposition. 1989.

[47] Madhusudana Shashanka, Bhiksha Raj, and Paris Smaragdis. Probabilistic latent variable models as nonnegative factorizations. *Computational intelligence and neuroscience*, 2008, 2008.

[48] Paris Smaragdis, Bhiksha Raj, and Madhusudana Shashanka. A probabilistic latent variable model for acoustic modeling. *Advances in models for acoustic processing, NIPS*, 148:8–1, 2006.

[49] Julius O Smith. *Physical Audio Signal Processing*. http://ccrma.stanford.edu/~jos/pasp/, accessed August 10, 2016. online book, 2010 edition.

[50] Julius Orion Smith and Xavier Serra. *PARSHL: an analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation*. CCRMA, Department of Music, Stanford University, 1987.

[51] Ljubisa Stankovic, Milos Dakovic, and Thayannathan Thayaparan. *Time-frequency signal analysis with applications*. Artech house, 2014.

[52] Charles Van Loan. *Computational frameworks for the fast Fourier transform*, volume 10. Siam, 1992.

[53] Jack K Wolf, Audrey M Viterbi, and Glenn S Dixon. Finding the best set of k paths through a trellis with application to multitarget tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 25(2):287–296, 1989.