Elements of Source Separation

Nicholas Esterer

July 2016

Contents

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 [16] [11] [26] 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.

Chapter 2

Methodology

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 [20] [19]. 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.

This thesis describes various approaches to the above tasks and utilises some of them in source separation experiments.

Chapter 3

Signal Modeling

3.1 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.

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 times $n \in \mathbb{Z}$ without any loss of information. $a_k, b_0 \in \mathbb{C}$ and are constants. Casually (no pun intended) 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 *delta function*

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

This delta function 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,$\omega \in \mathbb{R}$, $|r| < 1$ (recall that $j = \sqrt{-1}$ and is called the *imaginary number*). Then the difference equation is

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

Exciting this with the delta function we get

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

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 exponential. 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 delta function). This technique is presented as a motivation for the following techniques and is not pursued here. The interested reader is referred to [18] 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 in to one of time τ and frequency ω is known as the *Discrete-time Short-time Fourier Transform* (DTSTFT).

3.2 Polynomial Phase Models

The DTFT and DTSTFT are very useful because they are invertible [27] and fast algorithms exist for their computation by digital computer [34]. 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 [17] for a review). In the case of musical signals, however, we have some prior information and can make certain assumptions about the underlying functions.

3.2.1 Sinusoidal Representations

Many musical acoustic signals are quasi-harmonic, 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 $\varepsilon \sim \mathcal{N}(0, \sigma)$ and $A_p \in \mathbb{R}$ and $\phi_p \in \mathbb{R}$ 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 $c_q \in \mathbb{C}$.

The sum-of-sinusoids model of McAulay and Quatieri estimates these coefficients in an indirect way [21]. 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

$$\tilde{\phi}(n) = c_3(n - \tau_0)^3 + c_2(n - \tau_0)^2 + c_1(n - \tau_0) + c_0 \tag{3.10}$$

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

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ H^3 & H^2 & H & 1 \\ 0 & 0 & 1 & 0 \\ 3H^2 & 2H & 1 & 0 \end{pmatrix} \begin{pmatrix} \Im\{c_3\} \\ \Im\{c_2\} \\ \Im\{c_1\} \\ \Im\{c_0\} \end{pmatrix} = \begin{pmatrix} \angle\{X(\tau_0, \omega_0)\} \\ \angle\{X(\tau_1, \omega_1)\} + 2\pi M \\ \omega_0 \\ \omega_1 \end{pmatrix} \tag{3.11}$$

and choosing M so that

$$\int_0^H (\tilde{\phi}''(t))^2 dt \tag{3.12}$$

is minimized.

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

$$\begin{pmatrix} 0 & 1 \\ H & 1 \end{pmatrix} \begin{pmatrix} \Re\{c_1\} \\ \Re\{C_0\} \end{pmatrix} = \begin{pmatrix} \log(|X(\tau_0, \omega_0)|) \\ \log(|X(\tau_1, \omega_1)|) \end{pmatrix} \tag{3.13}$$

3.2.2 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 [9]. 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* $\psi(n)$

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

Differentiating with respect to n we obtain

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

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

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

which after rearranging is

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

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

$$\sum_{q=1}^{Q} q c_q \left\langle n^{q-1} x(n), \overline{\psi_r}(n) dn \right\rangle = - \left\langle x, \frac{d\overline{\psi_r}}{dn} \right\rangle \tag{3.14}$$

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 the 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

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

The inner-product between η and γ is 0, by the orthogonality principle [16]. 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

$$c_0 = \log\left(\langle s, \gamma \rangle \right) - \log\left(\langle \gamma, \gamma \rangle \right) \tag{3.15}$$

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

3.2.3 The choice of 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 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 [10]. We require that the window be at least once-differentiable and zero outside of a certain interval, therefore, somewhat informally, we require

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

The *Hann* window possesses this property

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

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 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

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

With $M = 2$ and $a_0 = a_1 = 0.5$, we have the Hann window. The *Blackman-Harris* family of windows are also sum-of-cosine windows. For these windows, optimization techniques were used to search for coefficients giving optimum properties, such as minimum height of the highest side-lobe (maximum out-of-band rejection) [29]. 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 essentially nulls any influence of signals outside of a bandwidth of interest.

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 **??**. This window will be referred to as the $\mathcal{C}^1$ 4-Term Blackman-Harris window.

Table 3.1

Window	a_0	a_1	a_2	a_3
Minimum 4-term Blackman-Harris	0.35857	0.48829	0.14128	0.01168
$\mathcal{C}^1$ 4-Term Blackman-Harris	0.35874	0.48831	0.14127	0.01170

3.3 Partial Tracking

In Section 3.2.1 we discussed how to determine reasonable values for the coefficients of a cubic phase polynomial by using the frequency, phase and time difference of two local maxima in the DTSTFT. In this section we discuss possible ways of determining which local maxima are connected. This is referred to as *peak matching* [21] or *partial tracking* [33] [4].

3.3.1 A Greedy Method

The original method of connecting peaks in the spectrogram is from [21]. This method is simple and fast but, as we will see, can be sensitive to spurious peaks.

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*. Consider the parameters of local maxima in adjacent frames k and $k+1$ with M maxima in frame k and N maxima in frame $k+1$. In [21] the parameters are the instantaneous amplitude, phase and frequency and are indexed by frequency as $\omega_0^k, \ldots, \omega_{M-1}^k$ and $\omega_0^{k+1}, \ldots, \omega_{N-1}^{k+1}$, but here we allow for an arbitrary set of parameters $\theta_0^k, \ldots, \theta_{M-1}^k$ and $\theta_0^{k+1}, \ldots, \theta_{N-1}^{k+1}$, such as the coefficients of a phase polynomial. Define a distance function $\mathcal{D}(\theta_i, \theta_j)$ that computes the similarity between two sets of parameters. We will now consider a method that finds L pairs of parameters that are closest.

We compute the cost matrix C

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

so that the ith row and jth column contain $C_{i,j} = \mathcal{D}\left(\theta_i^k, \theta_j^{k+1}\right)$. 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

Fig. 3.1

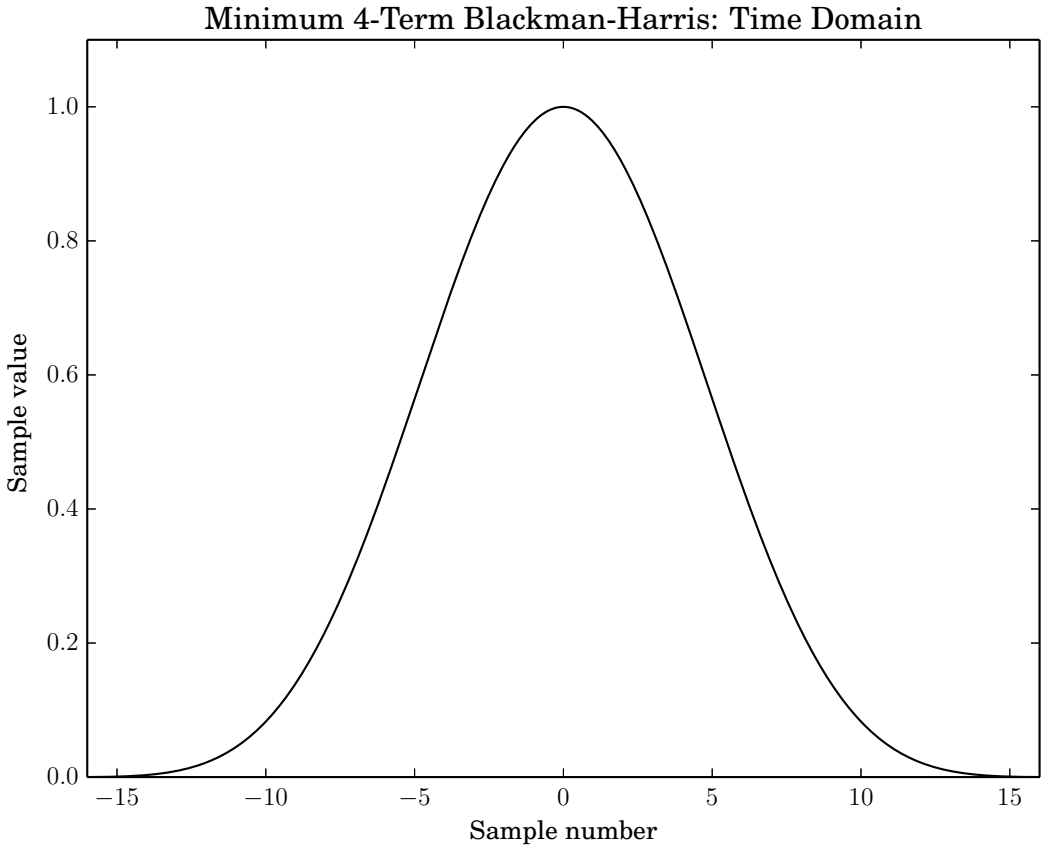

Minimum 4-Term Blackman-Harris: Time Domain
Sample value
1.0
0.8
0.6
0.4
0.2
0.0
−15
−10
−5
0
5
10
15
Sample number

Fig. 3.2

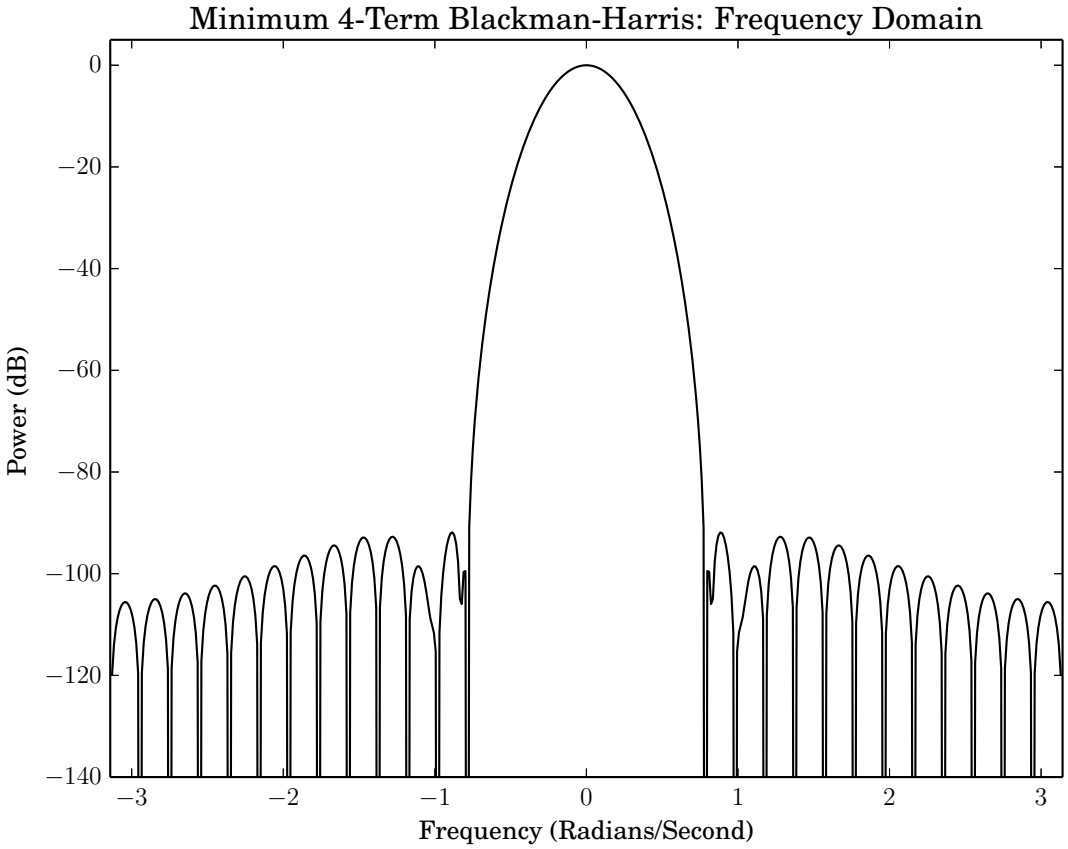

Minimum 4-Term Blackman-Harris: Frequency Domain
Power (dB)
0
−20
−40
−60
−80
−100
−120
−140
−3
−2
−1
0
1
2
3
Frequency (Radians/Second)

Fig. 3.3

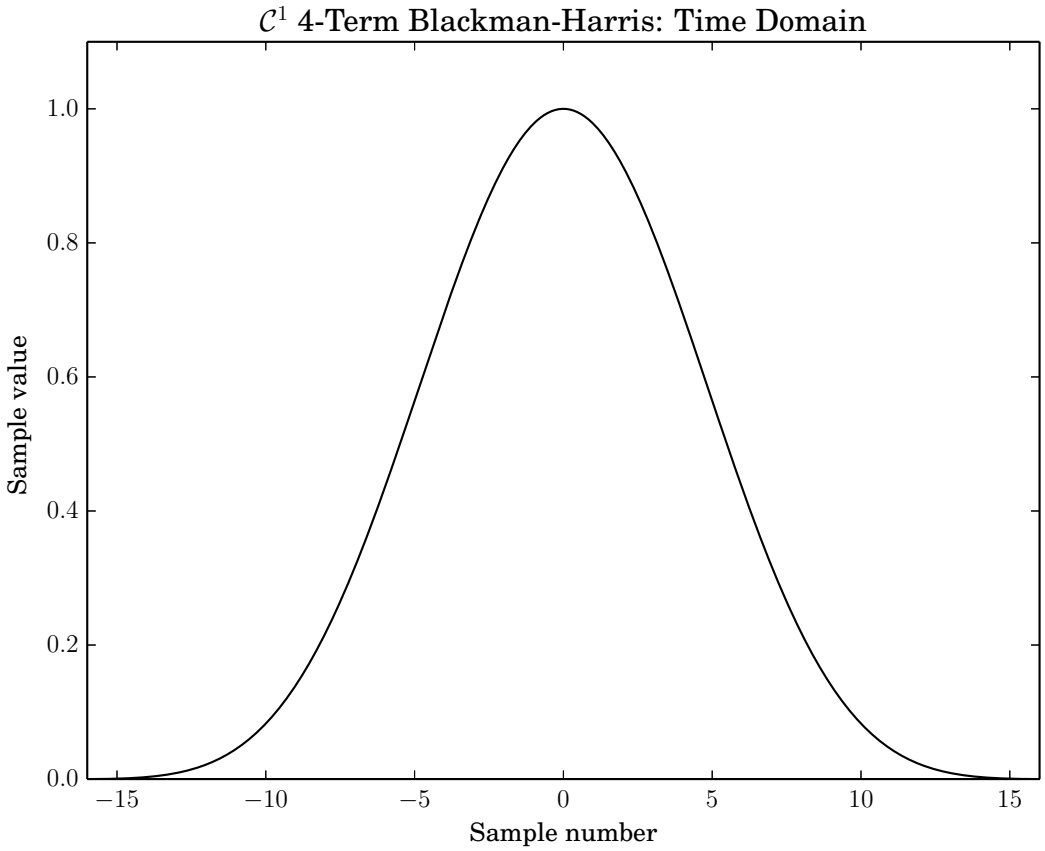
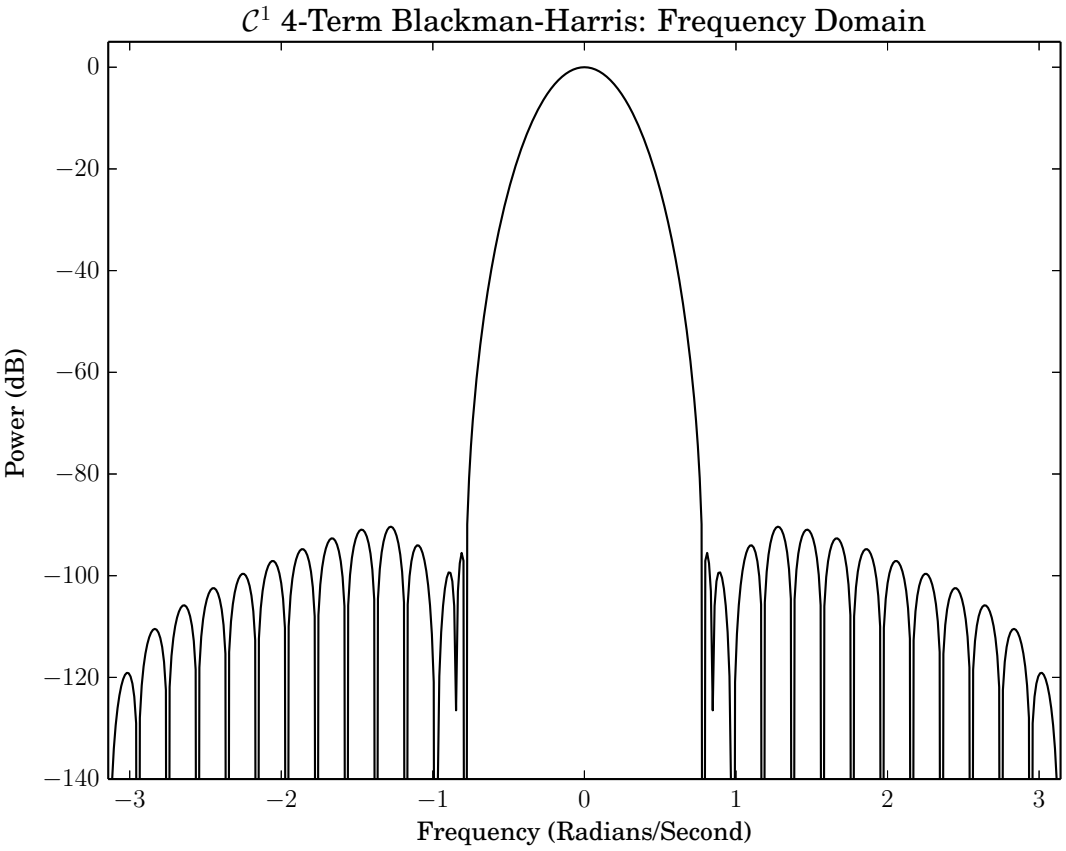

$\mathcal{C}^1$ 4-Term Blackman-Harris: Time Domain
Sample value
1.0
0.8
0.6
0.4
0.2
0.0
−15
−10
−5
0
5
10
15
Sample number

Fig. 3.4


$\mathcal{C}^1$ 4-Term Blackman-Harris: Frequency Domain
Power (dB)
0
−20
−40
−60
−80
−100
−120
−140
−3
−2
−1
0
1
2
3
Frequency (Radians/Second)

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 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**
  $i_l, j_l = \text{argmin}_{i \in [0,\ldots,M-1] \setminus \Gamma_i, j \in [0,\ldots,M-1] \setminus \Gamma_j} C_{i,j}$;
  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 sets of parameters, constructing an 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, and works well with a variety of signals encountered in audio [21] [33].

3.3.2 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 k, we make k 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. 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

$$c_{\text{total}} = \boldsymbol{c}^T \boldsymbol{x}$$

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 3.5 we have an example of a simple graph or lattice. The numbers are indices of nodes in the graph and the possible connections between them are indicated by lines, or *edges*. We would like to find the two shortest paths based on different criteria. In Figure 3.6 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 $C_{i,j,k}$ represents the cost of travelling on the path connecting the ith node in layer 0, the jth node in layer 1 and the kth node in layer 2. 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 3.6 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.

Fig. 3.5

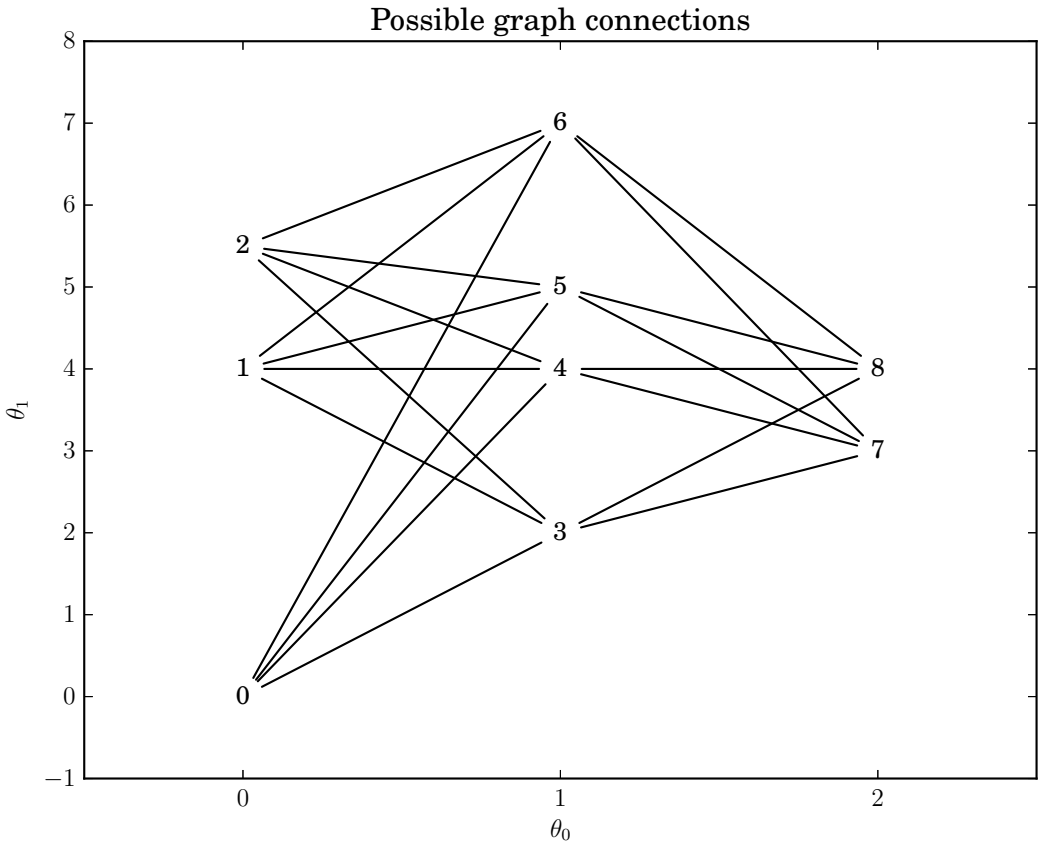
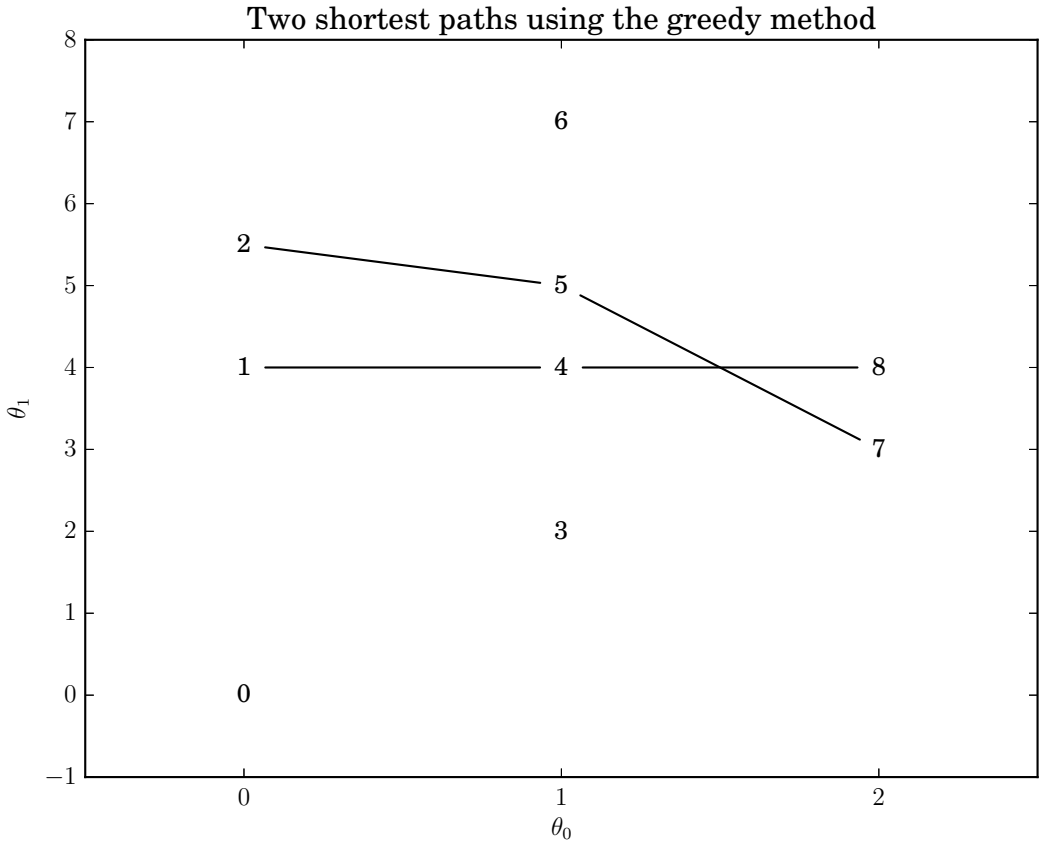


Fig. 3.6


Two shortest paths using the greedy method
8
7
6
5
4
3
2
1
0
−1
0
1
2
θ_0
θ_1
6
2
5
1
4
8
7
3
0

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 successive connecting nodes $j_0 \dots j_{R-1}$. Define the vector

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

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

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

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 previous connecting nodes $i_0 \dots i_{R-1}$, the vector

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

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

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

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. If r is the index of the node considered, the vector is simply

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

and the constraint

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

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 k and $k+1$. We constrain the number of paths going from edges Γ_k in frame k to Γ_{k+1} by forming the vector

$$\boldsymbol{a}^{c,k} = \sum_{j \in \Gamma_k} \boldsymbol{a}^{s,j}$$

and asserting the constraint

$$\left\langle \boldsymbol{a}^{c,k}, \boldsymbol{x} \right\rangle = 1$$

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

$$\min \boldsymbol{c}^T \boldsymbol{x}$$

subject to

$$\mathbf{0} \leq \begin{bmatrix} \boldsymbol{A}_s \\ \boldsymbol{A}_p \end{bmatrix} \boldsymbol{x} \leq \mathbf{1}$$

$$\begin{bmatrix} \boldsymbol{A}_b \\ \boldsymbol{A}_c \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \end{bmatrix}$$

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

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

The solution of the two best paths using the LP formulation is shown in Figure 3.7 and a comparison of the total costs is shown in Table 3.2

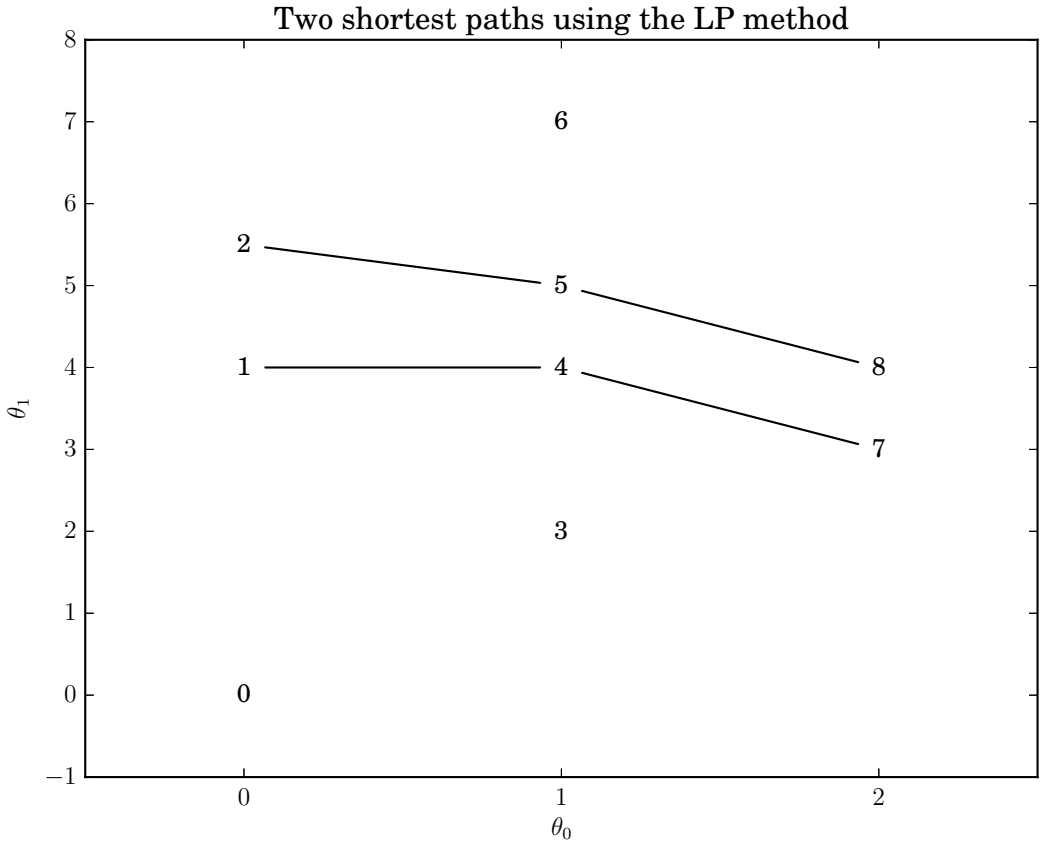
Table 3.2 Comparison of total costs

Greedy	LP
5.354102	4.946461

The LP formulation is based on a multiple object tracking algorithm for video [13]. A proof that the solution $\boldsymbol{x}^*$ will have entries equal to either 0 or 1 can be found in [25, p. 167]. The theoretical computational complexity of the linear program is polynomial in the number of variables, see [15] 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. 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].

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 [28] 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 [5].

Fig. 3.7


Two shortest paths using the LP method
8
7
6
5
4
3
2
1
0
−1
θ_1
0
1
2
θ_0
0
1
2
3
4
5
6
7
8

3.3.3 Partial paths on example signal

3.4 Classification

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.

3.4.1 Principal components analysis (PCA)

The following development is based on [14]. 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 $\boldsymbol{x}$. There are many ways to determine $\hat{h}$ based on its assumed form. For example, if it is assumed that $h(\boldsymbol{x}) = \hat{\boldsymbol{\beta}}^T\boldsymbol{x}$ then a common estimator of $\hat{\boldsymbol{\beta}}$, given $\boldsymbol{X}$, a matrix of observations of $\boldsymbol{x}$ with the observations as its rows, and the classifications $\boldsymbol{y}$, is [6, p. 12]

$$\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{3.16}$$

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 **??**). To automate classification we treat the dataset beforehand to increase the robustness of the classification algorithm, and allow the algorithm to estimate some class memberships. The classification algorithm will give some estimated $\hat{h}$ as above, but it is doubtful that we would find a situation to use it to classify new data-points. In the case that the classification gives a continuous variable as in Equation 3.16, thresholding values are required to convert $\hat{h}(\boldsymbol{x})$ to discrete values. Because we would have to choose these values beforehand, this sort of classification algorithm is probably not very useful in practice. For this reason we opt for classification algorithms that give discrete indicators as output, purely as a function of the input dataset. The algorithm we consider is the *Gaussian mixture model (GMM).*

3.4.2 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 [22], [3], which is an algorithm suitable for estimating missing data from known ones. First, define

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

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:

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

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 $\boldsymbol{\mu}_p^{k+1}$:

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

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 $\mathbf{\Sigma}_p^{k+1}$:

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

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

$$p^* = \mathrm{argmax}_p \, \mathrm{p}\left(\boldsymbol{x}_n|p\right)$$

3.5 Partial synthesis

Once the parameters have been estimated, the partials have been determined, and the partials have been grouped into sources, we can now synthesize a single source by choosing only those partials belonging to a single source and resynthesizing from the parameters in some fashion. A popular technique that is straightforward to implement is the *overlap-and-add* procedure [27], [23]. We assume that in the neighbourhood of τ_r the 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 samples. 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

$$L_- = \left\lfloor \frac{n-M}{H} + 1 \right\rfloor$$

and

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

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}. This is what is carried out in Section 3.2.1 and results in a cubic interpolating polynomial for phase. From Equation 3.14 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.

Chapter 4

Extended phase model

In Section 3.2.1 it was shown how a cubic phase function could be postulated from two connected measurements (see Section 3.3) giving a cubic phase polynomial. Here we will demonstrate how to include the additional phase polynomial coefficients estimated using DDM in calculating a phase function. In a later section we will compare the synthesis quality of the classical McAulay –Quatieri approach with our new method.

4.1 Cubic phase polynomial

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

A complex sinusoid with cubic phase has the following form:

$$\phi(n) = \exp\left(c_3 n^3 + c_2 n^2 + c_1 n + c_0\right) \tag{4.1}$$

One with quadratic phase has this form:

$$x(n) = \exp\left(a_2 n^2 + a_1 n + a_0\right) \tag{4.2}$$

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

We can estimate the coefficients of Equation 4.2 using the DDM. From Equation 3.14, we define the three functions

$$X_{p_1}(\tau, k) = \sum_{m=0}^{M-1} w(m)x(m+\tau)\exp(-j2\pi\frac{km}{M}) \tag{4.3}$$

$$X_{p_2}(\tau, k) = \sum_{m=0}^{M-1} 2mw(m)x(m+\tau)\exp(-j2\pi\frac{km}{M}) \tag{4.4}$$

$$X_{w'}(\tau, k) = -j2\pi\frac{k}{M}X_{p_1}(\tau, k) + \sum_{m=0}^{M-1}\frac{dw}{dn}(m)x(m+\tau)\exp(-j2\pi\frac{km}{M}) \tag{4.5}$$

Where M is the length of the window and k is the frequency "bin". 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_{p_1} 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 [31, p. 42]. To have a system of equations with a unique solution, take the two adjacent bins $k-1$ and $k+1$ to have enough unique atoms for Equation 3.14. 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 to have improved estimation accuracy in situations where components are adaquately separated in frequency. Then a_2 and a_1 can be determined by solving the linear system

$$\begin{pmatrix} X_{p_1}(\tau, k^*-1) & X_{p_2}(\tau, k^*-1) \\ X_{p_1}(\tau, k^*) & X_{p_2}(\tau, k^*) \\ X_{p_1}(\tau, k^*+1) & X_{p_2}(\tau, k^*+1) \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} X_{w'}(\tau, k^*-1) \\ X_{w'}(\tau, k^*) \\ X_{w'}(\tau, k^*+1) \end{pmatrix}$$

With a_1 and a_2 determined, we can use Equation 3.15 to estimate a_0. We will write a_i^τ to refer to coefficient i determined at time τ. 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 Section 3.3), 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 $c_0 = \Im\{a_0^{\tau_0}\}$. 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

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

where $U^* \in \mathbb{Z}$ is determined to minimize Equation 3.12, in this case:

$$\tilde{U} = \text{argmin}_U \int_0^H (6c_3t + 2c_2)^2\, dt \tag{4.6}$$

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

$$\begin{pmatrix} H^3 & H^2 & H \\ \frac{3}{4}H^2 & H & 1 \\ 3H & 2 & 0 \end{pmatrix} \begin{pmatrix} c_3 \\ c_2 \\ c_1 \end{pmatrix} = \begin{pmatrix} \Im\left\{a_0^{\tau_1}\right\} - \Im\left\{a_0^{\tau_0}\right\} + 2\pi U^* \\ \frac{1}{2}\left(\Im\left\{a_1^{\tau_0}\right\} + \Im\left\{a_1^{\tau_1}\right\}\right) \\ \frac{1}{2}\left(\Im\left\{a_2^{\tau_0}\right\} + \Im\left\{a_2^{\tau_1}\right\}\right) \end{pmatrix} \tag{4.7}$$

Solving for $c_1, \ldots, c_3$, we have

$$\begin{pmatrix} c_3 \\ c_2 \\ c_1 \end{pmatrix} = \begin{pmatrix} \frac{4}{H^3}\left(\Im\left\{a_0^{\tau_1}\right\} - \Im\left\{a_0^{\tau_0}\right\} + 2\pi U^*\right) - \frac{2}{H^2}\left(\Im\left\{a_1^{\tau_1}\right\} + \Im\left\{a_1^{\tau_0}\right\}\right) \\ \frac{-6}{H^2}\left(\Im\left\{a_0^{\tau_1}\right\} - \Im\left\{a_0^{\tau_0}\right\} + 2\pi U^*\right) - \frac{3}{H}\left(\Im\left\{a_1^{\tau_1}\right\} + \Im\left\{a_1^{\tau_0}\right\}\right) + \frac{1}{4}\left(\Im\left\{a_2^{\tau_0}\right\} + \Im\left\{a_2^{\tau_1}\right\}\right) \\ \frac{-H}{4}\left(\Im\left\{a_2^{\tau_0}\right\} + \Im\left\{a_2^{\tau_1}\right\}\right) + \frac{3}{H}\left(\Im\left\{a_0^{\tau_1}\right\} - \Im\left\{a_0^{\tau_0}\right\} + 2\pi U^*\right) - \Im\left\{a_1^{\tau_1}\right\} - \Im\left\{a_1^{\tau_0}\right\} \end{pmatrix}$$

and then $\tilde{U}$ is determined using Equation 4.6 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^*.

4.2 Cubic amplitude polynomial

Solving for the cubic polynomial describing the local amplitude function

$$\mu(n) = \exp\left(d_3 n^3 + d_2 n^2 + d_1 n + d_0\right) \tag{4.8}$$

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 $d_0 = \Re\left\{a_0^{\tau_0}\right\}$ as in Section 6. 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

$$\begin{pmatrix} 0 & 0 & 1 \\ 3H^2 & 2H & 1 \\ 0 & 2 & 0 \\ 6H & 2 & 0 \end{pmatrix} \begin{pmatrix} d_3 \\ d_2 \\ d_1 \end{pmatrix} = \begin{pmatrix} \Re\{a_1^{\tau_0}\} \\ \Re\{a_1^{\tau_1}\} \\ \Re\{a_2^{\tau_0}\} \\ \Re\{a_2^{\tau_1}\} \end{pmatrix} \tag{4.9}$$

subject to

$$\begin{pmatrix} H^3 & H^2 & H \end{pmatrix} \begin{pmatrix} d_3 \\ d_2 \\ d_1 \end{pmatrix} = \begin{pmatrix} \Re\{a_0^{\tau_1}\} \end{pmatrix}$$

This can be solved using numerical methods, in particular, using a specific interpretation of weighted least-squares [8, p. 266].

4.3 Quintic phase polynomial

Solving for the coefficients of a quintic phase polynomial is done very similarly to Section 6. The quintic phase polynomial is

$$\phi(n) = \exp\left(c_5 n^5 + c_4 n^4 + c_3 n^3 + c_2 n^2 + c_1 n + c_0\right) \tag{4.10}$$

We have

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

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

$$\begin{pmatrix} H^5 & H^4 & H^3 \\ 5H^4 & 4H^3 & 3H^2 \\ 20H^3 & 12H^2 & 6H \end{pmatrix} \begin{pmatrix} c_5 \\ c_4 \\ c_3 \end{pmatrix} = \begin{pmatrix} -\Im\{a_2^{\tau_0}\}\frac{H^2}{2} - \Im\{a_1^{\tau_0}\}H + \Im\{a_0^{\tau_1}\} - \Im\{a_1^{\tau_1}\} + 2\pi U^* \\ -\Im\{a_2^{\tau_0}\}H + \Im\{a_1^{\tau_1}\} - \Im\{a_1^{\tau_0}\} \\ \Im\{a_2^{\tau_1}\} - \Im\{a_2^{\tau_0}\} \end{pmatrix} \tag{4.11}$$

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

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

and then rounded to produce U^* as above.

4.4 Quintic amplitude polynomial

Solving for the quintic amplitude polynomial

$$\mu(n) = \exp\left(d_5 n^5 + d_4 n^4 + d_3 n^3 + d_2 n^2 + d_1 n + d_0\right) \tag{4.12}$$

is as follows:

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

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

$$\begin{pmatrix} H^5 & H^4 & H^3 \\ 5H^4 & 4H^3 & 3H^2 \\ 20H^3 & 12H^2 & 6H \end{pmatrix} \begin{pmatrix} d_5 \\ d_4 \\ d_3 \end{pmatrix} = \begin{pmatrix} -\Re\left\{a_2^{\tau_0}\right\} \frac{H^2}{2} - \Re\left\{a_1^{\tau_0}\right\} H + \Re\left\{a_0^{\tau_1}\right\} - \Re\left\{a_1^{\tau_1}\right\} \\ -\Re\left\{a_2^{\tau_0}\right\} H + \Re\left\{a_1^{\tau_1}\right\} - \Re\left\{a_1^{\tau_0}\right\} \\ \Re\left\{a_2^{\tau_1}\right\} - \Re\left\{a_2^{\tau_0}\right\} \end{pmatrix} \tag{4.13}$$

The quintic interpolating phase polynomial has been proposed in a previous paper [7] although they do not directly estimate the frequency slope, choosing instead to derive it using the difference in frequency between two analysis frames.

4.5 Evaluation

We compared the quality of an analysis-synthesis system using the original method proposed by McAulay and Quatieri, the cubic interpolation method, and the quartic interpolation method. A frequency- and amplitude-modulated sinusoid was synthesized and then analysed frame-by-frame using the DDM to estimate its initial phase (amplitude), frequency (amplitude slope), and frequency-modulation (amplitude-modulation). Afterwards, the signal was resynthesized using the estimated parameters and compared with the original.

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

$$x_\phi(n) = \exp\left(b_3 n^3 + b_2 n^2 + b_1 n + b_0\right)$$

the frequency break-points are summarized in Table 4.1. The initial phase is 0 radians.

A quartic polynomial is used for the amplitude function

$$x_\mu(n) = \exp\left(u_4 n^4 + u_3 n^3 + u_2 n^2 + u_1 n + u_0\right)$$

and its amplitude break-points are summarized in Table 4.2.

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 3.2.3).

Table 4.1

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

Table 4.2

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

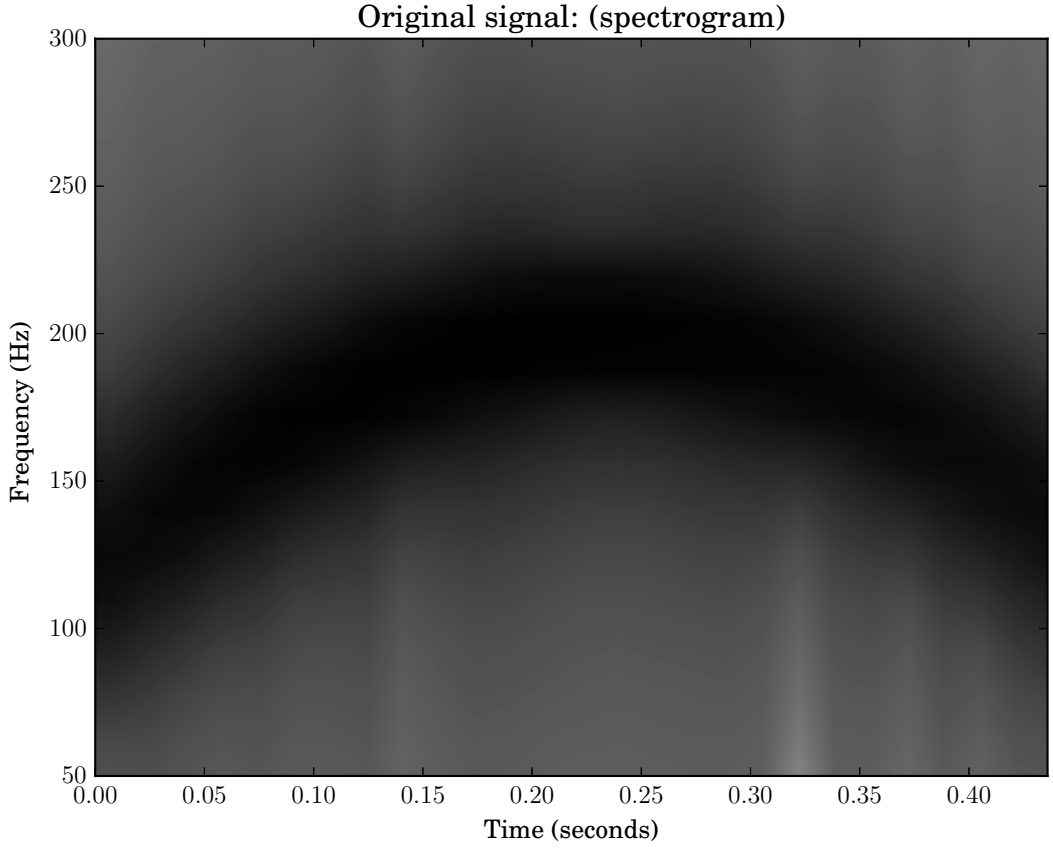


Fig. 4.1 This depicts the log spectrum. Darker regions represent greater values whereas lighter are smaller values.

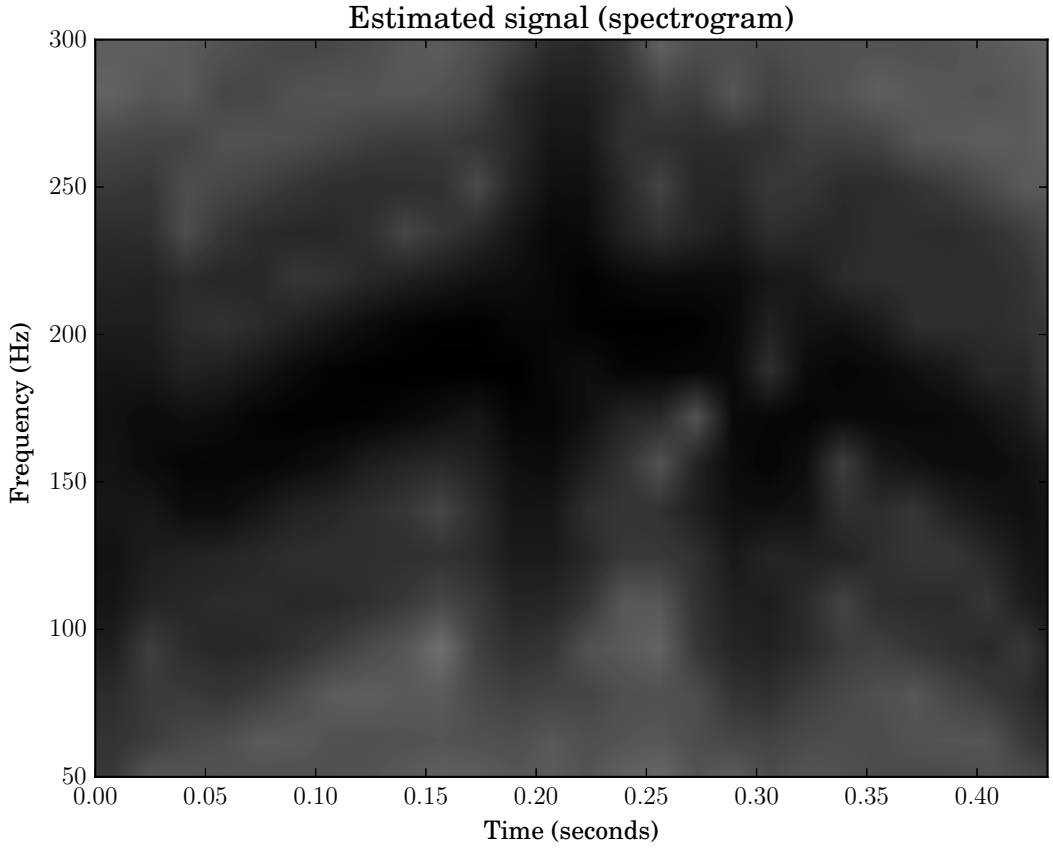


Fig. 4.2 This depicts the log spectrum. Darker regions represent greater values whereas lighter are smaller values. The estimated signal is computed using the original method proposed by McAulay and Quatieri.

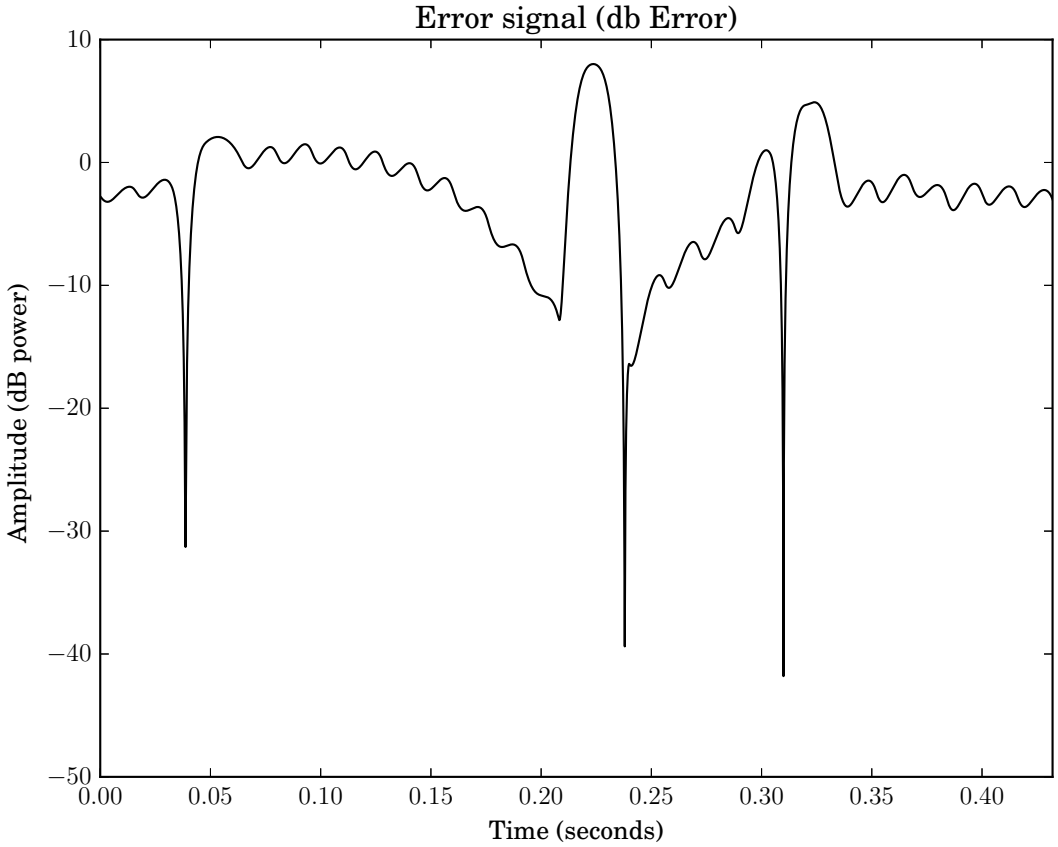


Fig. 4.3 The error when subtracting the original signal from the estimated signal. The estimated signal is computed using the original method proposed by McAulay and Quatieri.

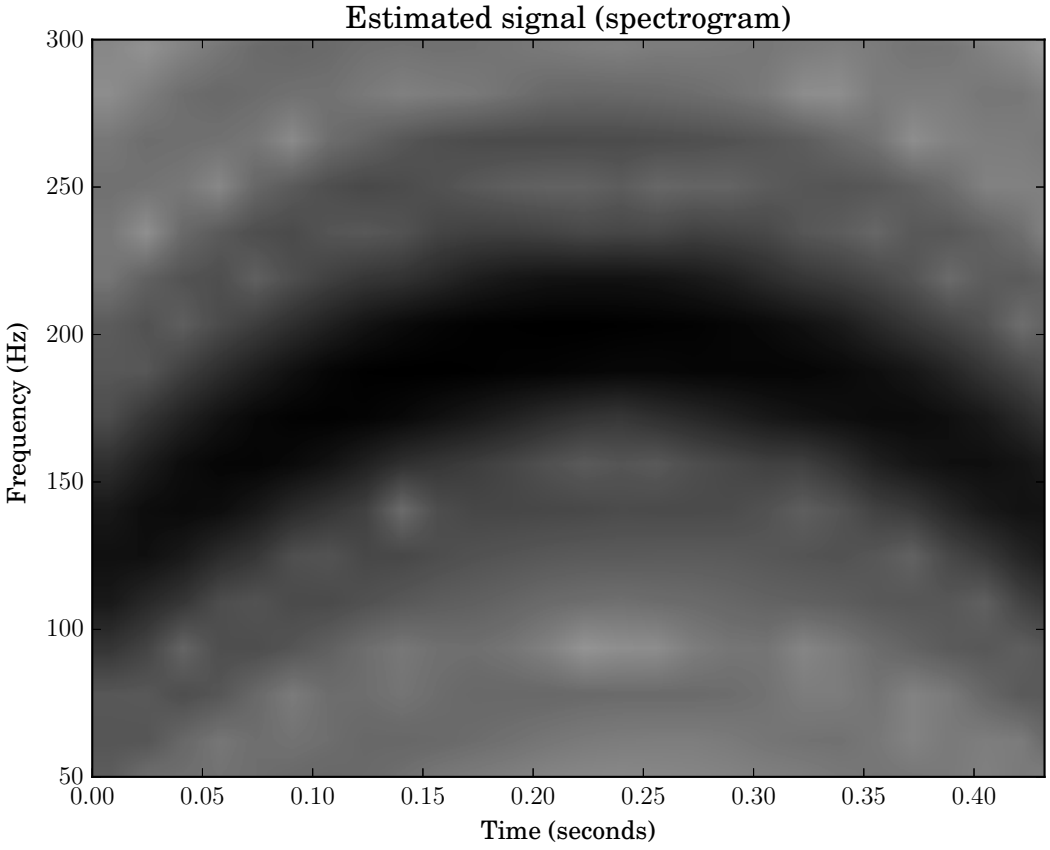


Fig. 4.4 This depicts the dB spectrum. Darker regions represent greater values whereas lighter are smaller values. The estimated signal is computed using the cubic interpolation method proposed here.

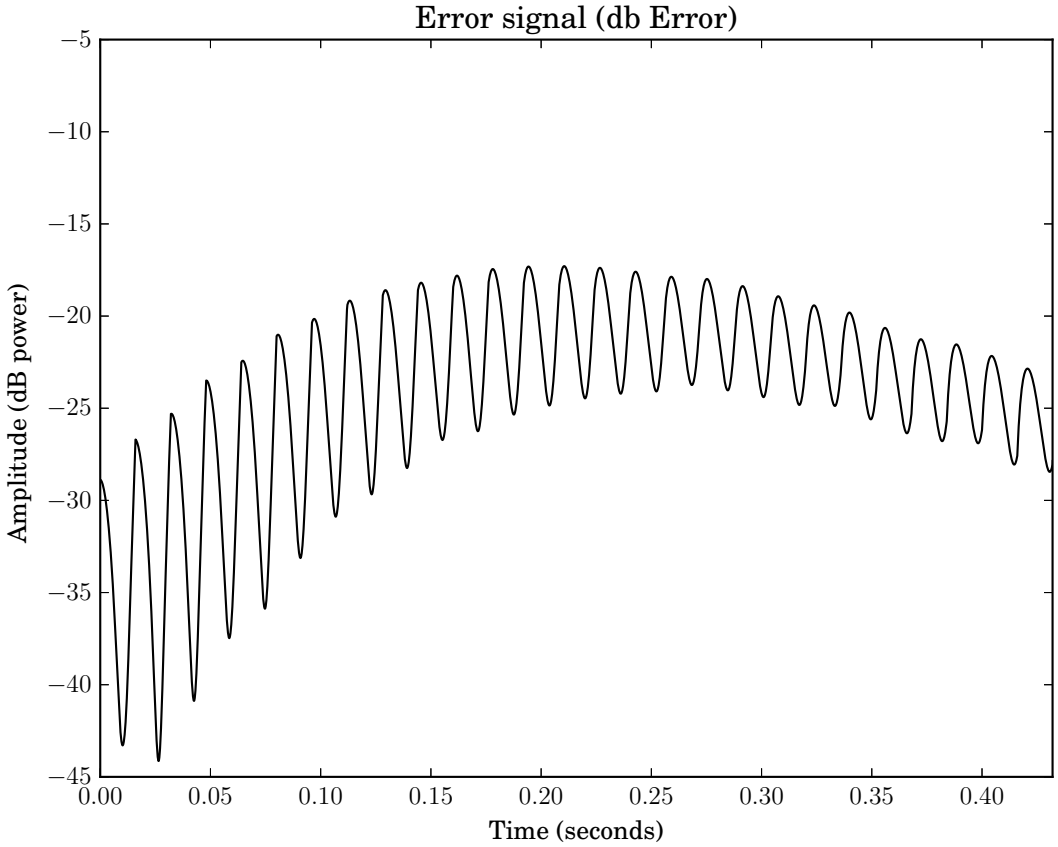


Fig. 4.5 The error when subtracting the original signal from the estimated signal. The estimated signal is computed using the cubic interpolation method proposed here.

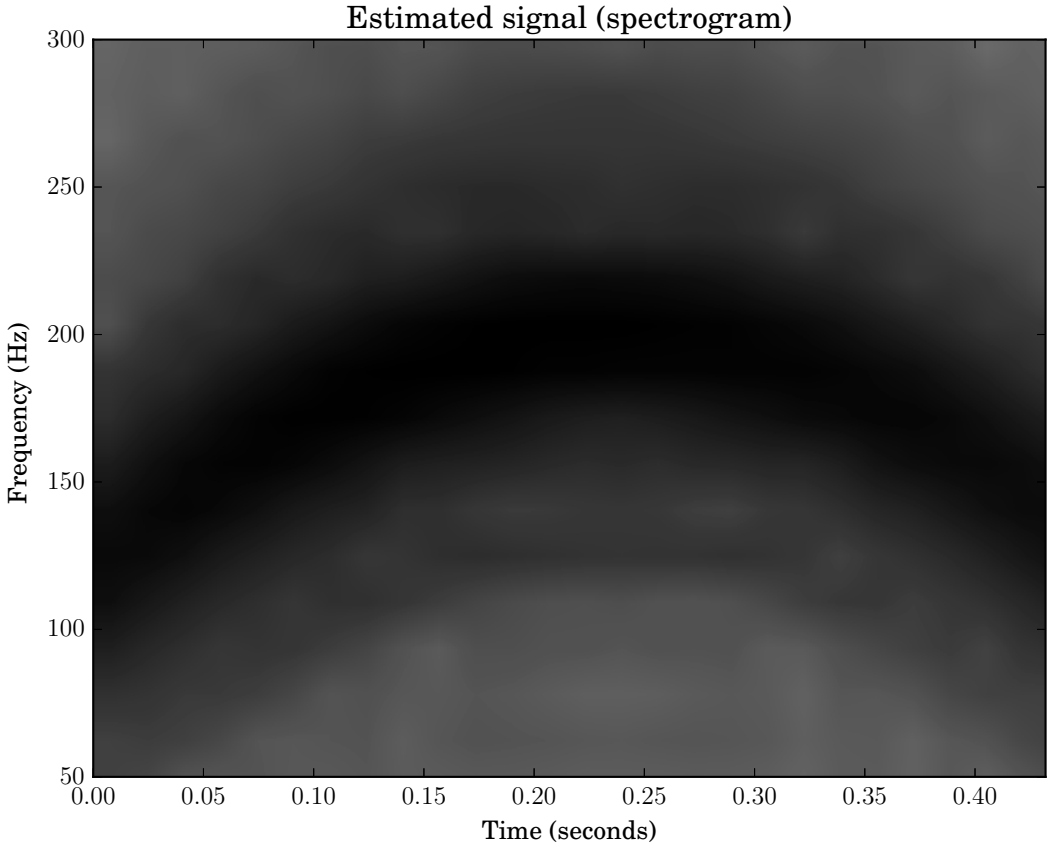


Fig. 4.6 This depicts the dB spectrum. Darker regions represent greater values whereas lighter are smaller values. The estimated signal is computed using the quintic interpolation method proposed here.

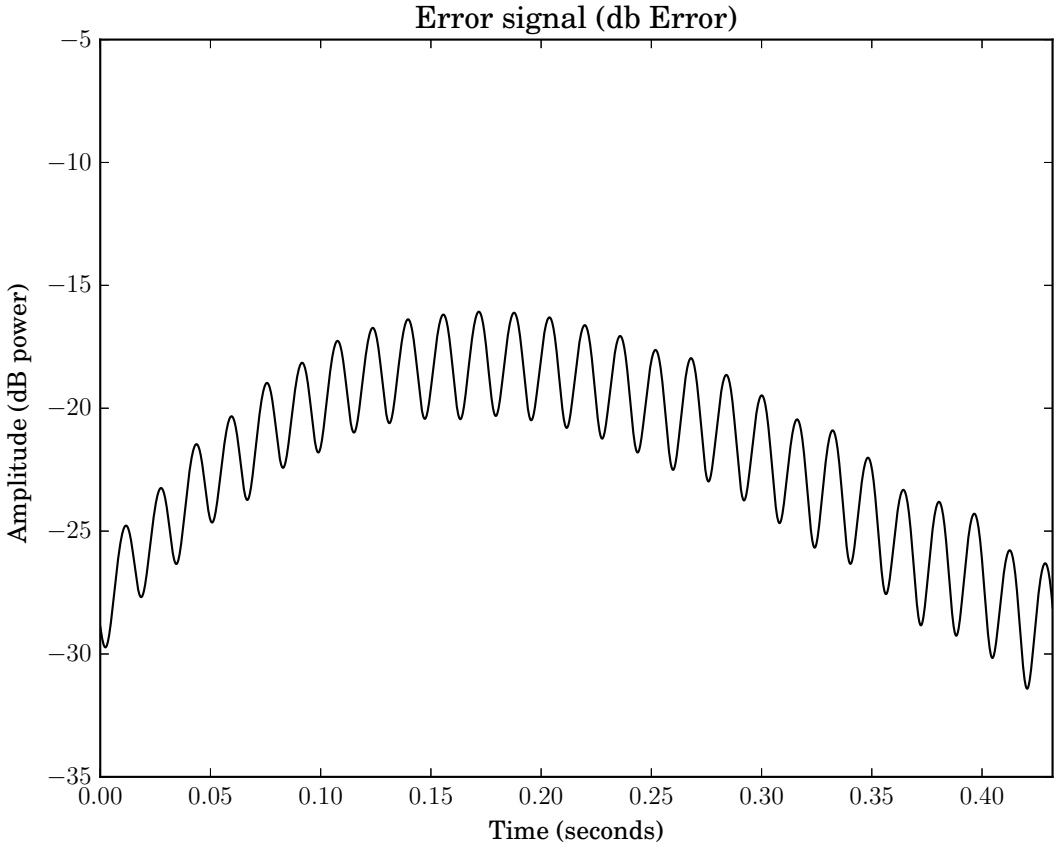


Fig. 4.7 The error when subtracting the original signal from the estimated signal. The estimated signal is computed using the quintic interpolation method proposed here.

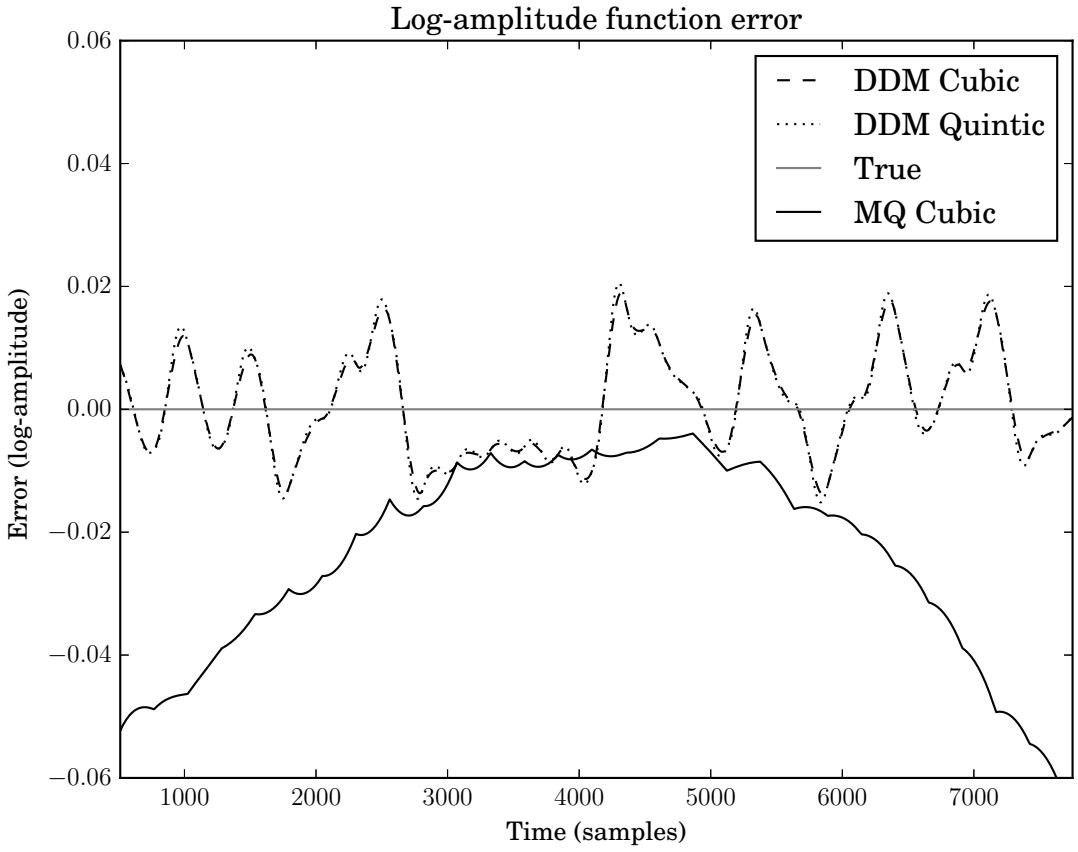


Fig. 4.8 This shows the error of the interpolated signals when compared with the original signal for the three proposed methods.

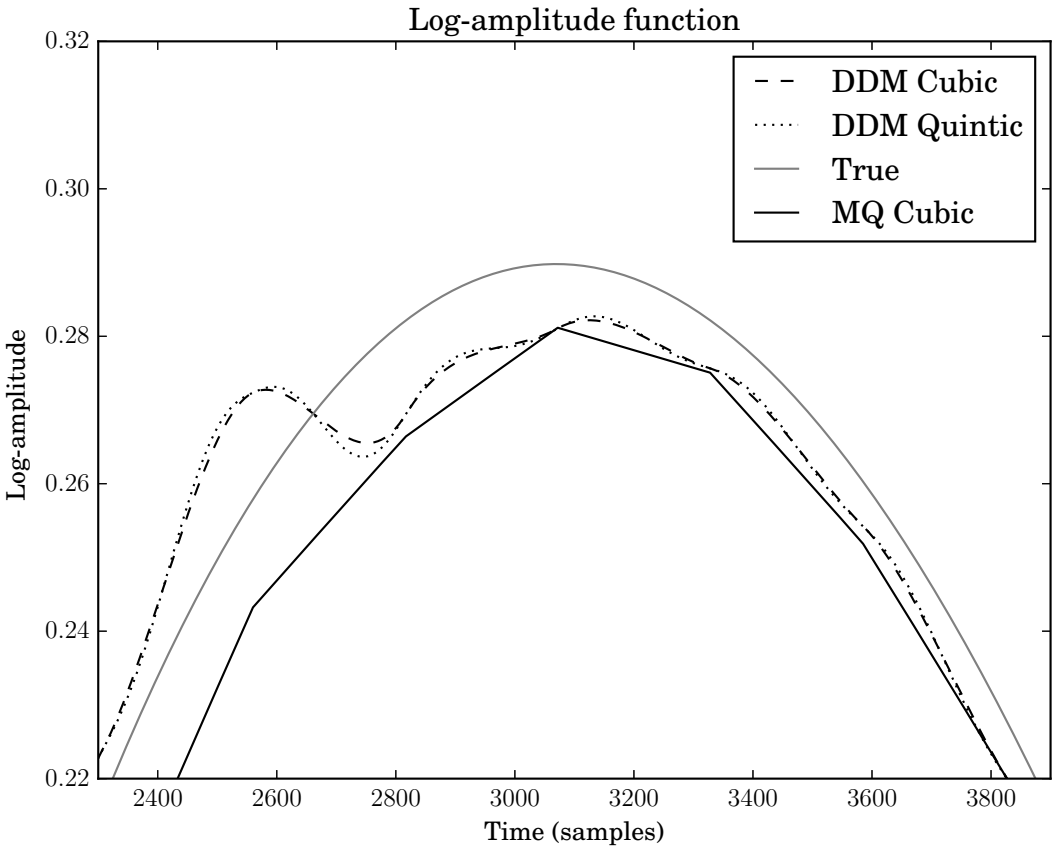


Fig. 4.9 This compares the original log-amplitude function with the interpolated log-amplitude functions. The log-amplitude functions are considered because these are the real part of the polynomial exponents in the complex sinusoid model.

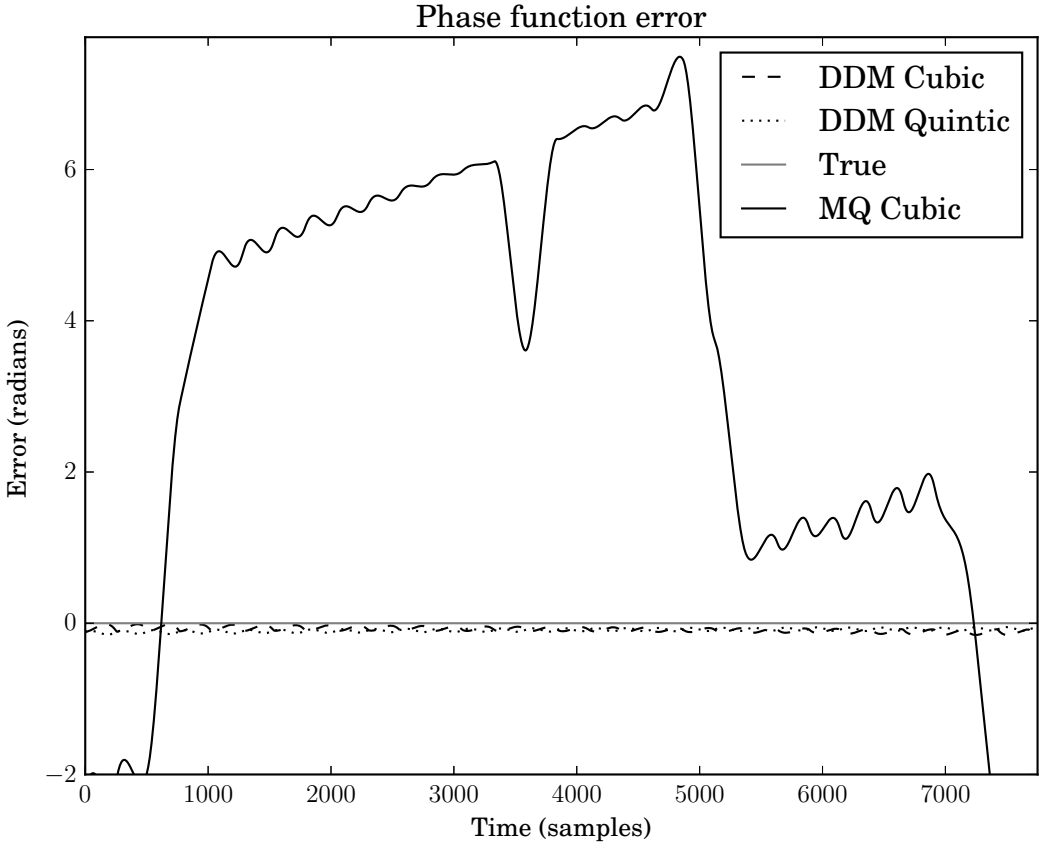


Fig. 4.10 This compares the phase functions with the interpolated phase functions. The phase functions are considered because these are the imaginary part of the polynomial exponents in the complex sinusoidal model. The error exhibited by the McAulay –Quatieri interpolation is misleading because phase values should be compared after computing the remaineder when dividing by 2π. This means there are only brief segments where the phase error is significant (compare with Figure 4.3). The plot is shown without adjusting the values to show the accuracy of the proposed methods in the interpolated regions.

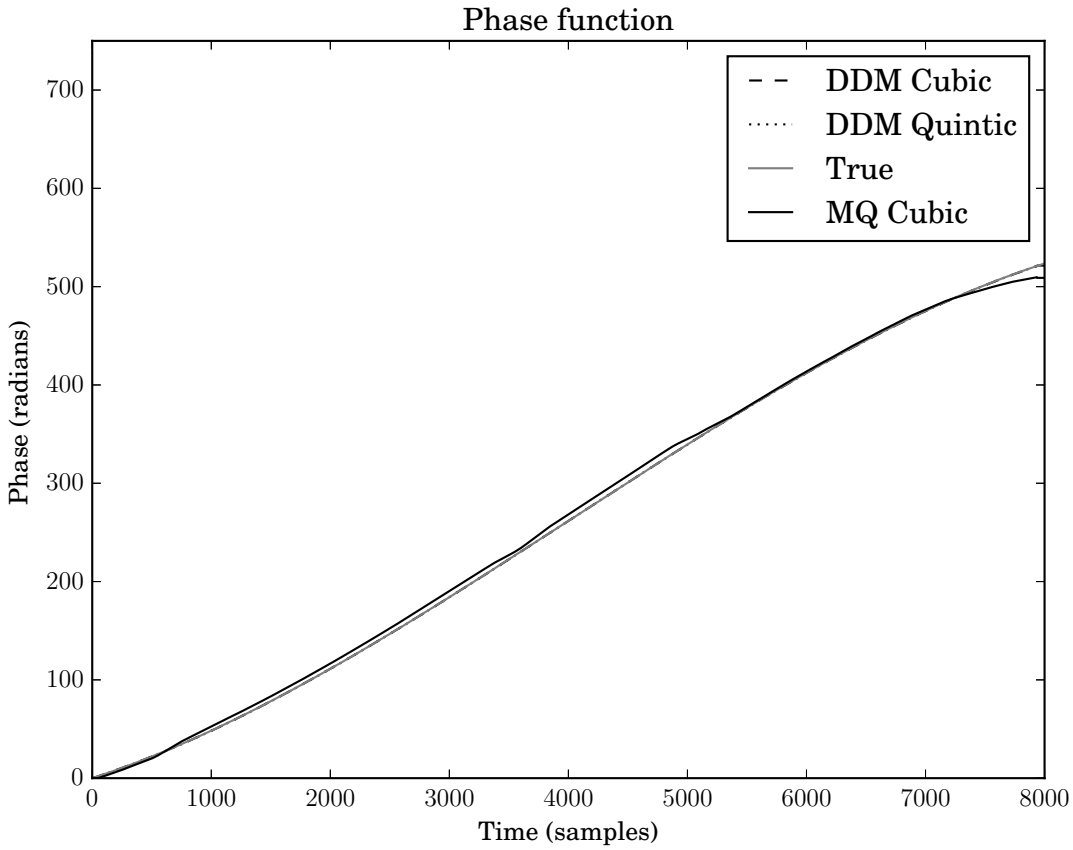

Phase function
DDM Cubic
DDM Quintic
True
MQ Cubic
Phase (radians)
Time (samples)
0
100
200
300
400
500
600
700
0
1000
2000
3000
4000
5000
6000
7000
8000

Fig. 4.11

4.6 Conclusion

Out of the three proposed methods it appears that the modified cubic interpolation method works superiorily for the signal model considered. The reduced accuracy of the higher-order quintic model can perhaps be explained by the "overfitting" of the log-amplitude function (see Figure 4.9). Indeed, even the proposed cubic model shows some overfitting in this case. From Figure 4.10 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 possiblity of errors arising from numerical accuracy when evaluating the quintic polynomials has been ruled out. We evalutated these polynomials using an implementation of Horner's method that keeps track of the error bound [12, p. 95]: the errors are negligible, see Figure 4.12 for the results.

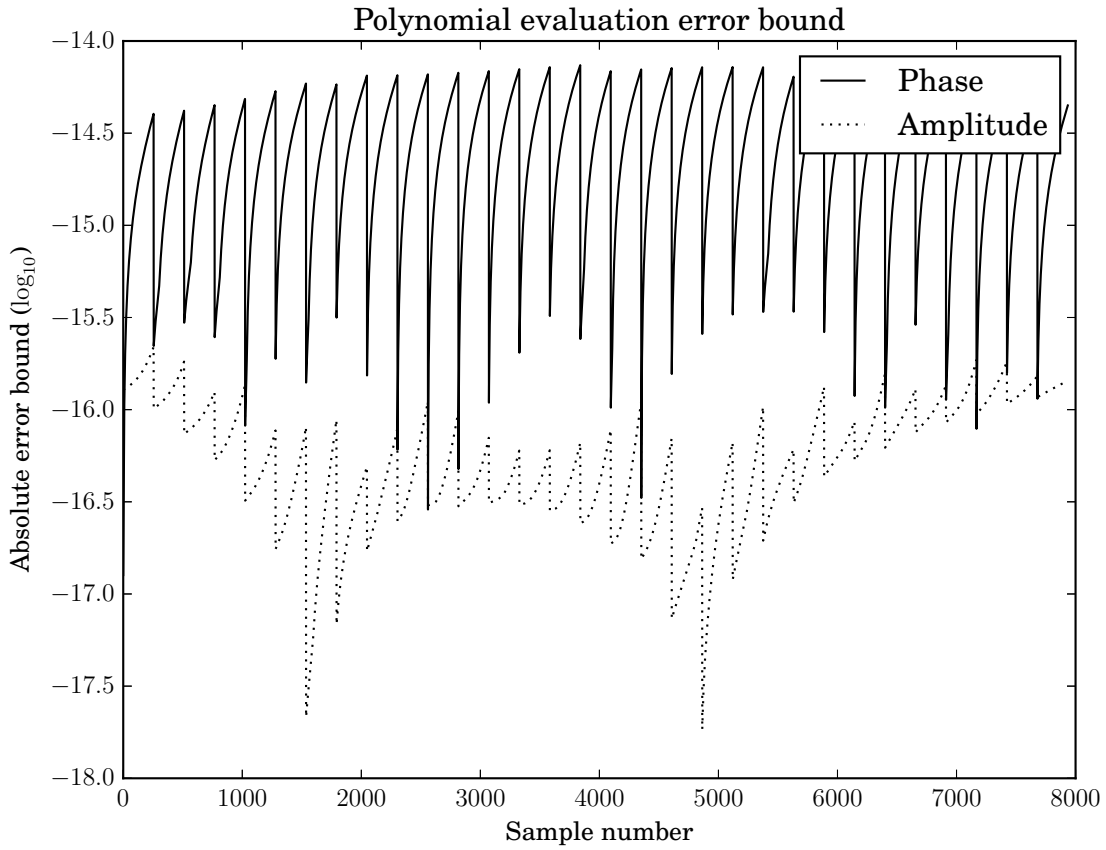


Fig. 4.12

Chapter 5

Experiment 2: Partial grouping in one frame

5.1 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.

5.2 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 1st 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.

5.3 Evaluation

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

5.4 Synthesis

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

- t time.
- f_s sampling frequency.
- N length of signal in samples.
- 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.

To incorporate inharmonicity often observed in real string instruments where the strings exhibit some stiffness, we define the *stretched* harmonic numbers as follows [32][1]

$$K_B(k) = k(1 + Bk^2)^{\frac{1}{2}} \tag{5.1}$$

Each source is synthesized using the following equation:

$$s_p(t) = \sum_{k=1}^{K_p} A_p(k,t) \exp(j(2\pi f_{0,p} t - \frac{A_{f,p}}{f_{f,p}} \cos(2\pi f_{f,p} t + \phi_{0,f,p}) K_{B_p}(k) + \phi_{0,p})) \tag{5.2}$$

where

$$A_p(k,t) = \begin{cases} \exp(a_{60,p} t + a_{k,60,p} k) \cos^2(\frac{\pi}{2}(\frac{t}{t_{\text{attack},p}} - 1)) & \text{if } t \leq t_{\text{attack},p}, \\ \exp(a_{60,p} t + a_{k,60,p} k) & \text{if } t > t_{\text{attack},p}, \\ 0 & \text{otherwise.} \end{cases} \tag{5.3}$$

$$a_{60,p} = \frac{\log(10^{-3})}{t_{60,p}} \tag{5.4}$$

$$a_{k,60,p} = \frac{\log(10^{-3})}{k_{60,p}} \tag{5.5}$$

The piecewise amplitude function is based on the amplitude function of the *Formant Wave Function (FOF)*[2] described in [30, p. 19].

The estimation of these parameters is a separate problem addressed by the DDM (see Section 3.2.2). 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 the model of these methods.

The DDM seeks signals $s_k \in \mathbb{C}$ of the following form:

$$s_k(n) = \exp(\log(A_k + \mu_k n + j(\phi_k + \omega_k n + \frac{1}{2}\psi_k n^2))) \tag{5.6}$$

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

$$\mu_{k,p}(t) = \frac{a_{60,p}}{f_s} \tag{5.7}$$

$$A_{k,p}(t) = \exp(a_{60,p} t + a_{k,60,p} k) \tag{5.8}$$

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

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

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

$$\hat{\mathbf{s}}(t) = \begin{pmatrix} \exp\left(a_{60,p}\left(t - \frac{N_w}{2f_s}\right) + a_{k,60,p}k\right)\cos^2\left(\frac{\pi}{2}\left(\frac{t - \frac{N_w}{2f_s}}{t_{\text{attack},p}} - 1\right)\right) \\ \vdots \\ \exp\left(a_{60,p}\left(t + \frac{N_w}{2f_s}\right) + a_{k,60,p}k\right)\cos^2\left(\frac{\pi}{2}\left(\frac{t + \frac{N_w}{2f_s}}{t_{\text{attack},p}} - 1\right)\right) \end{pmatrix} \tag{5.9}$$

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

$$\begin{bmatrix} 1 & \frac{-N_w}{2} \\ \vdots & \vdots \\ 1 & \frac{N_w}{2} \end{bmatrix} \begin{pmatrix} \log(A_{k,p}(t)) \\ \mu_{k,p}(t) \end{pmatrix} = \log \hat{\mathbf{s}}(t) \tag{5.10}$$

for the argument parameters (those multiplied by j in Equation (5.6))

$$\omega_{k,p}(t) = \frac{2\pi}{f_s}\left(f_{0,p} + A_{f,p}\sin\left(2\pi f_{f,p}t + \phi_{0,f,p}\right)\right)K_{B_p}(k) \tag{5.11}$$

$$\psi_{k,p}(t) = \left(\frac{2\pi}{f_s}\right)^2 A_{f,p}f_{f,p}\left(f_{0,p} + A_{f,p}\cos\left(2\pi f_{f,p}t + \phi_{0,f,p}\right)\right)K_{B_p}(k) \tag{5.12}$$

$$\phi_k(t) = \left(2\pi f_{0,p}t - \frac{A_{f,p}}{f_{f,p}}\cos\left(2\pi f_{f,p}t + \phi_{0,f,p}\right)\right)K_{B_p}(k) + \phi_{0,p} \tag{5.13}$$

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:

- $\tilde{\Psi}_{k,p}(t) \sim \mathcal{N}(\psi_{k,p}(t), \psi_{no})$
- $\tilde{\Omega}_{k,p}(t) \sim \mathcal{N}(\omega_{k,p}(t), \omega_{no})$
- $\tilde{\mu}_{k,p}(t) \sim \mathcal{N}(\mu_{k,p}(t), \mu_{no})$
- $\tilde{A}_{k,p}(t) \sim \mathcal{N}(A_{k,p}(t), A_{no})$

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 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_s of the number of true data-points. Their values are drawn from uniform distributions with boundaries $\theta_{s,min}$ and $\theta_{s,max}$, where θ is some parameter above, e.g., $\omega_{s,min}$ and $\omega_{s,max}$ for the ω parameter.

Data points are computed for the times $t = 0, \frac{H}{f_s}, \frac{2H}{f_s}, \ldots, \frac{\lfloor \frac{N}{H} \rfloor H}{f_s}$.

5.5 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 $\omega_l(t)$ (see thesis by Creager). The amplitude modulation $\mu_l(t)$ 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

$$\mathbf{x}_l(t) = \begin{pmatrix} \frac{\psi_l(t)}{\omega_l(t)} \\ \mu_l \end{pmatrix} \tag{5.14}$$

$$\mathbf{X}(t) = \begin{bmatrix} \mathbf{x}_1(t) \ldots \mathbf{x}_L(t) \end{bmatrix} \tag{5.15}$$

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 Jolliffe for a discussion about this).

Following the standard technique for producing principal components (see Jolliffe), 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

$$\mathbf{A}(t) = \mathbf{V}^T(t)\mathbf{X}(t) \tag{5.16}$$

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, hence if we are observing a random variable drawn from multiple distributions with sufficiently separated means and small enough variances, this separation will be most observable in the principal components corresponding to the greatest eigenvalues.

5.6 Preparing data for clustering

The Expectation Maximization underlying the Gaussian mixture model parameter estimation will only converge to a local maximum (see Dempster et al), 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 indeed follow a normal distribution). The initial guesses for the weights are simply $\frac{1}{N_p}$.

5.7 Clustering

Gaussian mixture model parameter estimation will be discussed in an appendix of this document. 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 following parameters: The length of the signal N is 8000 samples and the

Parameter	Source 1 value	Source 2 value
$f_{0,p}$	261.63	277.18
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
$A_{f,p}$	11.486	11.486
$f_{f,p}$	3	2

anlaysis hop size H is 256 samples.

The data-points are represented as line segments to show the frequency slope. Data points classified as coming from the same source are dark blue and cyan, respectively. Data

points classified as spurious are in black. Note that the classification is done on a frame by frame basis so it could be that data-points adjacent in time that seem to come from the same source are in different colours. This is to be addressed in the sequel. (TODO: how do we choose levels of noise to test against? My current method is to multiply the value by a random number drawn from a normal distribution + 1. The amount of change is nicely interpretable as a percentage deviating from the original

Chapter 6

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. We start with a recording of an acoustic guitar playing a_3 and a xylophone playing $f_4^\sharp$. The recordings are from [24] 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 6.4 and Figure 6.5. 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 [31, 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 [5] 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 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 6.1 to see a plot of these values and the thresholding function. The spectrogram of the mixture can be seen in Figure 6.2 and the partial paths in Figure 6.3.

A line function is fit via least-squares to each partial, as shown in Figure 6.6. 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

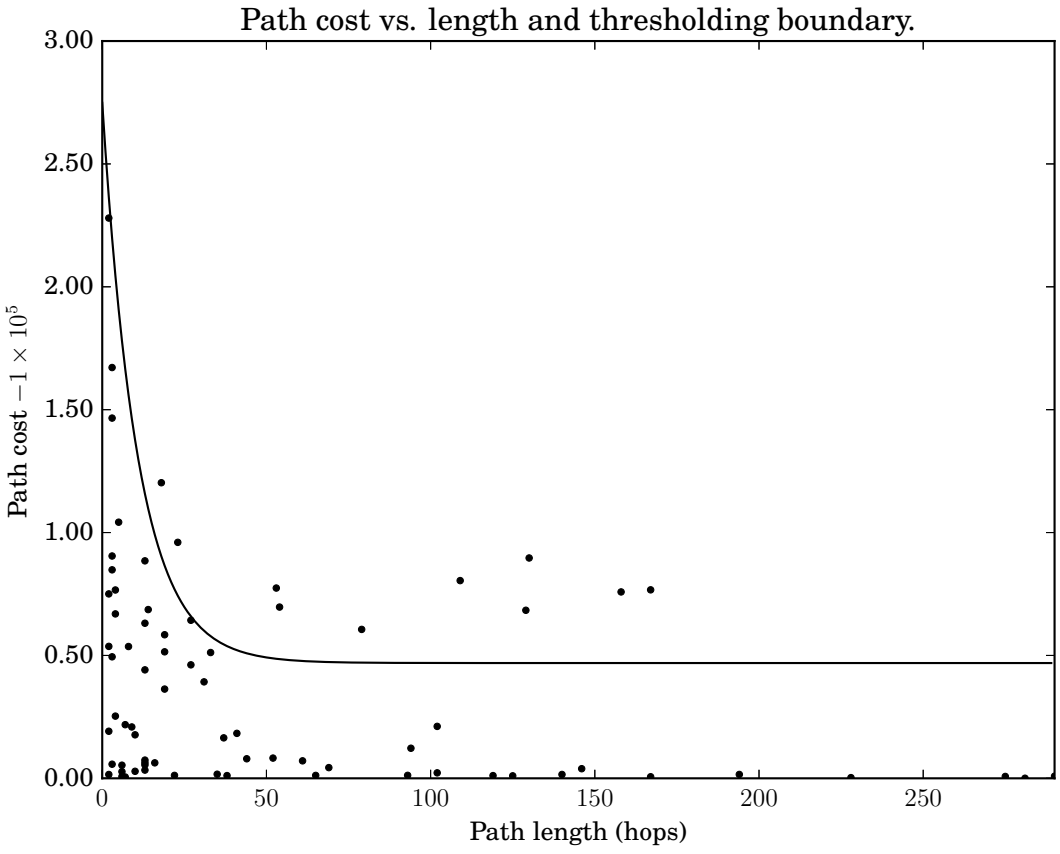


Fig. 6.1 Paths (represented as circles) are considered only if their path cost is smaller than the threshold function (represented in black).

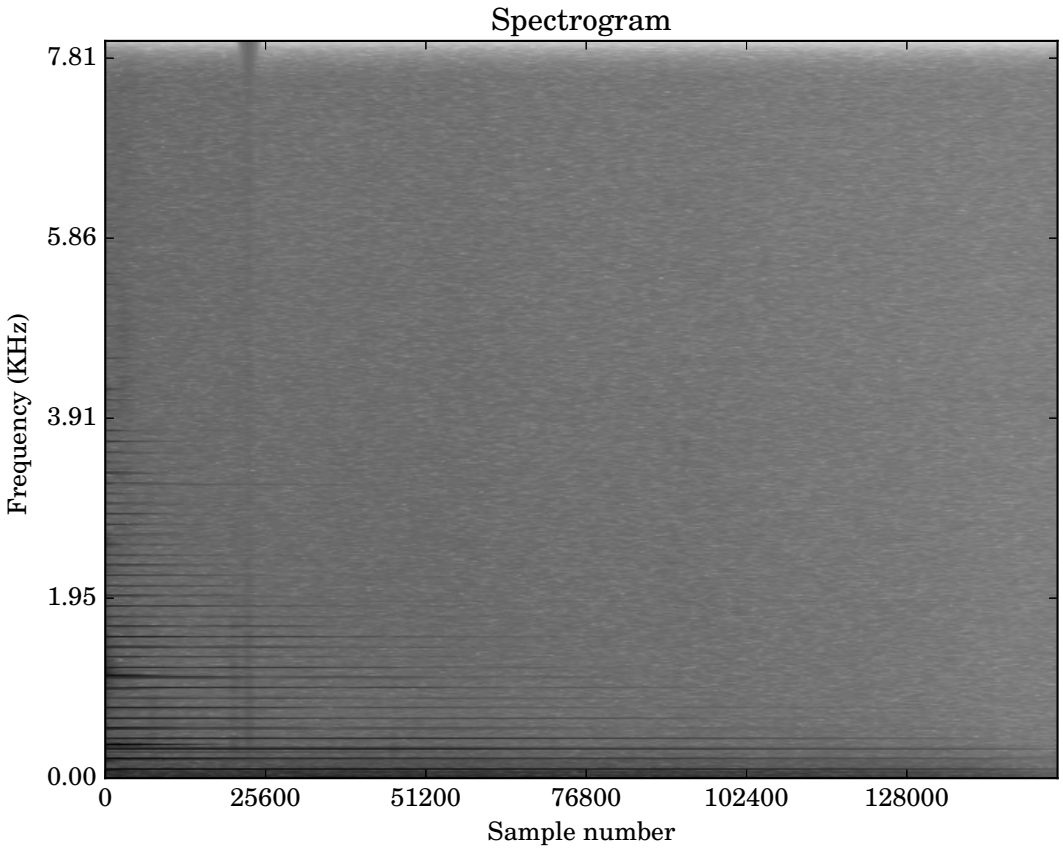

Spectrogram
7.81
5.86
3.91
1.95
0.00
Frequency (KHz)
0
25600
51200
76800
102400
128000
Sample number

Fig. 6.2

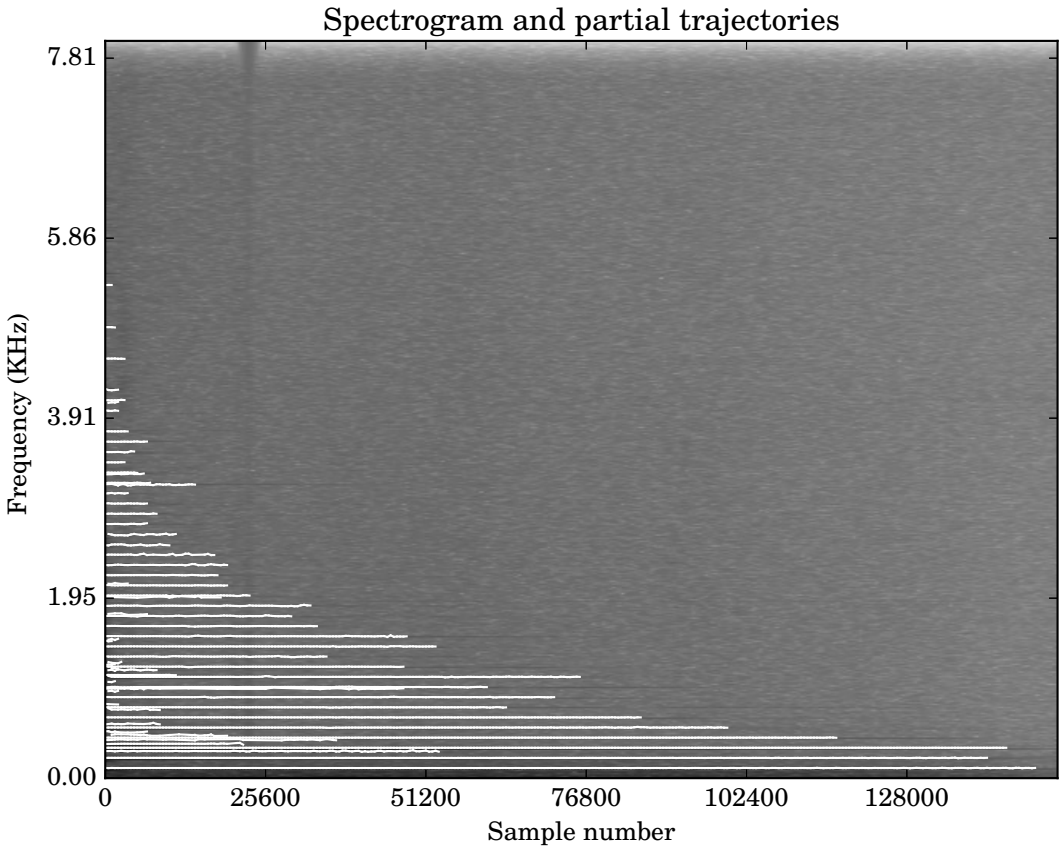

Spectrogram and partial trajectories
7.81
5.86
3.91
1.95
0.00
Frequency (KHz)
0
25600
51200
76800
102400
128000
Sample number

Fig. 6.3

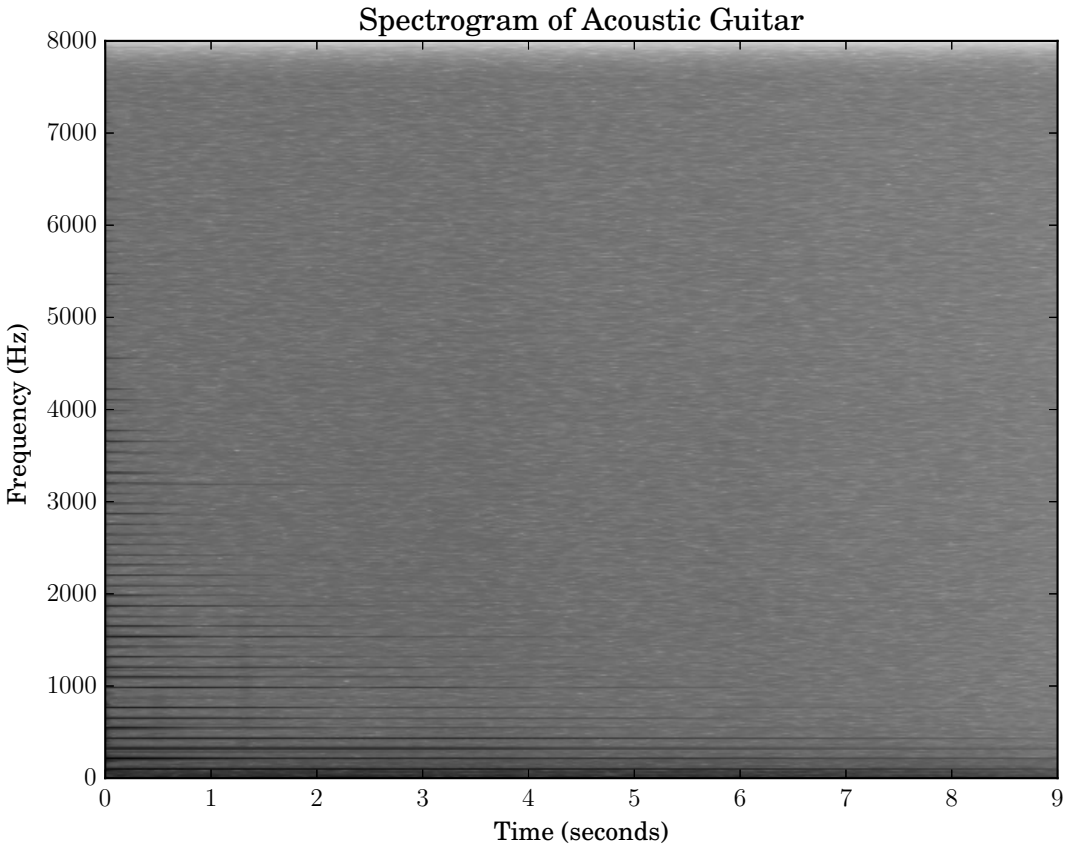

Spectrogram of Acoustic Guitar
Frequency (Hz)
8000
7000
6000
5000
4000
3000
2000
1000
0
0
1
2
3
4
5
6
7
8
9
Time (seconds)

Fig. 6.4

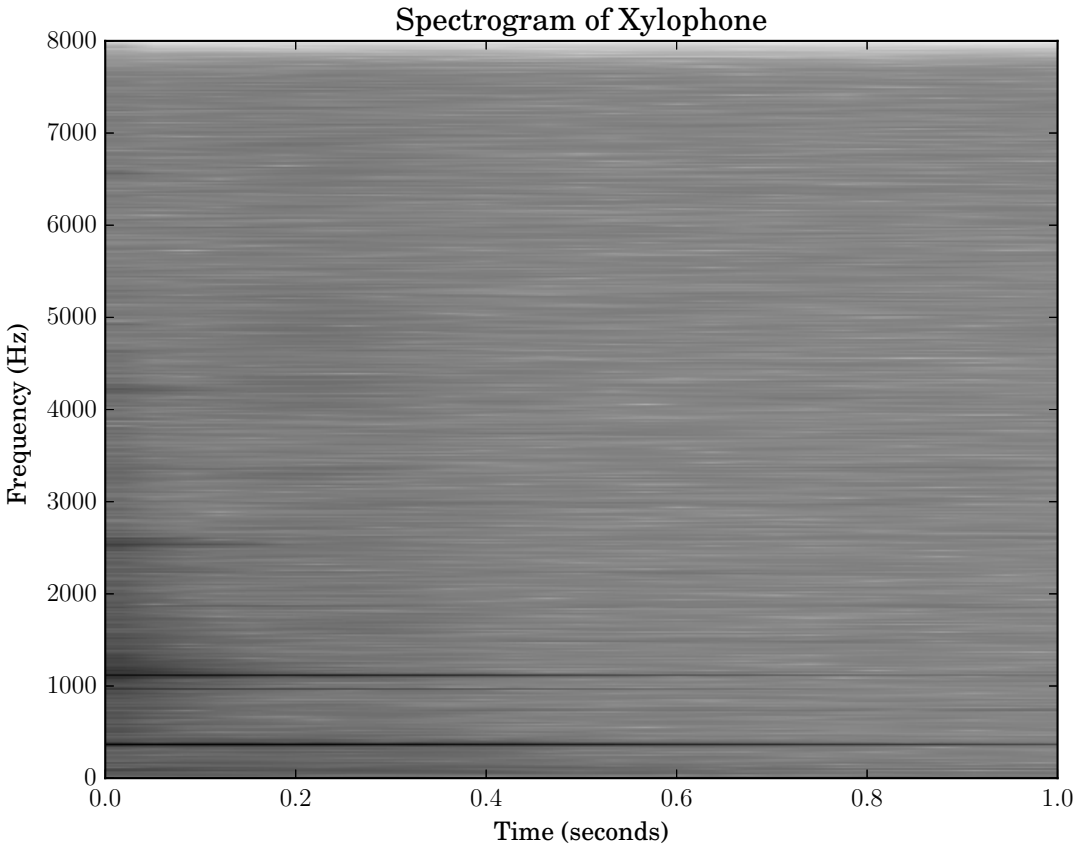

Spectrogram of Xylophone
Frequency (Hz)
8000
7000
6000
5000
4000
3000
2000
1000
0
0.0
0.2
0.4
0.6
0.8
1.0
Time (seconds)

Fig. 6.5

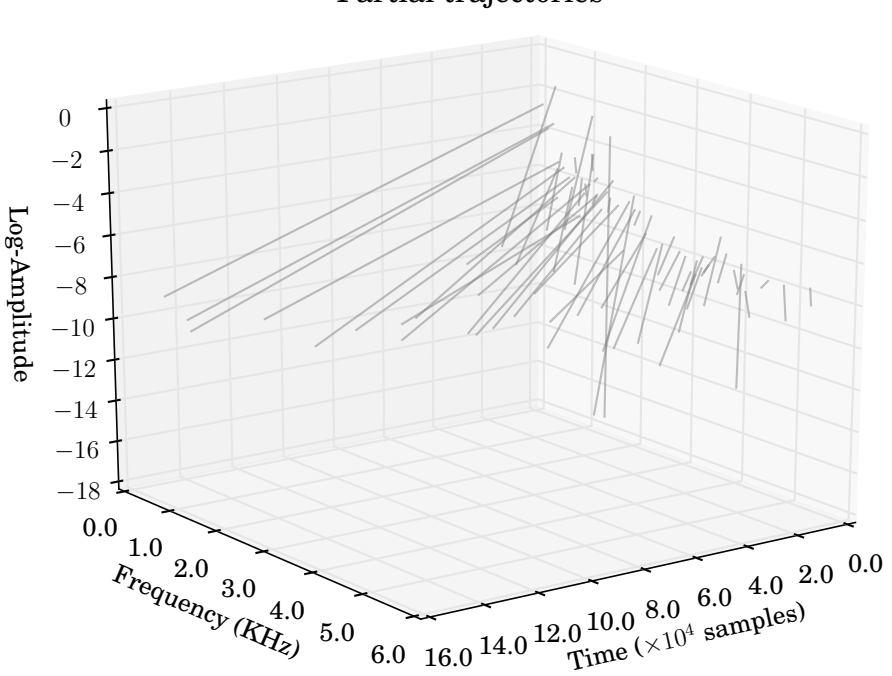

Partial trajectories
Log-Amplitude
0
−2
−4
−6
−8
−10
−12
−14
−16
−18
0.0
1.0
2.0
3.0
4.0
5.0
6.0
Frequency (KHz)
16.0
14.0
12.0
10.0
8.0
6.0
4.0
2.0
0.0
Time ($\times 10^4$ samples)

Fig. 6.6

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 6.7). The data-points have the 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

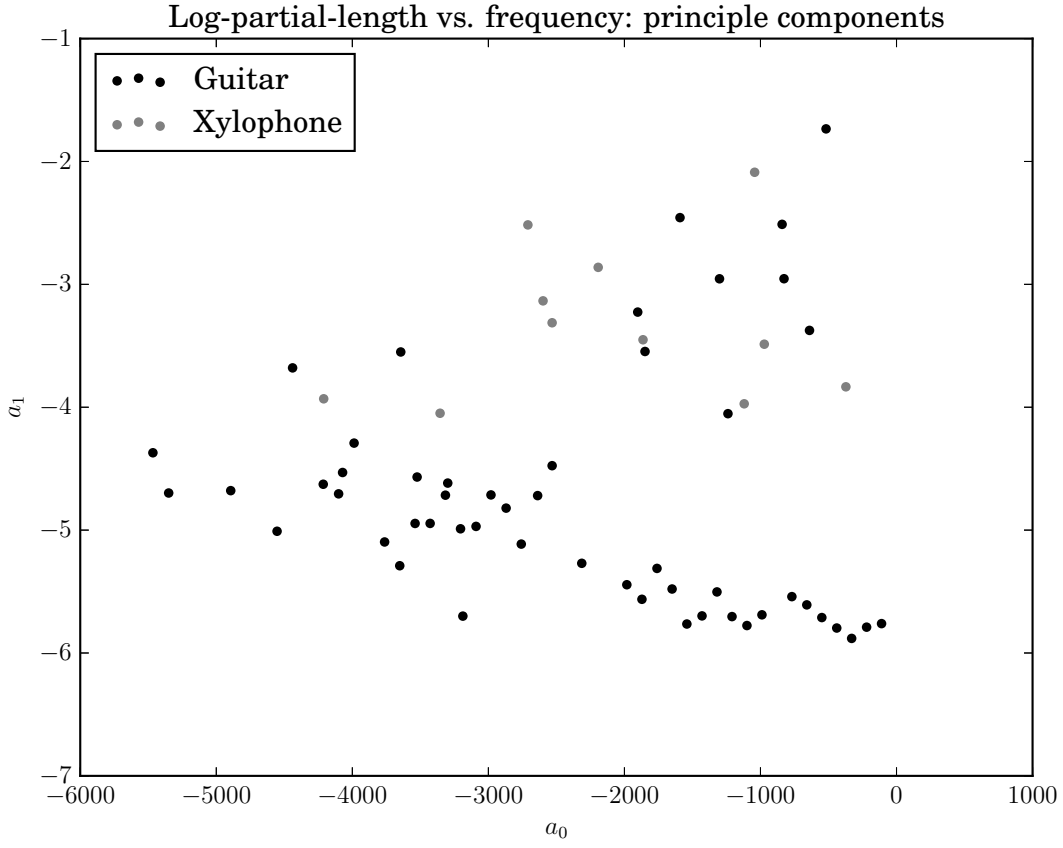


Fig. 6.7

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 [16, 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 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 $\theta_{\boldsymbol{\Sigma}_{\boldsymbol{a}}} = 1.2$. A contour plot of the resulting function is shown in Figure 6.8.

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

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

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 6.8.

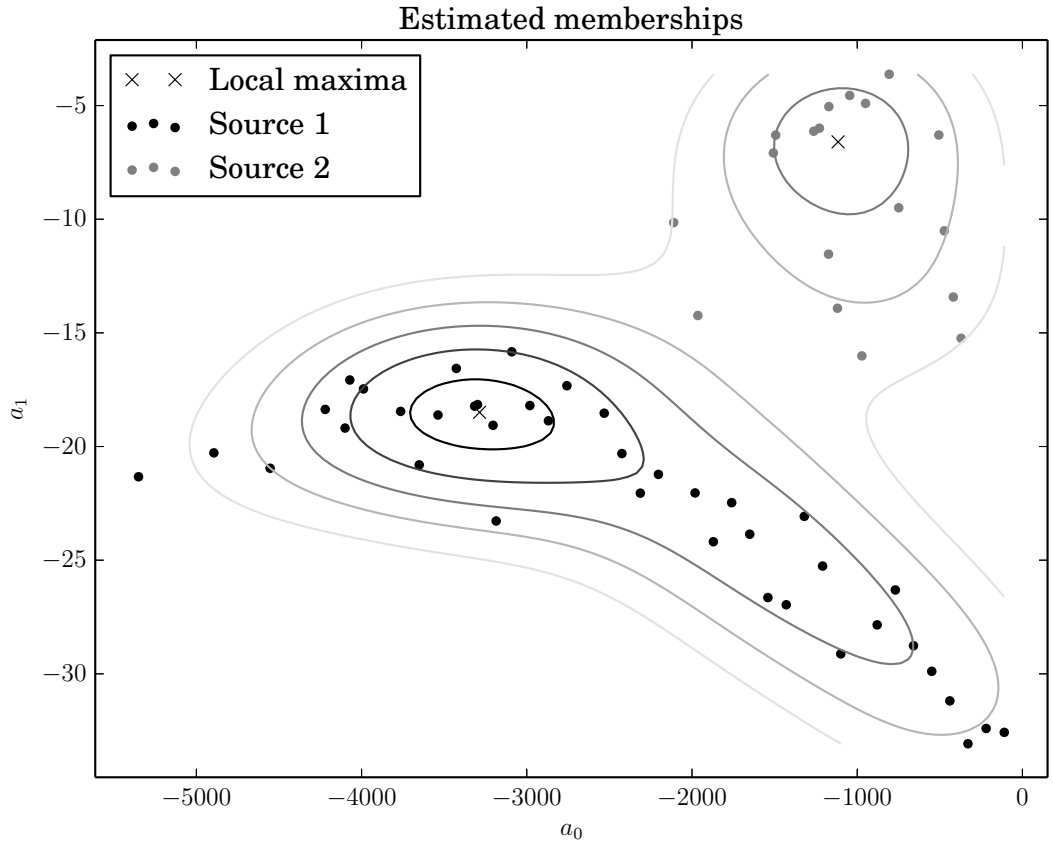


Fig. 6.8

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 6.9 and Figure 6.10.

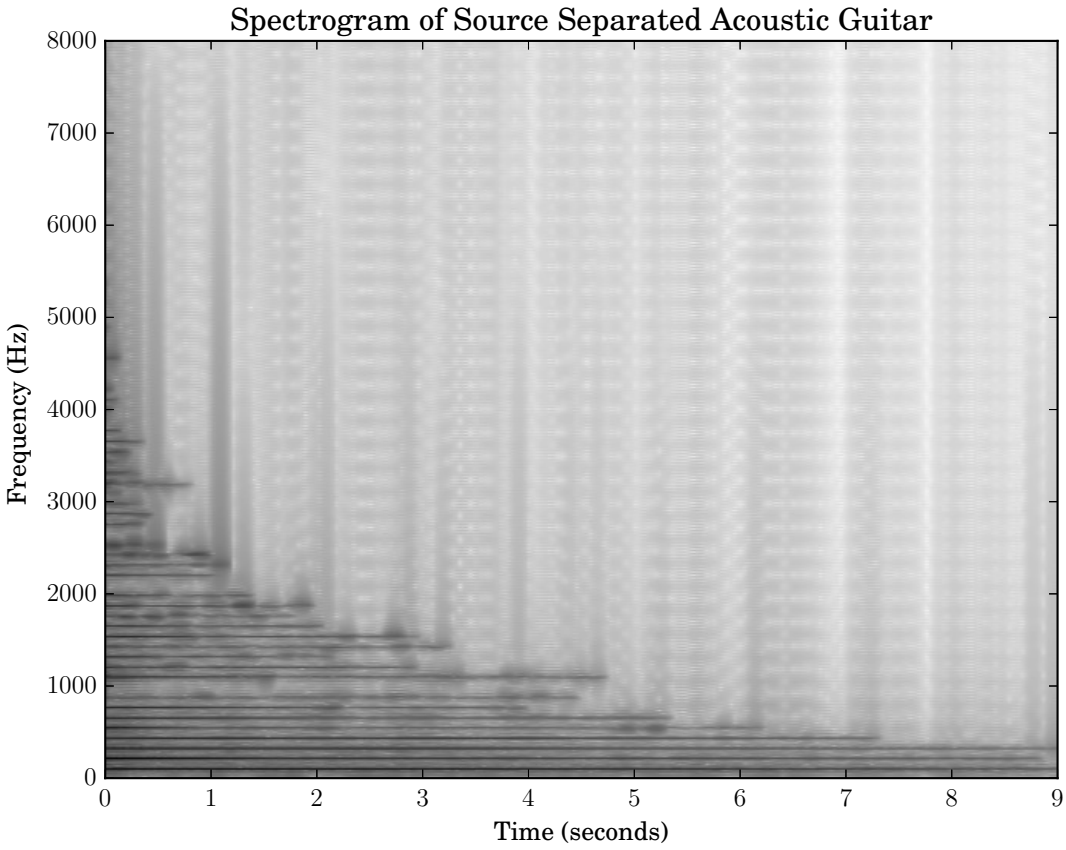

Spectrogram of Source Separated Acoustic Guitar
Frequency (Hz)
8000
7000
6000
5000
4000
3000
2000
1000
0
0
1
2
3
4
5
6
7
8
9
Time (seconds)

Fig. 6.9

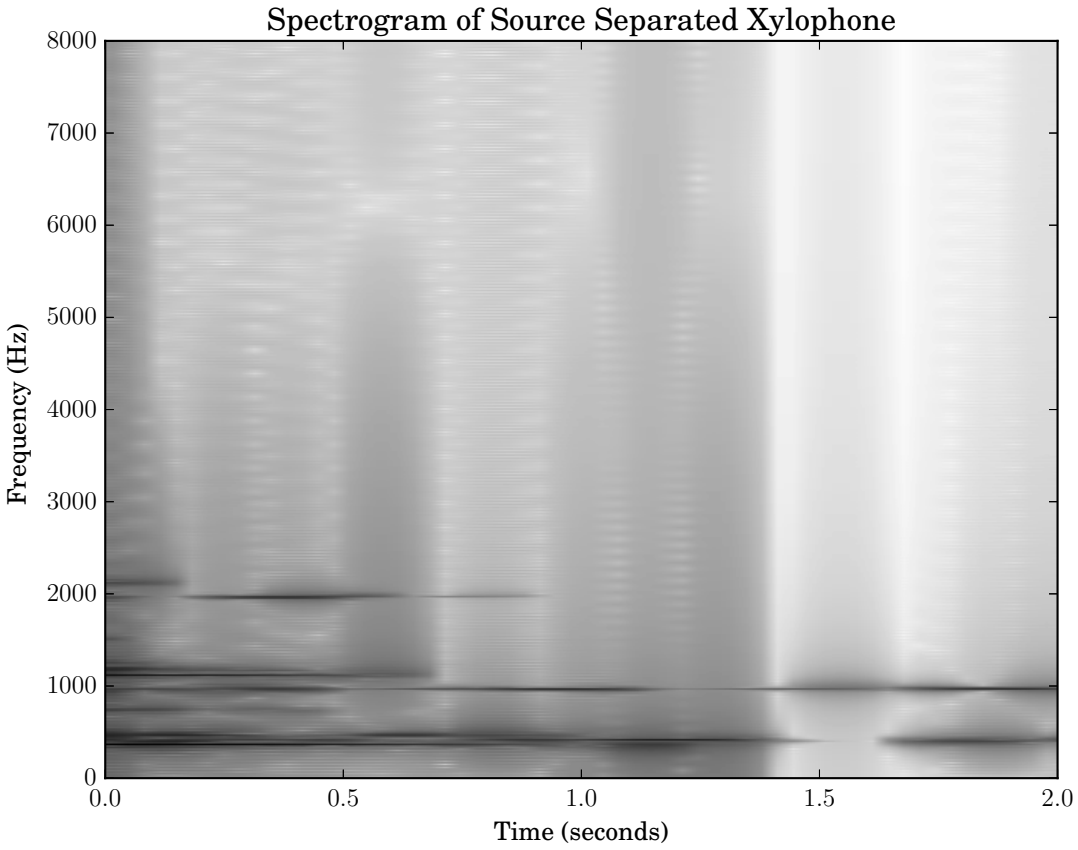

Spectrogram of Source Separated Xylophone
8000
7000
6000
5000
4000
3000
2000
1000
0
Frequency (Hz)
0.0
0.5
1.0
1.5
2.0
Time (seconds)

Fig. 6.10

6.0.1 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 6.8. Another drawback of the current technique is that it requires some tuning of the parameters θ_w and $\theta_{\mathbf{\Sigma}_a}$. Furthermore, 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 – the techniques discussed in Section **??** use fundamental frequency estimation or harmonicity assumptions in their models.

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

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] 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.

[4] 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.

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

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

[7] 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.

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

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

[10] 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.

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

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

[13] 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.

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

[15] 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.

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

[17] 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.

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

[19] 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.

[20] 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.

[21] 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.

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

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

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

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

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

[27] 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.

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

[29] 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.

[30] 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.

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

[32] Julius O Smith. *Physical Audio Signal Processing*. `http://ccrma.stanford.edu/~jos/pasp/`, accessed July 20, 2016. online book, 2010 edition.

[33] 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.

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