ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Signal and Information
Processing Laboratory
Institut für Signal- und
Informationsverarbeitung

Autumn Semester 2019 Prof. H.-A. Loeliger

Semester Thesis

Sound Source Localization via Onset Detection

Gustavo Cid Ornelas

Advisor: Elizabeth Ren

Abstract

Humans can perform sound source localization in a seemingly effortless manner, even in complex environments, where there are interfering sources and reverberation. There are different approaches aiming to enable machines to do the same. This work explores an algorithm that performs sound source localization via onset detection making use of autonomous linear state space models (LSSMs) as a modeling framework. The proposed algorithm has three main stages. In the first one, it detects the occurrence of sound onsets by locally fitting an onset model to the binaural audio signal and computing a local cost ratio (LCR), which is sparse and judges whether or not a sound onset is present. In the second stage, the LCR is locally approximated by a polynomial, which facilitates the next steps of signal processing. Finally, in the third stage, the delay estimate is produced and a relationship between the delay estimate and the sound source location in the horizontal plane is learned. The algorithm is validated on speech signals, illustrating it's working principles and computational efficiency when it comes to model fitting with recursive cost computations.

Contents

1 Introduction **1**

1.1 Sound source localization 1

1.2 Problem setting 1

2 Localized Autonomous Linear State Space Models **3**

2.1 Signal models 4

2.1.1 Exponentials 4

2.1.2 Polynomials 4

2.1.3 Sinusoids 4

2.2 Windows 5

2.3 Combining LSSMs 5

2.3.1 Linear combination of two LSSMs 5

2.3.2 Product of two LSSMs 6

2.4 Fitting a LSSM 6

3 Onset Detection **8**

3.1 Onset Models 9

3.1.1 Decaying sinusoid 9

3.1.2 Gammatone 9

3.2 Fitting Onset Models 10

3.2.1 Fitting a decaying sinusoid 11

3.2.2 Fitting a gammatone 11

3.3 Log-Cost Ratio (LCR) 12

4 Local Polynomial Fitting **14**

4.1 Polynomial LSSM 14

4.2 Locally Fitting Polynomials 15

5 Delay Estimation **17**
5.1 Algorithm 17

6 Experimental Results **20**
6.1 Dataset 20
6.2 Onset detection 20
6.3 Local polynomial fitting and delay estimation 23

7 Conclusion **29**

A Derivation of the Recursions for Onset Detection **31**

B Derivation of the Recursions for the Local Polynomial Fit **35**

C Experimental Results for a Female Speech Signal **37**

Bibliography **41**

List of Figures **43**

List of Tables **45**

Chapter 1

Introduction

1.1 Sound source localization

On the one hand, humans have little difficulty localizing sound sources, even in an environment with reverberation and interfering sources. On the other hand, machines have difficulties determining the azimuth of the sound source in the horizontal plane, even when making use of multiple microphones [1].

Humans perform such tasks based on a complex auditory mixture that arrives at their binaural sensors, i.e., their ears. To do so, they rely mainly on two different cues. The first are interaural level differences (ILDs), which correspond to the difference in the loudness and frequency distribution of the audio signal arriving at the two ears, hinting at the distance between the sound source and the listener. The second are interaural time differences (ITDs), which correspond to the difference in the time of arrival of the sound in each ear.

The idea of enabling machines to use these cues to perform sound source localization is not new. There are many proposed approaches to this problem, ranging from the use of cross-correlation methods to deep learning [2, 3].

This work explores an algorithm that performs sound source localization by estimating the ITD, using the toolbox of autonomous linear state space models (LSSMs).

1.2 Problem setting

Figure 1.1 illustrates the block diagram of the proposed algorithm. The input is a binaural audio signal and the output is the estimated difference

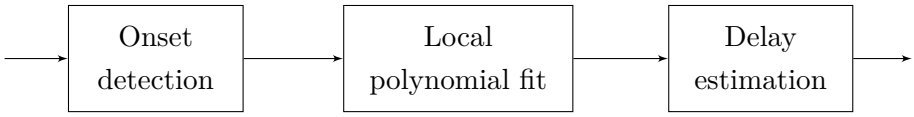


Figure 1.1: Proposed algorithm block diagram.

in the time of arrival between the two audio signals. This delay estimate is, then, mapped to the azimuth of the sound source in the horizontal plane.

The algorithm starts by detecting sound onsets, which indicate the first-arriving sound. The use of the sound onsets for the task at hand is justified by the precedence effect [4]. The onsets correspond to the sound signal that travelled the direct path from the sound source and can, thus, give accurate timing information related to the localization of the source. The output of the onset detection stage are log-cost ratio (LCR) signals, which are sparse signals that accurately represent if a sound onset was detected at each audio sample. The LCRs are, then, locally approximated by polynomials, to facilitate the last stage of processing, which corresponds to the delay estimation.

This work is organized as follows: in Chapter 2, the LSSM modeling framework used is presented; in Chapter 3, the first block of Figure 1.1 is thoroughly explained, showing how to go from the raw audio signals to the LCRs; in Chapter 4, the local polynomial fit is explained; in Chapter 5, the delay estimation algorithm based on the coefficients of the local polynomial fit is explored; Chapter 6 illustrates the algorithm in action for a sample audio signal and the process of tuning the algorithm parameters is discussed; finally, in Chapter 7, the conclusions are drawn and the future work direction is established.

Chapter 2

Localized Autonomous Linear State Space Models

In this chapter, we present the modeling framework that is extensively used throughout the project. The key idea is to locally fit autonomous linear state space models (LSSM) – which are state space models without an input – to a given signal. With the state space representation, we are able to perform a multitude of tasks that are ubiquitous in signal processing, such as signal detection, separation and reconstruction. The use of LSSMs is also justified by the fact that there are computationally efficient ways to fit such signals using recursive cost computations.

We start with LSSMs that run backwards in time, starting from instant k. The m-channel output signal $\tilde{y}_i$ at time $i \leq k$ of a LSSM of order n is determined by

$$x_i = Ax_{i+1} \tag{2.1}$$

$$\tilde{y}_i = Cx_i, \tag{2.2}$$

where $A \in \mathbb{R}^{n\times n}$ is the *state transition matrix*, $C \in \mathbb{R}^{m\times n}$ is the *output matrix*, $x_i \in \mathbb{R}^n$ is the *state vector* and $\tilde{y}_i \in \mathbb{R}^m$ is the *output signal*. Throughout this work, we fix the initial state $x_k = s$ and vary C. The output signal at instant i is given by (2.3), which represents the trajectory defined by the LSSM.

$$\tilde{y}_i = CA^{k-i}s \tag{2.3}$$

2.1 Signal models

In this section, we show that it is possible to model a rich set of signals with LSSMs. The signal models presented here serve as building blocks for the more complex signals explored later on in this work.

2.1.1 Exponentials

An exponential function $\tilde{y}_i = a\rho^\ell$, with $\rho \in \mathbb{R}$ and $a \in \mathbb{R}^m$ can be represented by

$$A = \rho \tag{2.4}$$

$$s = 1. \tag{2.5}$$

Note that the output matrix (in this case, a scalar) $C = a$ determines the signal amplitude.

2.1.2 Polynomials

Polynomials of the form $\tilde{y}_i = a_0 + a_1\ell + \cdots + a_N\ell^N$, with degree N can be obtained with a LSSM of order $N+1$ with parameters given by equations (2.6) and (2.7).

$$A = \begin{pmatrix} 1 & 1 & 0 & \dots & 0 \\ 0 & 1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 1 & 1 \\ 0 & \dots & \dots & 0 & 1 \end{pmatrix} \tag{2.6}$$

$$s = \begin{pmatrix} 0 & \dots & 0 & 1 \end{pmatrix}^\intercal \tag{2.7}$$

The output vector C determines the coefficients of the polynomial.

2.1.3 Sinusoids

A sinusoid of the form $\tilde{y}_i = a\cos(\Omega\ell) + b\sin(\Omega\ell)$, with $a, b \in \mathbb{R}$ and frequency $\Omega \in (0, 2\pi) \setminus \pi$ can modeled by the LSSM represented in equations (2.8) and (2.9).

$$A = \begin{pmatrix} \cos(\Omega) & -\sin(\Omega) \\ \sin(\Omega) & \cos(\Omega) \end{pmatrix} \tag{2.8}$$

$$s = \begin{pmatrix} 1 & 0 \end{pmatrix}^\intercal \tag{2.9}$$

In this case, the output vector $C = (a\ b)$ determines the parameters of the sinusoid.

It is also worth noting that linear combination of sinusoids with different frequencies are easily realizable by using a block-diagonal matrix A with diagonal blocks given by (2.8).

2.2 Windows

Instead of using the LSSM in the whole signal domain, we only use it locally, within a window centered around the current time index. The window, then, runs over the whole signal. This is done to detect local signal patterns. In this section, we present some useful windows.

The windows considered in this work are the rectangular window, defined by

$$w_\ell = \begin{cases} 1 & \text{if } a \leq \ell \leq b \\ 0 & \text{otherwise,} \end{cases} \tag{2.10}$$

the one-sided exponential window

$$w_\ell = \lambda^\ell, \tag{2.11}$$

for $\lambda \in (0, 1)$ and the gamma window

$$w_\ell = \begin{cases} \beta|\ell|^{\nu-1}\gamma^{|\ell|} & \text{if } \ell \leq 0 \\ 0 & \text{if } \ell > 0, \end{cases} \tag{2.12}$$

where ν is a positive integer and $\beta > 0$ is a scale factor.

2.3 Combining LSSMs

We can work with more complex signal models by adding or multiplying together simpler LSSMs. The properties presented in this section are useful when dealing with this kinds of combinations.

2.3.1 Linear combination of two LSSMs

Let $\tilde{y}_i^{(1)}$ be the signal generated by the LSSM of order n_1 defined by $\{A_1, C_1, s_1\}$ and $\tilde{y}_i^{(2)}$, generated by $\{A_2, C_2, s_2\}$ of order n_2. Given these two

LSSMs, the LSSM for $\tilde{y}_i = \alpha\tilde{y}_i^{(1)} + \beta\tilde{y}_i^{(2)}$, $\alpha, \beta \in \mathbb{R}$, of order $n_1 + n_2$ is given by equations (2.13), (2.14) and (2.15).

$$A = \text{diag}(A_1, A_2) \tag{2.13}$$

$$s = \begin{pmatrix} s_1^\intercal & s_2^\intercal \end{pmatrix} \tag{2.14}$$

$$C = \begin{pmatrix} \alpha C_1 & \beta C_2 \end{pmatrix} \tag{2.15}$$

2.3.2 Product of two LSSMs

Let $\tilde{y}_i^{(1)}$ be the signal generated by the LSSM of order n_1 defined by $\{A_1, C_1, s_1\}$ and $\tilde{y}_i^{(2)}$, generated by $\{A_2, C_2, s_2\}$ of order n_2. Given these two LSSMs, the LSSM for $\tilde{y}_i = \tilde{y}_i^{(1)} \otimes \tilde{y}_i^{(2)}$, of order $n_1.n_2$ is given by equations (2.16), (2.17) and (2.18).

$$A = A_1 \otimes A_2 \tag{2.16}$$

$$s = s_1 \otimes s_2 \tag{2.17}$$

$$C = C_1 \otimes C_2, \tag{2.18}$$

where $\otimes$ denotes the Kronecker product.

2.4 Fitting a LSSM

Let y_i be a given multi-channel signal at instant i. We fit a LSSM to the signal y_i by minimizing the cost function defined by (2.19), which is the squared error weighted by the window function $w_i \in \mathbb{R}$.

$$\begin{aligned} J_k(C) &= \sum_{i=1}^{k} w_i \|y_i - \tilde{y}_i\|_2^2 \\ &= \sum_{i=1}^{k} w_i \left\| y_i - CA^{k-i}s \right\|_2^2 \end{aligned} \tag{2.19}$$

As mentioned before, in this work, when working with LSSMs, we fix the initial state and we vary C. It is possible to think of minimizing (2.19) in the following way: we fix A and s, which fully determine the family of functions that we are fitting; then, we are looking for the output matrix C –

which contains the model's parameters – that minimizes the cost. Therefore, we look for an output matrix such that

$$\hat{C} = \underset{\hat{C} \in \mathcal{C}}{\operatorname{argmin}} J_k(C), \tag{2.20}$$

where $\mathcal{C}$ is the feasible set.

This minimization problem can be efficiently solved recursively. This approach will be explored in more details in the next chapters, in which we are concerned with fitting specific LSSMs.

Chapter 3

Onset Detection

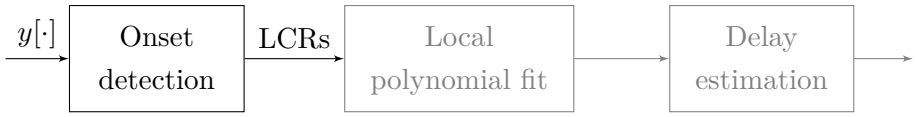


Figure 3.1: Onset detection block, where $y[\cdot]$ corresponds to the binaural audio signal.

In this project, we explore two different onset models for onset detection. The first one is a decaying sinusoid. The second, is a gammatone filter. In this chapter, we present their corresponding LSSMs and the recursions used to fit them. Then, we define the log-cost ratio (LCR), which is the measure used to detect the presence of the onsets.

In practical terms, this Chapter focuses on the highlighted block in Figure 3.1.

Figure 3.2 depicts the onset models considered.

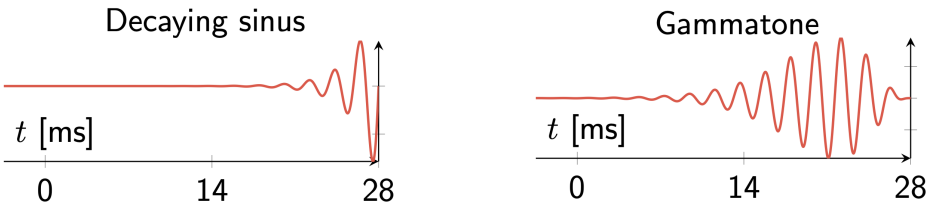


Figure 3.2: Decaying sinusoid and gammatone filter signals

3.1 Onset Models

3.1.1 Decaying sinusoid

A decaying sinusoid is defined by

$$\tilde{y}_i = \alpha\gamma^{\ell}\cos(\Omega\ell + \phi). \tag{3.1}$$

We note, thus, that the decaying sinusoid is the product of a sinusoid and an exponential function. We can, then, use the product of two LSSMs property presented in Section 2.3.2 and the corresponding LSSM models, which yields in:

$$A = \gamma\begin{pmatrix}\cos(\Omega) & -\sin(\Omega)\\ \sin(\Omega) & \cos(\Omega)\end{pmatrix} \tag{3.2}$$

$$s = \begin{pmatrix}1 & 0\end{pmatrix}^{\mathsf{T}} \tag{3.3}$$

$$C = \alpha\begin{pmatrix}\sin(\phi) & -\cos(\phi)\end{pmatrix}. \tag{3.4}$$

From equations (3.2) and (3.4), it is possible to note that the decay and the frequency of the decaying sinusoid are encoded in the matrix A, while the amplitude and the phase are encoded in C.

3.1.2 Gammatone

The gammatone filter response is given by

$$\tilde{y}_i = \alpha\ell^3\gamma^{\ell}\cos(\Omega\ell + \phi). \tag{3.5}$$

Therefore, the gammatone filter is the product of a 3^{rd} degree polynomial and a decaying sinusoid. Using, again, the product of two LSSMs property from Section 2.3.2, we can arrive at the LSSM for the gammatone filter.

The polynomial part ℓ^3 is defined by

$$A_1 = \begin{pmatrix}1 & 1 & 0 & 0\\ 0 & 1 & 1 & 0\\ 0 & 0 & 1 & 1\\ 0 & 0 & 0 & 1\end{pmatrix} \tag{3.6}$$

$$s_1 = \begin{pmatrix}0 & 1 & 6 & 6\end{pmatrix}^{\mathsf{T}} \tag{3.7}$$

$$C_1 = \begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix}. \tag{3.8}$$

The decaying sinusoid part, $\alpha\gamma^{\ell}\cos(\Omega\ell+\phi)$, has LSSM $\{A_2, C_2, s_2\}$ given by equations (3.2) and (3.4). The resulting LSSM is found by directly applying the properties presented in Section 2.3.2, which results in a LSSM or order $n = 8$.

3.2 Fitting Onset Models

We start with the cost formulation, as in (2.19). The window functions we consider can also be represented as LSSMs $\{A_w, C_w, s_w\}$ of order n_w, hence, the cost can be written as

$$J_k(C) = \sum_{i=1}^{k} C_w A_w^{k-i} s_w \left\| y_i - CA^{k-i}s \right\|_2^2. \tag{3.9}$$

In Appendix A, we show that the cost function can be re-written as

$$J_k(C) = C_w \vec{\chi}_k - 2\mathrm{tr}((C \otimes C_w)\vec{\zeta}_k) + \mathrm{tr}((C \otimes C_w)\vec{s}_k C^{\mathsf{T}}), \tag{3.10}$$

where $\vec{\chi}_k \in \mathbb{R}^{n_w}$, $\vec{\zeta}_k \in \mathbb{R}^{nn_w \times m}$ and $\vec{s}_k \in \mathbb{R}^{nn_w \times n}$ can be efficiently with the following forward recursions

$$\begin{aligned} \vec{\chi}_k &= \sum_{i=1}^{k} A_w^{k-i} s_w \|y_i\|_2^2 \\ &= A_w \vec{\chi}_{k-1} + s_w \|y_k\|_2^2 \end{aligned} \tag{3.11}$$

$$\begin{aligned} \vec{\zeta}_k &= \sum_{i=1}^{k} (A \otimes A_w)^{k-i} (s \otimes s_w) y_i^{\mathsf{T}} \\ &= (A \otimes A_w)\vec{\zeta}_{k-1} + (s \otimes s_w) y_k^{\mathsf{T}} \end{aligned} \tag{3.12}$$

$$\begin{aligned} \vec{s}_k &= \sum_{i=1}^{k} (A \otimes A_w)^{k-i} (s \otimes s_w) s^{\mathsf{T}} (A^{k-i})^{\mathsf{T}} \\ &= (A \otimes A_w)\vec{s}_{k-1} A^{\mathsf{T}} + (s \otimes s_w) s^{\mathsf{T}}. \end{aligned} \tag{3.13}$$

The cost function can be further simplified and re-written as

$$J_k(C) = \vec{\kappa}_k - 2\mathrm{tr}(C\vec{\xi}_k) + \mathrm{tr}(C\vec{W}_k C^{\mathsf{T}}), \tag{3.14}$$

where $\vec{\kappa}_k \in \mathbb{R}$, $\vec{\xi}_k \in \mathbb{R}^{n \times m}$ and $\vec{W}_k \in \mathbb{R}^{n \times n}$ are defined as

$$\vec{\kappa}_k = C_w \vec{\chi}_k \tag{3.15}$$

$$\{\vec{\xi}_k\}_{j,p} = \mathrm{tr}((P_{p,j}^{(m,n)} \otimes C_w)\vec{\zeta}_k) \tag{3.16}$$

$$\{\vec{W}_k\}_{j',p'} = \mathrm{tr}((P_{p',j'}^{(n,n)} \otimes C_w)\vec{s}_k), \tag{3.17}$$

where $P_{i,j}^{(m,r)}$ is the $m \times r$-matrix with a one at index (i, j) and zeros everywhere else.

In this work, when fitting a LSSM, we encounter two scenarios. The first one consists in an unconstrained optimization problem, where the feasible set $\mathcal{C}$ is equal to $\mathbb{R}^{m\times n}$. In this case, we can simply derive (3.14) with respect to C and set it to zero, which yields in

$$\hat{C} = (\vec{W}_k^{-1}\vec{\xi}_k)^\intercal. \tag{3.18}$$

The second scenario consists in a constrained optimization setting, which arises when we wish to optimize only part of C. In this case, the feasible set $\mathcal{C} = \{\tilde{C}P : \tilde{C} \in \mathbb{R}^{m\times d}\}$, $P \in \mathbb{R}^{d\times n}$ is fixed. We can replace, then, C in (3.14) by $\tilde{C}P$ and find the optimal C as

$$\hat{C} = ((P\vec{W}_k P^\intercal)^{-1} P\vec{\xi}_k)^\intercal P. \tag{3.19}$$

3.2.1 Fitting a decaying sinusoid

Fitting a decaying sinusoid consists in finding the best phase and amplitude of the sinusoid, both of which are encoded in C. Therefore, we are dealing with the unconstrained optimization situation and the optimal fit is given by (3.18).

3.2.2 Fitting a gammatone

In the case of the gammatone, the envelope $\gamma^\ell \ell^3$ defines the shape of the onset model and we want it to be fixed. Fitting in this case consists of optimizing over the phase and amplitude of the sinusoidal part and leaving the envelope fixed. Since, in this case, $C = C_1 \otimes C_2$, as described in the previous section, and we want to optimize only over C_2 (the sinusoidal part),

we can re-write C as

$$\begin{aligned} C &= \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} \otimes C_2 \\ &= \begin{pmatrix} C_2 & 0_{1\times 6} \end{pmatrix} \\ &= C_2 P, \end{aligned} \tag{3.20}$$

where $P = \begin{pmatrix} \mathbb{1}_2 & 0_{2\times 6} \end{pmatrix}$ and the optimal fit is given by (3.19).

3.3 Log-Cost Ratio (LCR)

The basic idea is that we want to detect the occurrence of sound onsets in a certain audio signal. For that purpose, we introduce the log-cost ratio (LCR). We first fit the onset model of choice to the whole audio signal, then, we can measure the goodness of fit by comparing the cost of our resulting fit with that of a zero-signal model. At the sound onsets, we hope that our fit will be much better – i.e., will have a lower cost – than the zero-signal. At the other parts of the signal, our onset model will probably have a bad fit if compared to the zero-signal. The LCR at time step k is then defined as

$$\mathrm{LCR}_k = -\frac{1}{2} \log \frac{\min_{C\in\mathcal{C}} J_k(C)}{J_k(0)}. \tag{3.21}$$

The larger the LCR_k, the better the fit of the onset model is if compared to the zero-signal. It is also possible to note that the LCR_k is always non-negative.

In order to robustly detect the sound onsets, the LCR should have some desirable characteristics. Firstly, the value of the LCR should be significantly increased, forming a peak, when a sound onset is occurring. Second, it should be sparse and be mostly zero when there is no onset in the audio signal. In this work, we experimented with two different heuristic metrics that should indicate whether the LCR is robustly detecting the sound onsets. These metrics are of utmost importance for the experimental results of this project.

The first metric evaluated is the bandwidth of the Discrete Fourier Transform (DFT) of the LCR signal. We expect signals that have the characteristics described in the previous paragraph to have smaller bandwidths, since the majority of the undesirable characteristics of the LCR are associated to high frequency components in the Fourier domain.

The second metric we experimented with is the entropy. First, we normalize the samples from the LCR to sum to one, as a probability mass

function. Then, with the normalized LCR, we calculate the entropy as in (3.22). We can interpret the entropy as a measure of uncertainty and we expect sparse signals to be less uncertain, containing just a few peaks.

$$H(p) = -\sum_i p_i \log(p_i) \tag{3.22}$$

Chapter 4

Local Polynomial Fitting

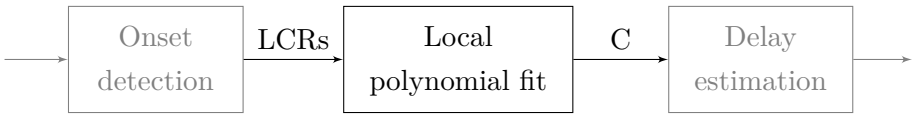


Figure 4.1: Local polynomial fitting block, where C corresponds to the set of coefficients calculated for each sample.

From the previous step, we obtain two LCRs: one computed from the audio signal arriving from the microphone on the left and one from the right. These LCRs serve the purpose of detecting onsets, as extensively discussed in the previous chapter. The idea, now, is to try to estimate the time delay between these two LCRs. To do so, we introduce an intermediate step, as underscored in Figure 4.1. We locally approximate the LCRs by polynomials in order to make these signals more tractable for the delay estimation.

4.1 Polynomial LSSM

As presented in the Subsection 2.1.2, polynomials can be easily represented by a LSSM. In this work, we approximate the LCR locally with a 3^{rd} degree polynomial with LSSM defined by

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.1}$$

$$s = \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}^\intercal. \tag{4.2}$$

The output vector C encodes the polynomial coefficients.

Here, we open a parenthesis to talk about the output vector. From the formulation of the trajectory defined by an LSSM in Equation (2.3), it should be noted that the LSSM parametrization of a polynomial is done on the binomial basis $(\binom{k}{0}, \binom{k}{1}, \binom{k}{2}, \dots)$, as the binomial coefficients naturally appear when computing matrix powers. If we wish to use the canonical basis representation of a polynomial, namely $(1, k, k^2, \dots)$, we need to map the coefficients in C from the binomial basis to the canonical basis. An extensive explanation of how to perform such transformation and why this is the case can be found in [5]. For our practical purposes, since we are dealing with polynomials of 3$^{\text{rd}}$ degree, we use the transformation matrix $T \in \mathbb{R}^{4\times4}$ defined as

$$T = \begin{pmatrix} 0 & 1/3 & -1/2 & 1/6 \\ 0 & -1/2 & 1/2 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \tag{4.3}$$

and we transform the output vector in the binomial basis C_b to the output vector in the canonical basis with

$$C_c = C_b T \tag{4.4}$$

4.2 Locally Fitting Polynomials

Polynomials explode to either plus or minus infinity into the future and into the past, therefore, we only locally fit the polynomial, using a specific window. In this work, we used a rectangular window.

Since the rectangular window has a finite length, the cost per sample is computed based only on the samples that are inside the window at that given instant. Let $a < 0$ and $b > 0$ be the window limits for a rectangular window centered at sample k. Let $\text{LCR}_k \in \mathbb{R}^2$ be comprised of the LCR samples at instant k. We treat both LCRs at the same time, interpreting it as a multi-channel signal with two channels: one generated from the signal from the left and one from the right. We also consider the signal to be generated by a LSSM moving forward in time. The cost at sample k is, then, given by

$$J_k(C) = \sum_{i=a}^{b} \left\| \mathrm{LCR}_{i+k} - CA^i s \right\|_2^2. \tag{4.5}$$

Following a similar derivation as the one presented in Appendix A, the cost can be re-written as in (3.14), with $\vec{\kappa}_k$ and $\vec{\xi}_k$ defined by Equations (4.6) and (4.7), that are easily calculated recursively. The message $\vec{W}_k$, defined by (4.8) does not depend on k, so it can be pre-computed.

$$\begin{aligned} \vec{\kappa}_k &= \sum_{i=a}^{b} \|\mathrm{LCR}_{i+k}\|_2^2 \\ &= \vec{\kappa}_{k-1} - \|\mathrm{LCR}_{k+a-1}\|_2^2 + \|\mathrm{LCR}_{k+b}\|_2^2 \end{aligned} \tag{4.6}$$

$$\begin{aligned} \vec{\xi}_k &= \sum_{i=a}^{b} A^i s (\mathrm{LCR}_{i+k})^\intercal \\ &= A^{-1}(\vec{\xi}_{k-1} - A^a s (\mathrm{LCR}_{k+a-1})^\intercal + A^{b+1} s (\mathrm{LCR}_{k+b})^\intercal) \end{aligned} \tag{4.7}$$

$$\vec{W}_k = \sum_{i=a}^{b} A^i s s^\intercal (A^i)^\intercal, \tag{4.8}$$

The derivation for the recursions presented in Equations (4.6) and (4.7) can be found in Appendix B.

The optimal value of C is, then, calculated as (3.18). The derivation of the recursions for the messages for this section can be found in Appendix B.

What we get from the fit is a set of coefficients C in the binomial basis. We, then, map it to the canonical basis using the transformation matrix (4.3). For every LCR sample k we get two sets of coefficients, which correspond to the coefficients of the 3^{rd} degree polynomials that locally approximate each LCR within the rectangular window. The obtained coefficients are the quantities used to perform the delay estimation, as discussed in the next chapter.

Chapter 5

Delay Estimation

Figure 5.1: Delay estimation block, where $\delta[\cdot]$ corresponds to the delay estimate for each sample.

The delay estimation is performed with the local polynomial fits obtained from the previous step, as it can be observed in Figure 5.1. In this chapter, we present the algorithm used to estimate the delay between the two audio signals from the LCR local polynomial approximations.

5.1 Algorithm

We do not perform a continuous delay estimation. As mentioned earlier, the desired LCRs are sparse and an accurate delay estimate can be obtained only in certain specific instants. One could perform the delay estimation by observing the adjacent peaks of the two LCR signals, for example, but this is not the approach followed here. We choose to estimate the delay in the moments in which the LCRs are rising.

The first step in the delay estimation is, then, detecting the instants in which both LCRs are rising. This can be easily accomplished by observing the coefficients of the local polynomial fits obtained in the previous part. A rising polynomial is characterized by a large first derivative (since the signal level is rapidly increasing) and a small second derivative (since we expect

the rise to be almost linear). For each time step k of the LCRs, we get the polynomials $p_k^{(L)}(\ell)$ and $p_k^{(R)}(\ell)$ centered in k which approximate the LCR from the left and from the right, respectively, in the form of (5.1) and (5.2).

$$p_k^{(L)}(\ell) = a_{k0}^{(L)} + a_{k1}^{(L)}(\ell - k) + a_{k2}^{(L)}(\ell - k)^2 + a_{k3}^{(L)}(\ell - k)^3 \tag{5.1}$$

$$p_k^{(R)}(\ell) = a_{k0}^{(R)} + a_{k1}^{(R)}(\ell - k) + a_{k2}^{(R)}(\ell - k)^2 + a_{k3}^{(R)}(\ell - k)^3 \tag{5.2}$$

We set hard thresholds for the values of the first and second derivatives at time k. Thus, we continuously check if the first derivative at the current sample is above the first threshold, i.e., if $a_{k1}^{(L,R)} > \tau_1$ and if the second derivative is below the second threshold, i. e., if $2a_{k2}^{(L,R)} < \tau_2$. If both conditions are satisfied for both the polynomial fit from the left LCR as well as the polynomial fit from the right LCR, we proceed with the delay estimation.

We, then, approximate the 3rd degree polynomials at that instant, defined by Equations (5.1) and (5.2), by 2nd degree polynomials $p_2^{(L)}(\ell)$ and $p_2^{(R)}(\ell)$, in order to simplify the problem of delay estimation even further. The 2nd degree polynomials $p_2^{(L)}(\ell)$ and $p_2^{(R)}(\ell)$ have coefficients $b_{k0}^{(L)}, b_{k1}^{(L)}, b_{k2}^{(L)}$ and $b_{k0}^{(R)}, b_{k1}^{(R)}, b_{k2}^{(R)}$, respectivelly. This is done analytically, by choosing three points in each 3rd degree polynomial LCR and solving the resulting linear system of equations to determine the 2nd degree polynomial parameters, as shown in (5.3), where the three points chosen were $k + c, k$ and $k + d$.

$$\begin{pmatrix} b_{k0}^{(L,R)} \\ b_{k1}^{(L,R)} \\ b_{k2}^{(L,R)} \end{pmatrix} = \begin{pmatrix} 1 & c & c^2 \\ 1 & 0 & 0 \\ 1 & d & d^2 \end{pmatrix}^{-1} \begin{pmatrix} p_k^{(L,R)}(k+c) \\ p_k^{(L,R)}(k) \\ p_k^{(L,R)}(k+d) \end{pmatrix} \tag{5.3}$$

With the 2nd degree polynomial coefficients in hand, estimating the delay is straightforward. First, we check which of the polynomials arrived first by looking at their value at time k, i.e., $p_2^{(L)}(k)$ and $p_2^{(R)}(k)$ and verifying whether $p_2^{(L)}(k) > p_2^{(R)}(k)$ or $p_2^{(L)}(k) < p_2^{(R)}(k)$

Then, we solve the equation that gives us when the polynomial that arrived later reaches the same level as the one that arrived first. This equation has at most one solution that makes sense and this is the delay estimate.

The pseudo-code for the algorithm can be observed in Algorithm 1.

for *every sample* k **do**
  if $a_{k1}^{(L,R)} > \tau_1$ *and* $2a_{k2}^{(L,R)} < \tau_2$ **then**
    approximate each 3rd degree polynomial by a 2nd degree polynomial, $p_2^{(L,R)}(\ell)$;
    check which 2^{nd} degree polynomial is in front and name it $p_2^{(front)}(\ell)$. The other is named $p_2^{(back)}(\ell)$;
    solve $p_2^{(back)}(\ell) = p_2^{(front)}(\ell)$, resulting in roots ℓ_1 and ℓ_2 ;
    if $\ell_1 < 0$ *and* $\ell_2 > 0$ **then**
      $delay[k] = \ell_2$;
    end
    if $\ell_1 > 0$ *and* $\ell_2 < 0$ **then**
      $delay[k] = \ell_1$;
    end
    if $\ell_1 > 0$ *and* $\ell_2 > 0$ **then**
      $delay[k] = \min(\ell_1, \ell_2)$;
    end
  else
    $delay[k] = delay[k-1]$;
  end
end

Algorithm 1: Delay estimation algorithm

Chapter 6

Experimental Results

6.1 Dataset

To assess the proposed algorithm's working principles, we conduct a series of experiments in a dataset with speech signals, where the sound source is located at different azimuths in the horizontal plane. The speech signal is comprised of a male voice recording, with approximately 12 seconds duration. The speech signal is, then, convolved with the room impulse responses to obtain the signals with the sources at 36 different azimuths, namely from $-180°$ to $170°$ in steps of $10°$. Finally, to map from the estimated ITD to the azimuth, we have considered only the audio signals for which the source was in the front part of the horizontal plane, i.e., within the azimuths of $-90°$ and $90°$. We obtain the results by using only the signals for the source truly located within $-90°$ and $90°$ azimuth as well as mapping the back azimuths to the front.

A similar, but briefer, analysis was performed on a speech signal of a female voice recording. These results can be found in Appendix C.

6.2 Onset detection

In order to robustly detect the sound onsets in a way that facilitates the next steps of processing, we experiment with the different onset models and windows presented in Chapter 3. More specifically, we obtained the LCRs for both decaying sinusoids and gammatone filters with gamma and exponential windows. We perform a grid search, sweeping over the onset and window models' hyperparameters with the objective of tuning them for the task at

hand.

Intuitively, we can see the grid search process as trying to find the perfect balance between the localization provided by the window and the similarity between the onset model and the actual sound onsets. Since the LCR measures how good the fit was to each part of the audio signal, we want the fit to be particularly good when a sound onset is occurring and, thus, we need to find the onset hyperparameters, namely frequency and decay, that mimic well a real sound onset.

We start by thinking about the onset models' hyperparameters and then we discuss the window models' hyperparameters. The onset models we consider have two hyperparameters: the frequency and the decay. These hyperparameters must be tuned with the objective of making the onset model look as similar as possible to an actual sound onset. The onset model's frequency has a direct implication in the resulting LCR. If the onset model's frequency matches the frequency of the sound onset, we have a good fit. The human voice contains base frequencies within the range of 80 to 255 Hz [6], so if the sound onsets to be detected come from speech samples, choosing a frequency within that range is a good idea. One might also consider using frequency banks or sums of onsets with different frequencies. In both these cases, as long as a frequency similar to the one of the sound onset, the resulting LCR should be satisfactory with respect to the frequency parameter. The decay parameter for both onset models can be seen as, in a sense, controlling the onset model's complexity. A low value of the decay – which corresponds to an abrupt decay in time – produces a simpler onset model, while a high value of the decay – which corresponds to a slower decay in time – produces a higher model complexity.

The windows considered – gamma window and exponential window – both have a single hyperparameter: the decay. The decay parameter of the window can be thought of in terms of controlling the localization of the fit. The lower the value of the decay parameter for the window model, the more localized is the fit, while the higher the value of the decay, the more global is the fit.

The process of hyperparameter tuning can be thought of as striving to find a balance between the trade-offs of onset model complexity and localization. Graphically, we can think of it as illustrated in Figure 6.1.

In Region I, since the window decay parameter is low, the fit is too local.

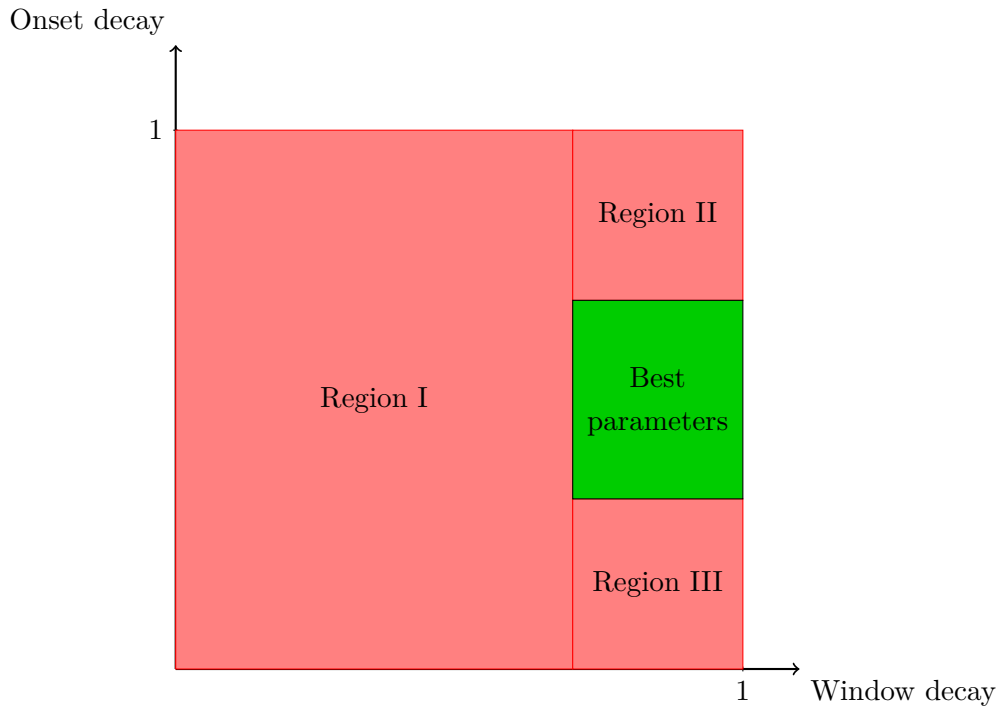


Figure 6.1: Parameter search regions, not drawn to scale.

When this is the case, there are numerous instants when the fit is considered good, since we are looking at a small number of samples. The resulting LCR, then, contains many spurious peaks.

In Region II, the window decay parameter is acceptable, but the onset decay is too high. This implies that the onset model is too complex and we might not find any good fit and the resulting LCR signal is weak, with all values close to zero.

In Region III, the onset decay parameter is too low, so the onset model is too simple. Therefore, there are many moments in which a good fit is achieved and the LCR contains many spurious peaks.

Finally, for parameters inside the region depicted in green, a good balance of localization and model complexity is achieved and the LCR is satisfactory.

To determine if the LCR resulted from one fit is better or worse than the one produced by another, we rely on the heuristic metrics also presented in Chapter 3. Empirically, we observed that these metrics produce similar results.

The audio signal from which we generate the LCRs is from a male voice.

We chose to fix the frequency to 80 Hz, since it seemed to match the frequency in the recording. We have experimented with all the different combinations of onset models and window models and, consistently, the best results produced were with the gamma window.

The decaying sinusoid and the gammatone filter onset models produce comparable results. This can be observed graphically in Figure 6.2. The LCR entropy was the measure used to guide the grid search and the difference between the calculated entropy for both onset models in their best cases was negligible. The hyperparameters that produce such results, though, were different. A summary of the best hyperparameters found for each onset model with the gamma window can be observed in Table 6.1, where the decay in time is defined as the time interval between the peak of the signal and the instant when it reaches 50% of its maximum amplitude.

Onset	**Onset decay**(value/in time)	**Window decay**(value/in time)
Decaying sinusoid	0.99/1.56 ms	0.999/56.32 ms
Gammatone filter	0.987/4.32 ms	0.999/56.32 ms

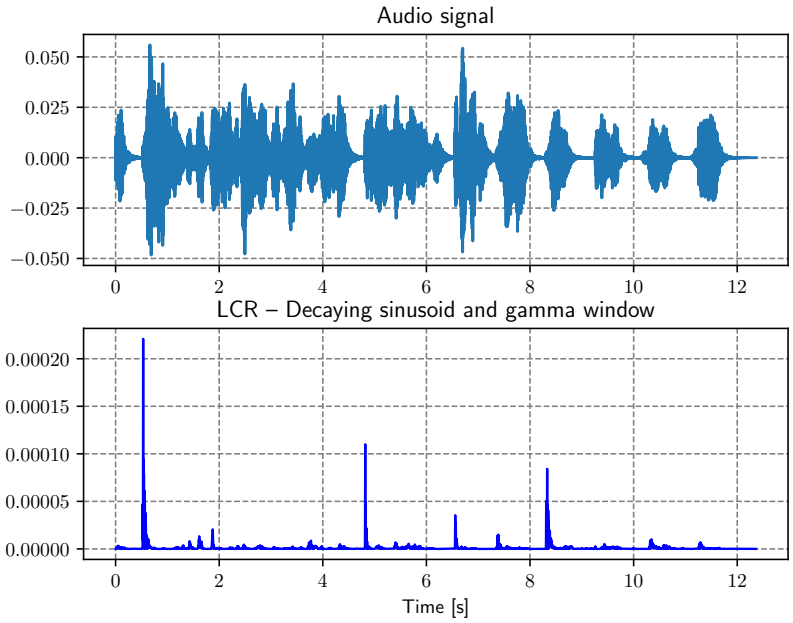
Table 6.1: Best parameters for each onset model.

6.3 Local polynomial fitting and delay estimation

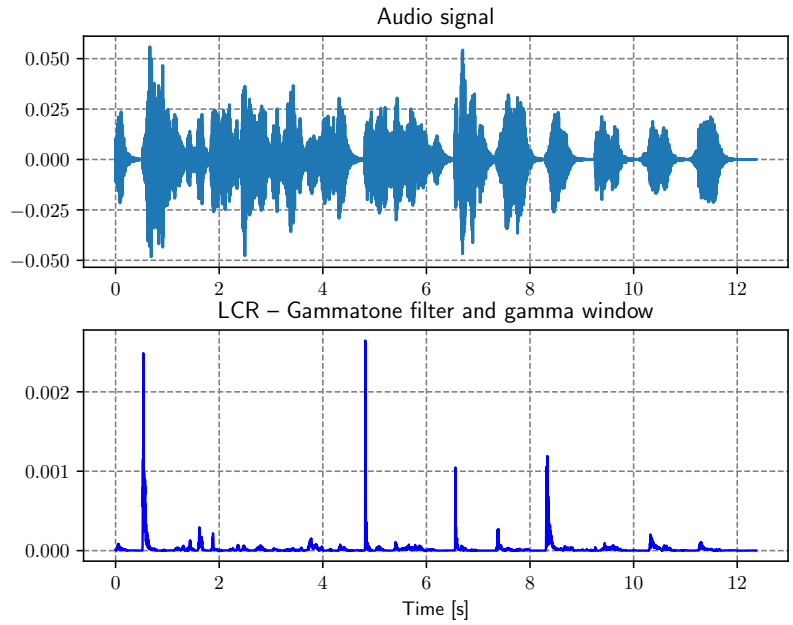
Once the best hyperparameters for onset detection are found, which produce sparse LCRs, the next step in processing is the local polynomial fit, decribed in Chapter 4, followed by the delay estimation, described in Chapter 5.

Table 6.2 displays all the hyperparameters used for the local polynomial fit and for the delay estimation. For both polynomial fits, a rectangular window is used. For the 3^{rd} degree polynomial fit described in Chapter 4, we used the rectangular window with parameters $a = -100$ and $b = 100$, resulting in a window length of 201 samples. For the 2^{nd} degree polynomial fit described in Chapter 5, we have used a rectangular window of length 81 samples, which correspond to choosing values of $c = -40$ and $d = 40$. It is also important to take into account that the audio signal for which we perform the delay estimates has a sampling frequency equal to 44.1 kHz.

The sequence of plots displayed in Figure 6.3 illustrates the steps taken by the proposed algorithm to perform the delay estimation. First, as illustrated in Figure 6.3a, a moment in which both the LCRs are starting to rise is



(a) LCR for a decaying sinusoid onset model with a gamma window.



(b) LCR for a gammatone filter onset model with a gamma window.

Figure 6.2: LCRs for each onset model.

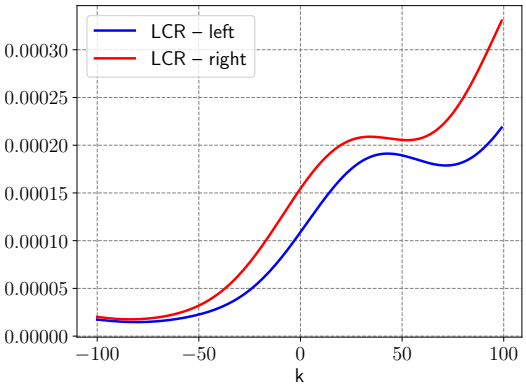
Parameter	**Value**
Window length for the 3^{rd} degree polynomial fit	201 samples
Window length for the 2^{nd} degree polynomial fit	81 samples
Threshold for the first derivative (τ_1)	$1.2\ 10^{-6}$
Threshold for the second derivative (τ_2)	10^{-13}

Table 6.2: Local polynomial fit and delay estimation algorithm parameters.

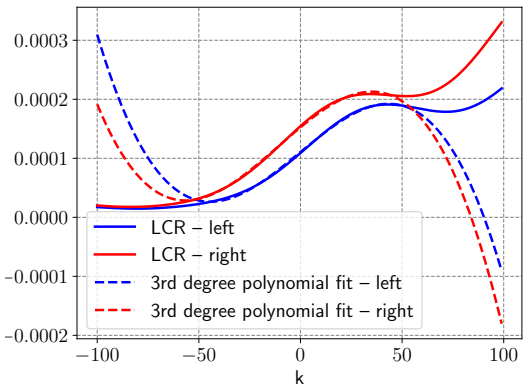
detected, meaning that a sound onset was detected. As shown in Algorithm 1, this is the condition to proceed with the delay estimation. Then, as seen in Figure 6.3b, the LCR is locally approximated by a 3^{rd} degree polynomial within a rectangular window. Finally, the 3^{rd} degree polynomial is approximated by a 2^{nd} degree polynomial within a smaller rectangular window. The delay is then estimated based on the shift calculated between the two 2^{nd} degree polynomials. As it can be observed in Figure 6.3c, the delay estimate produced by the algorithm is equal to approximately 20 samples, which, considering that the signal was sampled at 44.1 kHz, corresponds to a delay close to 0.45 ms.

Figure 6.4 illustrates the successive delay estimates performed by the algorithm in different instants. In this case, the sound source was located at an azimuth of $-130°$. As it can be observed, the instants in which the algorithm re-estimates the delay coincides with the instants that the LCRs are rising. With the threshold parameters τ_1 and τ_2 chosen, the delay is only estimated for the two strongest rises in the LCR, so the last two peaks, observed at the top of Figure 6.4, do not fulfill the condition defined by the threshold parameters and a delay estimate is not produced. One could, of course, lower the threshold values, re-estimating the delay for every visible LCR peak, but we chose to be conservative and only perform the delay estimate when the LCR signal was strongest.

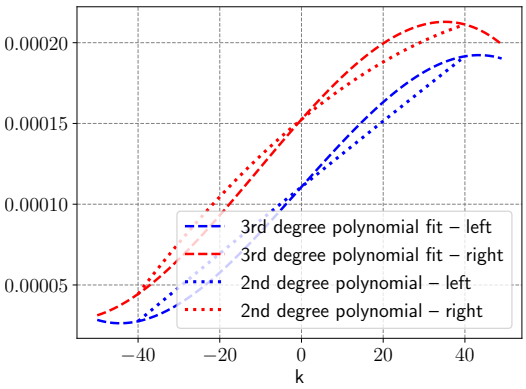
Finally, to learn the relationship between the delay estimate and the sound source localization, we run the proposed algorithm for the same audio signal, but for the source at different azimuths in the interval of $[-90°, 90°]$. Figure 6.5 depicts the trend between the azimuth and the delay estimate. A 3^{rd} degree polynomial is fit to this set of points to obtain the mapping. The mean squared error for the fit is equal to 0.40. Figure 6.6 illustrates the same relationship, but considering the delay estimates for the sound source at each



(a) LCRs from the left and from the right starting to rise as an onset is detected.



(b) Local 3$^{\text{rd}}$ degree polynomial fits to the LCRs.



(c) 2$^{\text{nd}}$ degree approximation for delay estimation.

Figure 6.3: Delay estimation algorithm steps.

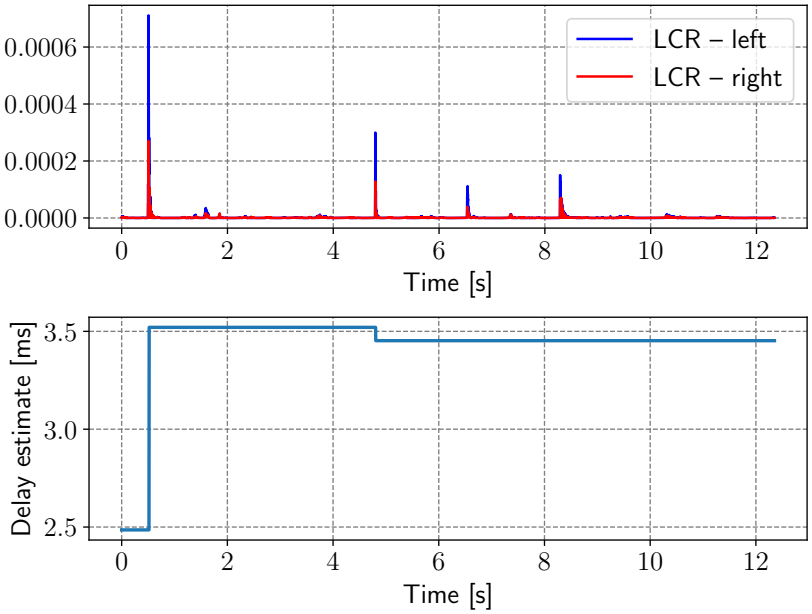


Figure 6.4: Delay estimates across the time.

azimuth within the $-180°$ to $170°$ range, by mapping the back azimuths to the front, resulting in a plot for the $-90°$ to $90°$ range. The mean squared error for this case is equal to 0.820.

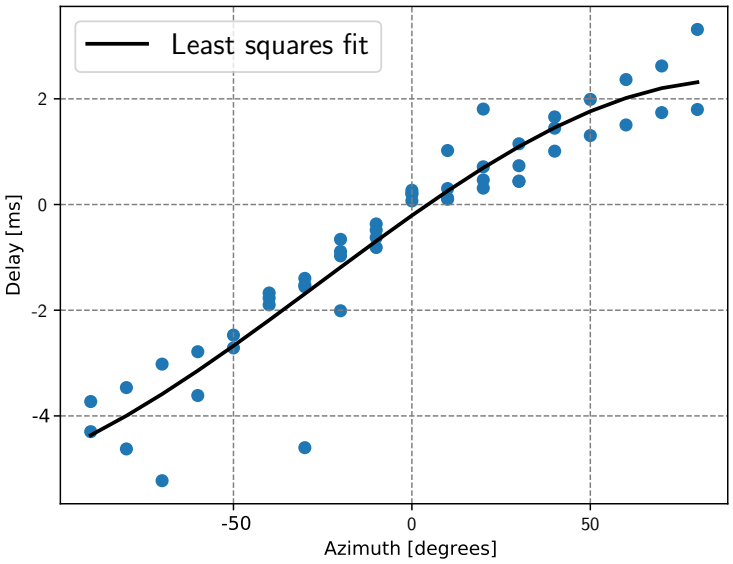


Figure 6.5: Delay estimation as a function of the source's azimuth only for the front part.

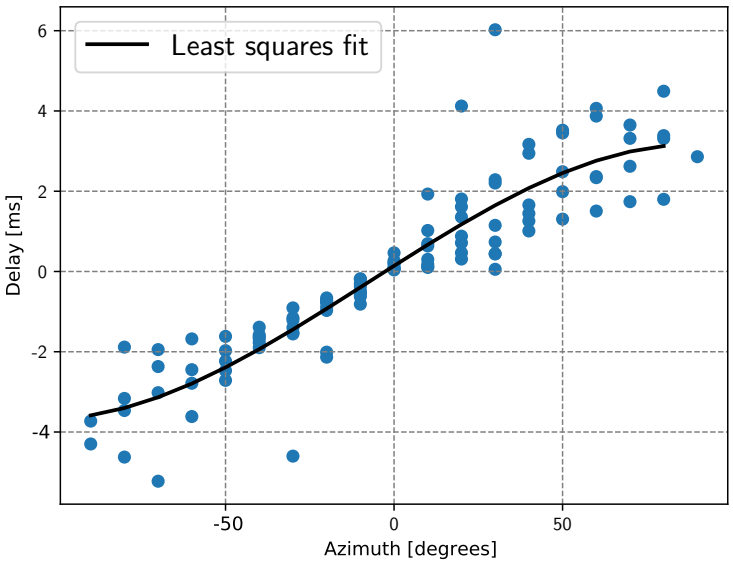


Figure 6.6: Delay estimation as a function of the source's azimuth, mapping the back azimuths to the front.

Chapter 7

Conclusion

In this work, we explored the problem of sound source localization via onset detection. In our exploration process, we extensively used LSSMs as a modeling framework, reinforcing their versatility for the numerous tasks that are ubiquitous in signal processing. The delay estimation algorithm studied here is completely based on LSSMs and provides the delay estimates in an almost online manner.

We have validated the algorithm for a few audio samples for illustration purposes and future work should aim to make the algorithm robust to a wider class of audio signals. This would correspond to exploring the use of filter banks with different onset model frequencies, to ensure a reliable onset detection, which is critical for the whole delay estimation pipeline.

Appendix A

Derivation of the Recursions for Onset Detection

In Chapter 3, it was claimed that there are computationally efficient recursive ways to calculate the cost and the optimal fit. Here, we show how to derive expressions (3.10) and (3.14).

To go from the basic cost formulation in (3.9) to (3.10), we start as follows:

$$\begin{aligned} J_k(C) &= \sum_{i=1}^{k} C_w A_w^{k-i} s_w \|y_i - CA^{k-i}s\|_2^2 \\ &= \sum_{i=1}^{k} C_w A_w^{k-i} s_w (y_i - CA^{k-i}s)^\intercal (y_i - CA^{k-i}s) \\ &= \sum_{i=1}^{k} C_w A_w^{k-i} s_w (y_i^\intercal - s^\intercal A^{k-i\intercal} C^\intercal)(y_i - CA^{k-i}s) \\ &= \sum_{i=1}^{k} C_w A_w^{k-i} s_w (y_i^\intercal y_i - s^\intercal A^{k-i\intercal} C^\intercal y_i - y_i^\intercal CA^{k-i}s \\ &\quad + s^\intercal A^{k-i\intercal} C^\intercal CA^{k-i}s). \end{aligned} \tag{A.1}$$

Notice that since $s \in \mathbb{R}^n$, $A \in \mathbb{R}^{n\times n}$, $C \in \mathbb{R}^{m\times n}$ and $y_i \in \mathbb{R}^m$, $s^\intercal A^{k-i\intercal} C^\intercal y_i \in \mathbb{R}$ and hence, $s^\intercal A^{k-i\intercal} C^\intercal y_i = y_i^\intercal CA^{k-i}s$. Thus, we continue with

$$\begin{aligned} J_k(C) &= \sum_{i=1}^{k} C_w A_w^{k-i} s_w (y_i^\intercal y_i - s^\intercal - 2y_i^\intercal C A^{k-i} s + s^\intercal A^{k-i\intercal} C^\intercal C A^{k-i} s) \\ &= \sum_{i=1}^{k} C_w A_w^{k-i} s_w \|y_i\|_2^2 - 2\sum_{i=1}^{k} C_w A_w^{k-i} s_w y_i^\intercal C A^{k-i} s \\ &\quad + \sum_{i=1}^{k} C_w A_w^{k-i} s_w s^\intercal A^{k-i\intercal} C^\intercal C A^{k-i} s. \end{aligned} \tag{A.2}$$

Using the fact that $\mathrm{tr}(ab^\intercal) = a^\intercal b = b^\intercal a$ for real vectors a and b with the same dimension, we get that

$$\begin{aligned} J_k(C) &= \sum_{i=1}^{k} C_w A_w^{k-i} s_w \|y_i\|_2^2 - 2\sum_{i=1}^{k} C_w A_w^{k-i} s_w \mathrm{tr}(C A^{k-i} s y_i^\intercal) \\ &\quad + \sum_{i=1}^{k} C_w A_w^{k-i} s_w \mathrm{tr}(C A^{k-i} s s^\intercal A^{k-i\intercal} C^\intercal). \end{aligned} \tag{A.3}$$

Now, using the fact that the window coefficient is a scalar and the linearity of the trace operator, we get

$$\begin{aligned} J_k(C) &= \sum_{i=1}^{k} C_w A_w^{k-i} s_w \|y_i\|_2^2 - 2\mathrm{tr}\left(\sum_{i=1}^{k} C_w A_w^{k-i} s_w C A^{k-i} s y_i^\intercal\right) \\ &\quad + \mathrm{tr}\left(\sum_{i=1}^{k} C_w A_w^{k-i} s_w C A^{k-i} s s^\intercal A^{k-i\intercal} C^\intercal\right) \\ &= \sum_{i=1}^{k} C_w A_w^{k-i} s_w \|y_i\|_2^2 - 2\mathrm{tr}\left(\sum_{i=1}^{k} C A^{k-i} s C_w A_w^{k-i} s_w y_i^\intercal\right) \\ &\quad + \mathrm{tr}\left(\sum_{i=1}^{k} C_w A_w^{k-i} s_w C A^{k-i} s s^\intercal A^{k-i\intercal} C^\intercal\right). \end{aligned} \tag{A.4}$$

It is possible to note that in the two last terms of (A.4), we have a product of two LSSMs. Therefore, we can use the property presented in subsection 2.3.2, which yields in

$$
\begin{aligned}
J_k(C) =& \sum_{i=1}^{k} C_w A_w^{k-i} s_w \|y_i\|_2^2 - 2\mathrm{tr}\left(\sum_{i=1}^{k} (C \otimes C_w)(A \otimes A_w)(s \otimes s_w) y_i^\intercal\right) \\
&+ \mathrm{tr}\left(\sum_{i=1}^{k} (C \otimes C_w)(A \otimes A_w)(s \otimes s_w) s^\intercal A^{k-i\intercal} C^\intercal\right) \\
=& C_w \sum_{i=1}^{k} A_w^{k-i} s_w \|y_i\|_2^2 - 2\mathrm{tr}\left((C \otimes C_w) \sum_{i=1}^{k} (A \otimes A_w)(s \otimes s_w) y_i^\intercal\right) \\
&+ \mathrm{tr}\left((C \otimes C_w) \sum_{i=1}^{k} (A \otimes A_w)(s \otimes s_w) s^\intercal A^{k-i\intercal} C^\intercal\right).
\end{aligned}
\tag{A.5}
$$

Defining $\vec{\chi}_k$, $\vec{\zeta}_k$ and $\vec{s}_k$ as in (3.11), (3.12), (3.13), we get that

$$
J_k(C) = C_w \vec{\chi}_k - 2\mathrm{tr}((C \otimes C_w)\vec{\zeta}_k) + \mathrm{tr}((C \otimes C_w)\vec{s}_k C^\intercal), \tag{A.6}
$$

as in (3.10).
To obtain (3.14), it is possible to start from (A.4) and proceed as follows

$$
\begin{aligned}
J_k(C) =& \sum_{i=1}^{k} C_w A_w^{k-i} s_w \|y_i\|_2^2 - 2\mathrm{tr}\left(\sum_{i=1}^{k} C A^{k-i} s C_w A_w^{k-i} s_w y_i^\intercal\right) \\
&+ \mathrm{tr}\left(\sum_{i=1}^{k} C_w A_w^{k-i} s_w C A^{k-i} s s^\intercal A^{k-i\intercal} C^\intercal\right) \\
=& C_w \sum_{i=1}^{k} A_w^{k-i} s_w \|y_i\|_2^2 - 2\mathrm{tr}\left(C \sum_{i=1}^{k} A^{k-i} s C_w A_w^{k-i} s_w y_i^\intercal\right) \\
&+ \mathrm{tr}\left(\sum_{i=1}^{k} C A^{k-i} s C_w A_w^{k-i} s_w s^\intercal A^{k-i\intercal} C^\intercal\right) \\
=& C_w \sum_{i=1}^{k} A_w^{k-i} s_w \|y_i\|_2^2 - 2\mathrm{tr}\left(C \sum_{i=1}^{k} A^{k-i} s C_w A_w^{k-i} s_w y_i^\intercal\right) \\
&+ \mathrm{tr}\left(C \left(\sum_{i=1}^{k} A^{k-i} s C_w A_w^{k-i} s_w s^\intercal A^{k-i\intercal}\right) C^\intercal\right).
\end{aligned}
\tag{A.7}
$$

Let

$$
\vec{\kappa}_k = C_w \sum_{i=1}^{k} A_w^{k-i} s_w, \tag{A.8}
$$

$$\vec{\xi}_k = \sum_{i=1}^{k} A^{k-i} s C_w A_w^{k-i} s_w y_i^\intercal, \tag{A.9}$$

and

$$\vec{W}_k = \sum_{i=1}^{k} A^{k-i} s C_w A_w^{k-i} s_w s^\intercal A^{k-i\intercal}. \tag{A.10}$$

Then, the cost can be rewritten as

$$J_k(C) = \vec{\kappa}_k - 2\mathrm{tr}(C\vec{\xi}_k) + \mathrm{tr}(C\vec{W}_k C^\intercal), \tag{A.11}$$

which is equal to (3.14).

The expressions (3.16) and (3.17) can be obtained by considering the decomposition of a $n \times m$ matrix X as in (A.12), where $P_{i,j}^{(m,n)}$ is a $n \times m$ matrix with a 1 at index (i, j) and zero everywhere else.

$$X = \sum_{i,j} \{X\}_{i,j} P_{i,j}^{(m,n)} \tag{A.12}$$

Appendix B

Derivation of the Recursions for the Local Polynomial Fit

In Chapter 4, the recursive messages for the cost computation for the local polynomial fit were presented. Here, we show how to derive such recursions, namely Equations (4.6) and (4.7).

Following a similar procedure as the one outined in Appendix A, one can arrive at the cost formulation as in

$$J_k(C) = \vec{\kappa}_k - 2\mathrm{tr}(C\vec{\xi}_k) + \mathrm{tr}(C\vec{W}_k C^\intercal), \tag{B.1}$$

where

$$\vec{\kappa}_k = \sum_{i=a}^{b} \|\mathrm{LCR}_{i+k}\|_2^2 \tag{B.2}$$

$$\vec{\xi}_k = \sum_{i=a}^{b} A^i s(\mathrm{LCR}_{i+k})^\intercal \tag{B.3}$$

$$\vec{W}_k = \sum_{i=a}^{b} A^i s s^\intercal (A^i)^\intercal. \tag{B.4}$$

It is possible to note that $\vec{W}_k$ does not depend on k, so it can be pre-computed and used for all of the cost computation steps. The recursive

computation of $\vec{\kappa}_k$ is straightforward and follows directly from

$$\begin{aligned}\vec{\kappa}_k =& \sum_{i=a}^{b} \|\mathrm{LCR}_{i+k}\|_2^2 \\ =& \sum_{i=a-1}^{b-1} \|\mathrm{LCR}_{i+k}\|_2^2 - \|\mathrm{LCR}_{k+a-1}\|_2^2 + \|\mathrm{LCR}_{k+b}\|_2^2 \\ =& \vec{\kappa}_{k-1} - \|\mathrm{LCR}_{k+a-1}\|_2^2 + \|\mathrm{LCR}_{k+b}\|_2^2.\end{aligned} \tag{B.5}$$

The recursive computation of $\vec{\xi}_k$ is obtained by first making the substitution $j = i + 1$ on (B.3).

$$\begin{aligned}\vec{\xi}_k =& \sum_{j=a+1}^{b+1} A^{j-1} s(\mathrm{LCR}_{j-1+k})^\intercal \\ =& A^{-1} \sum_{j=a+1}^{b+1} A^{j} s(\mathrm{LCR}_{j-1+k})^\intercal \\ =& A^{-1} \left(\sum_{j=a+1}^{b} A^{j-1} s(\mathrm{LCR}_{j-1+k})^\intercal + A^{b+1} s(\mathrm{LCR}_{b+k})^\intercal \right) \\ =& A^{-1} \Big(\vec{\xi}_{k-1} + A^{b+1} s(\mathrm{LCR}_{b+k})^\intercal - A^{a} s(\mathrm{LCR}_{a+k-1}) \Big),\end{aligned} \tag{B.6}$$

which correspond exactly to the recursions shown on Equations (4.6) and (4.7).

Appendix C

Experimental Results for a Female Speech Signal

The experimental results presented in Chapter 6 were obtained for a speech signal of a male voice recording. In order to validate the proposed to a wider class of audio signals, we also perform some experiments to a speech signal of a female speaker. The results presented in this Appendix are obtained for a female speaker which reproduces the same phrase as the male speaker from the signal used in Chapter 6.

The first stage in the proposed algorithm is the onset detection. As previously discussed, in order to obtain sparse LCRs, the onset model should look as similar as possible to an actual sound onset. The parameter that should surely change is the onset model frequency. The male voice recording has a lower base frequency than the female voice recording. Therefore, the onset model frequency should be higher for the female speech signal. This is indeed the case. While for the results presented in Chapter 6 we used an onset model frequency of 80 Hz, the best results found for the female audio signal correspond to an onset model frequency equal to 95 Hz. The onset frequency was found by visual inspection, by observing the onset model and the actual speech signal side to side. By changing only the frequency parameter and keeping all other onset and window parameters identical to those shown in Table 6.1, it is possible to obtain a LCR as depicted in Figure C.1.

From Figure C.1, it is possible to not that the LCR is indeed sparse, with only a few peaks when the sound onsets occur. If one compares Figure C.1

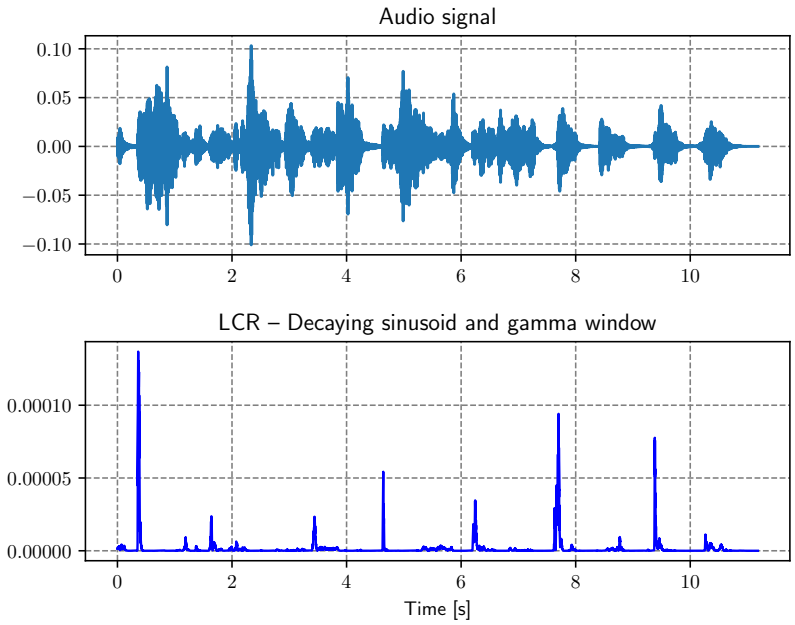


Figure C.1: LCR for a decaying sinusoid onset model with a gamma window for the female speech signal.

with Figure 6.2a, though, it is possible to note that the LCR for the female speech signal contains a few more prominent peaks. The presence of such prominent peaks might have to do with the pace with which the speaker speaks and can be indeed observed on the top part of Figure C.1.

The second part of the algorithm, i.e. the local 3^{rd} degree polynomial fit, remained exactly the same as the one presented in Chapter 6, as there is no evident reason to modify it's parameters for the given audio signal.

Finally, the third stage of the algorithm, namely the delay estimation, had to be slightly modified. As it can be noted by comparing Figures C.1 and 6.2a, the amplitude of the peaks for the LCR generated for the female speech signal is lower than the amplitude for the male speech signal. Therefore, to perform the delay estimation adequately, the threshold parameters had to be slightly adjusted.

Table C.1 summarizes the parameters used for the local polynomial fit and delay estimation for the female speech signal. If compared to Table 6.2, it is possible to note that the only line that differs is the threshold parameter for the first derivative, τ_1, which had to be decreased due to the above

Parameter	**Value**
Window length for the 3rd degree polynomial fit	201 samples
Window length for the 2nd degree polynomial fit	81 samples
Threshold for the first derivative (τ_1)	5. 10^{-7}
Threshold for the second derivative (τ_2)	10^{-13}

Table C.1: Local polynomial fit and delay estimation algorithm parameters for the female speech signal.

reasons.

In Figure C.2, it is possible to observe the delay estimates for the source at the different azimuths from $-90°$ to $90°$. When comparing to Figure 6.5, note the higher variance of the delay estimates near the azimuths of $-90°$ and $90°$. This higher variance is possibly related to the fact that by decreasing one of the threshold parameters for the delay estimation, we end up using less prominent peaks of the LCR or regions where the LCR are not rising as strongly to re-estimate the delay, leading to less accurate delay estimates. The trend in Figure C.2, though, is still clear, and it would be possible to map from the delay estimates to the sound source azimuth, shown by the black line, representing the 3^{rd} degree polynomial regression fit. The mean squared error for the fit shown in Figure C.2 is equal to 0.871, which is significantly higher than 0.40 found for Figure 6.5.

It is possible to observe in Figure C.3 that, indeed, the delay estimate is, in general, performed more times for the female speech signal due to the lower threshold. In this case, the delays correspond to a source at azimuth $20°$.

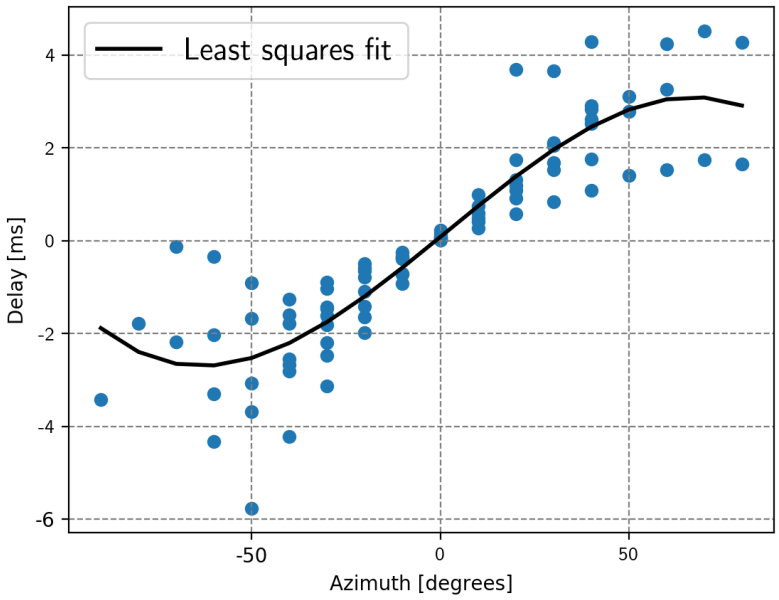


Figure C.2: Delay estimation as a function of the source's azimuth – only for the front part – for the female speech signal.

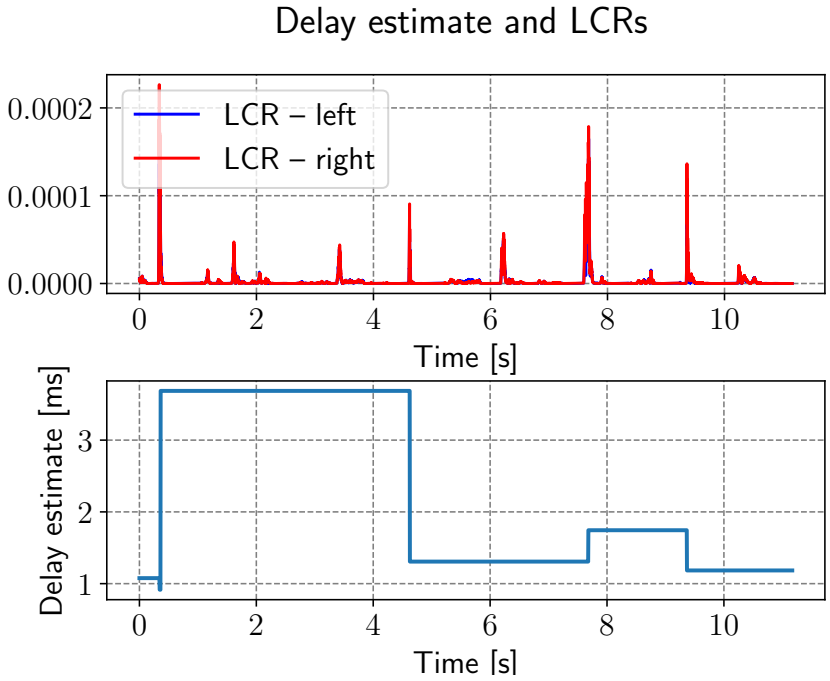


Figure C.3: Delay estimates across the time for the female speech signal.

Bibliography

[1] O. Nadiri and B. Rafaely, "Localization of multiple speakers under high reverberation using a spherical microphone array and the direct-path dominance test," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 10, pp. 1494–1505, Oct 2014.

[2] D. Wang and G. J. Brown, *Binaural Sound Localization*. IEEE, 2006, pp. 147–185. [Online]. Available: https://ieeexplore.ieee.org/document/5769587

[3] N. Ma, T. May, and G. J. Brown, "Exploiting deep neural networks and head movements for robust binaural localization of multiple sources in reverberant environments," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 12, pp. 2444–2453, Dec 2017.

[4] A. Brown, G. C. Stecker, and D. Tollin, "The precedence effect in sound localization," *Journal of the Association for Research in Otolaryngology : JARO*, vol. 16, 12 2014.

[5] N. Zalmai, "A state space world for detecting and estimating events and learning sparse signal decompositions," Ph.D. dissertation, ETH Zurich, Zürich, Switzerland, 2017. [Online]. Available: http://d-nb.info/1138847720

[6] I. R. Titze, *Principles of Voice Production*. Prentice Hall, 1994.

List of Figures

1.1 Proposed algorithm block diagram. 2

3.1 Onset detection block, where $y[\cdot]$ corresponds to the binaural audio signal. 8
3.2 Decaying sinusoid and gammatone filter signals 8

4.1 Local polynomial fitting block, where C corresponds to the set of coefficients calculated for each sample. 14

5.1 Delay estimation block, where $\delta[\cdot]$ corresponds to the delay estimate for each sample. 17

6.1 Parameter search regions, not drawn to scale. 22
6.2 LCRs for each onset model. 24
6.3 Delay estimation algorithm steps. 26
6.4 Delay estimates across the time. 27
6.5 Delay estimation as a function of the source's azimuth only for the front part. 28
6.6 Delay estimation as a function of the source's azimuth, mapping the back azimuths to the front. 28

C.1 LCR for a decaying sinusoid onset model with a gamma window for the female speech signal. 38
C.2 Delay estimation as a function of the source's azimuth – only for the front part – for the female speech signal. 40
C.3 Delay estimates across the time for the female speech signal. . 40

List of Tables

6.1 Best parameters for each onset model. 23
6.2 Local polynomial fit and delay estimation algorithm parameters. 25

C.1 Local polynomial fit and delay estimation algorithm parameters for the female speech signal. 39