Universiteit Antwerpen

Faculteit Wetenschappen

Departement Wiskunde-Informatica
Departement Natuurkunde

Automatic Estimation of Control Parameters for Musical Synthesis Algorithms

Automatische Schatting van Controleparameters voor Muzikale Synthesealgoritmes

Proefschrift voorgelegd tot het behalen van de graad van
Doctor in de Wetenschappen
aan de Universiteit Antwerpen te verdedigen
door

Wim D'haes

in het domein
Akoestiek, Signaalverwerking en Informatica Toegepast op Muziek

Promotor:
Prof. Dr. Dirk van Dyck, Visielab,
Universiteit Antwerpen
Antwerpen, België
Copromotor:
Dr. Xavier Rodet, Équipe Analyse/Synthese,
Institut de Recherche et Coordination Acoustique et Musique (IRCAM)
Parijs, Frankrijk

Automatic Estimation of Control Parameters for Musical Synthesis Algorithms

Wim D'haes

PhD Dissertation

may 2004

University of Antwerp

Promotor:
Prof. Dr. Dirk van Dyck,
Copromotor:
Dr. Xavier Rodet.

— To Ilona and my parents —

Thanks

- First of all I would like to thank my promotor Prof. Dr. Dirk "the Boss" van Dyck for giving me this great opportunity. His openness to new ideas, enthusiasm, optimism and "drive" were always motivating and inspiring. Thanks for supporting my ideas and providing me with new ones.
- To Dr. Xavier Rodet, for accepting me in the Analysis/Synthesis team of IRCAM and providing me with this challenging research topic. For helping me taking my first steps in this exciting research domain.
- To the Flemish Institute for the Promotion of Scientific and Technological Research in the Industry (IWT) for the financial support. Indeed the government should provide you with more money. Hope you will pay my postdoc too ...
- Everybody at the Vision Lab of the University of Antwerp.
- Everybody at the Analysis/Synthesis Team of IRCAM.
- Special thanks to Steve de Backer for the in depth discussions on various topics.
- To Ilona. For allowing me to work late.

Antwerp
May, 2004 *Wim D'haes*

Contents

List of Figures

Overview

This disseration addresses the problem of determining the control parameters of a musical synthesis algorithm in order to simulate a given signal. In the field of music synthesis, two paradigms are distinguished namely; *signal modelling synthesis* which is based on a mathematical model and *physical modelling synthesis* which is based on the simulation of the acoustical and mechanical properties of an acoustic instrument.

Signal modelling techniques describe a sound signal in terms of a mathematical model for which the parameters are computed by minimizing the difference between the signal and the model. A commonly used method method is sinusoidal modelling where the signal is described by a sum of sinusoids with time-varying amplitudes and frequencies. Typically, the residual is modelled by filtered white noise. Many techniques are known that allow to determine these parameters in an accurate way, resulting in a synthesis of a very high quality.

For physical models, the estimation of the control parameters is far more difficult since the control parameters are not related to the produced sound signal in a trivial way. The behavior of most musical instruments is highly nonlinear. This is for instance the case for the collision of the lips of a trumpet player, the bow that excites the string of a violin and the closure and opening of the vocal tract. In addition, a delayed feedback must be taken into account such as the pressure wave which is reflected at the end of a tube or transversal waves which are reflected at the end of a string. In this work, the objective is to use as little prior knowledge about the physical model as possible so that the estimation procedures can be applied easily to other models than the one which is studied throughout the thesis. The proposed methods originate from the pattern recognition field and are purely based on a data set containing input and output values.

The structure of the dissertation is depicted in Fig. 0.1. Each node represents a chapter and each arrow illustrates their dependence.

In chapter 1, a generic non parametric approach is proposed for the estimation of control parameters for musical synthesis algorithms. The method consists of modelling a multidimensional function which takes as input a vector of perceptually relevant signal features and returns the control parameters. This estimation is realized by K-nearest neighbor classification which retrieves the most similar vector from a data set. This data set is realized by synthesizing a large set of sounds and storing the signal features concatenated with the corresponding control parameters.

Since the computational complexity of a nearest neighbor classifier can prohibit its practical use when the data set is large, branch and bound search algorithms are developed in chapter 2. These algorithms decompose the data in a hierarchic manner. This decomposition is traversed in a depth first order while nodes that cannot contain nearest neighbors are avoided.

One shortcoming of the non parametric approach is that the physical constraints of the real instrument are not respected. For instance, for a sound with vibrato, control parameters are returned containing a periodically varying tube length. Chapter 3 discusses the physical model and its implementation, and derives a set of tube lengths that are adapted to a given tuning frequency.

In order to express the spectral similarity between two sounds, discrete Mel frequency cepstrum coefficients are used. However, when these coefficients are plot over time, it is shown that they are very sensitive to noise. One reason is that overfitting can occur since the envelope is only defined at discrete points and not in between these points. A second problem is that the high frequency band contains many low amplitude partials with a low signal to noise ratio. The noise of these partials is amplified enormously by the log function. These problems are addressed in chapter 4.

In chapter 5, the results of the previous chapters are applied in order to obtain a data set for which the physical constraints are respected and contain stabilized features. A new estimation method was developed which takes into account the coupling between the different control parameters.

The feature extraction that is used to estimate the parameters is based on sinusoidal modelling. However, the method which was initially used, estimates the complex amplitude and frequency in an iterative manner. This requires the use of rather large analysis windows since the frequency responses may not overlap. As a result, no fast variations in amplitude or frequency can be captured implying that the feature extraction fails during the transients. When smaller analysis window are used, the frequency responses of the sinusoidal components will overlap, requiring the computation of all amplitudes simultaneously. This method is not frequently used because of its high computational complexity. In chapter 6 and 7 it is shown that this can be improved considerably by including an analysis window with a bandlimited frequency response in the least square derivation.

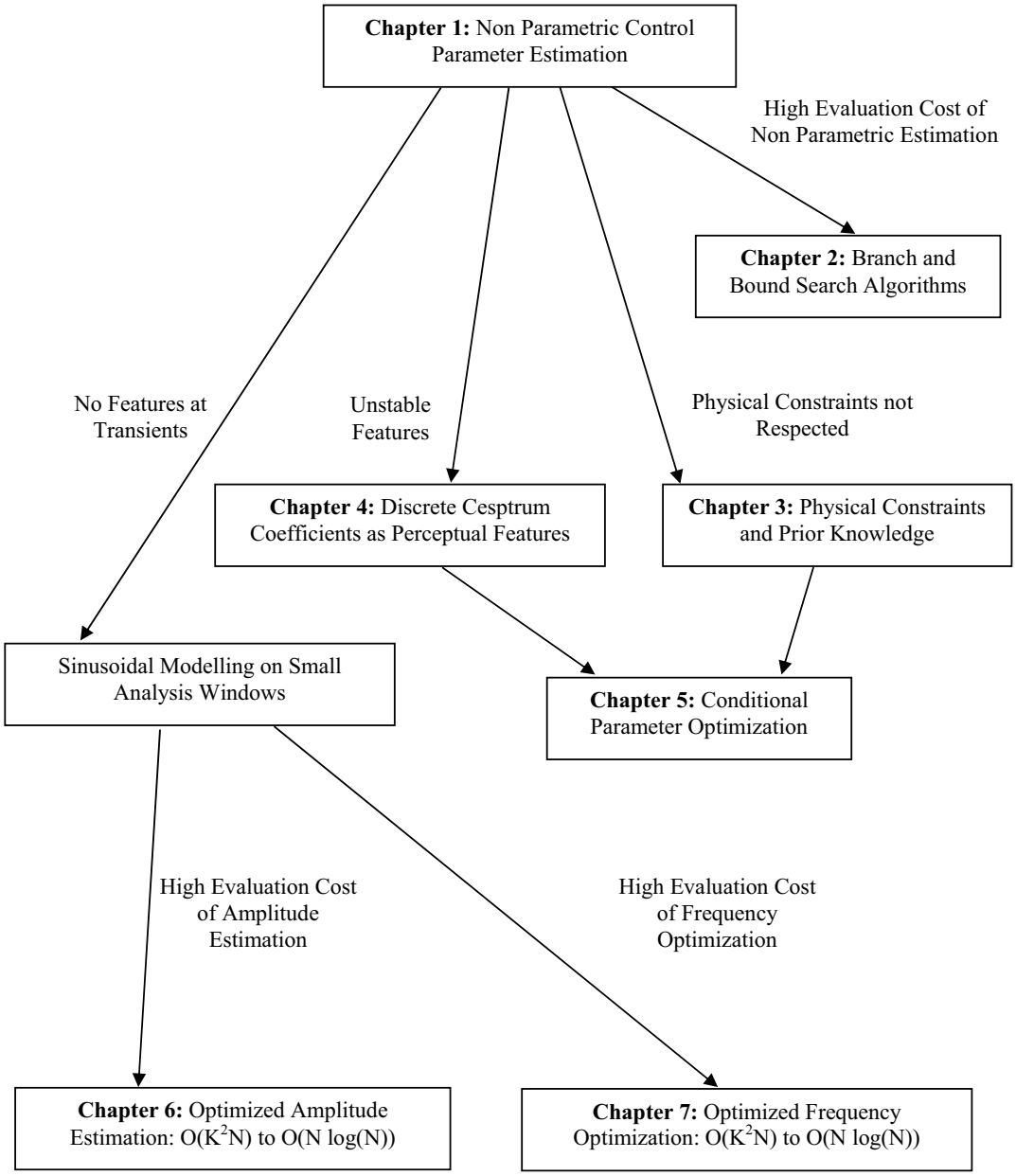


Fig. 0.1: *Overview of the Dissertation*

Part I

CONTROL PARAMETER ESTIMATION FOR PHYSICAL MODELS

CHAPTER 1

Non Parametric Control Parameter Estimation

1.1 Chapter Overview

A non parametric method is described for the estimation of the control parameters of a physical model of a trumpet. This method can be applied on a large class of systems for which a simulation, or "model" is available. We describe its application on a physical model of a trumpet where the "system" is an acoustical instrument, a trumpet, and the "model" a computer program that simulates trumpet sounds. A non parametric method was developed that determines the control parameters of the physical model from a set of features of the desired sound. This approach is particularly interesting since it is not strictly limited to this particular synthesis algorithm.

When describing the problem formally, the estimation problem can be considered as a multidimensional function that computes the control parameters from the extracted features (1.3). Since no analytic form of this function can be assumed, a non parametric estimation technique is used, based on K-nearest neighbor classification (1.4). The implementation (1.5) and results (1.6) of this method are discussed. Finally, some conclusions are given, including an overview of encountered problems that led to the work in subsequent chapters (1.7).

1.2 Introduction

When using a physical model to simulate an acoustic instrument, the way it is controlled is as important as the quality of the model itself. A physical model that is potentially capable of simulating any sound of an acoustic instrument will still sound very unnatural if it is not controlled correctly. Therefore, in order to produce realistic approximations of instrument tones, it is important to use appropriate time-varying control functions. A

real-time implementation of the model controlled by an adapted instrument-like interface may provide a preliminary solution, however it is not known how the interface parameters must be mapped to the control parameters of the synthesis model. In addition, it is impossible for a musician to control such an interface in order to obtain a professional musical performance. Therefore, techniques that can determine the control parameters in order to simulate a given sound are very interesting.

For synthesis techniques based on a signal model (for example sinusoidal modelling), efficient estimation techniques are available leading to sound manipulations of a very high quality [5, 71]. Some recent advances are presented in the last chapters and in [18, 17]. For physical models, techniques that determine automatically the control parameters for a given sound are actively researched. We cite work on plucked strings [60, 88, 89], bowed strings [79], the Sho[1][83] and the trumpet [38, 16, 23, 25].

Although one approach to finding control parameters would be to invert the mathematical equations on which the model is based [38], the non parametric approach considers the model as a "black box" neglecting the inner workings of the system. Only the output of the system is observed for different inputs. The method is described specifically for the parameter estimation of musical synthesis algorithms, but can be applied on any system-model couple for which the inputs and outputs are recordable and small in number. If the parameters of the model can be calculated so as to reproduce accurately a recorded trumpet performance of a professional musician playing a high quality instrument, very interesting applications arise such as coding and sound modifications.

Related research involving the control of synthesis algorithms, waveform synthesis in particular, has been done by [14]. Neural networks and memory based machine learning were used in the work of [100]. In these examples a signal model was controlled by the fundamental frequency and amplitude envelope functions. This work addresses the problem of the control of *physical models* using continuous time-varying parameters that have a physical meaning but are less directly related to characteristics of the produced sound. A similar technique is Code-Excited Linear Prediction (CELP) speech coding where the most appropriate innovation sequence is searched from a code book to optimize a given similarity criterion [76].

1.3 Formalization of the Problem

The problem of determining automatically the control parameters of a musical synthesis algorithm is formalized as follows. An acoustic system S is considered which is controlled by a set of parameters $\mathbf{p}(t)$ and produces a signal $s(t) = S(\mathbf{p}(t))$. This system is simulated by a model X with similar (but not necessarily exactly the same) control parameters $\mathbf{q}(t)$ which produces a signal $x(t) = X(\mathbf{q}(t))$. The main goal of the proposed method consists of determining an estimate of the control parameters $\tilde{\mathbf{q}}(t)$ so that it simulation by the synthesizer $\tilde{s}(t) = X(\tilde{\mathbf{q}}(t))$ is most similar to the original signal $s(t)$.

In order to determine the similarity between the original signal and its resynthesis, a distance measure must be developed. This distance measure is defined in terms of

[1] Chinese flute

perceptually relevant features $K(s(t))$ that are estimated from the signal $s(t)$ for regular time intervals. In the case of the trumpet, the sustained part of the sound is harmonic and can be described appropriately by its fundamental frequency and a characterization of its spectral envelope, such as linear prediction coefficients or cepstrum coefficients. Since the cepstrum coefficients do not adequately represent the spectrum of a pitched sound, the *discrete cepstrum* is preferred, which is calculated from peaks in the short time spectrum [31, 78]. The curve defined by the discrete cepstrum is divided in eight equal bins according to the Mel scale yielding a feature vector of eight elements that indicate the similarity between two spectra. Thus, the operator K denotes the feature extraction and computes the discrete cepstra $\bar{c}$ and the fundamental frequency ω from the signal

$$(\omega, \bar{c}) = K(x(t)) \tag{1.1}$$

The distance metric $d(.,.)$ between two sets of features computed from the short time signals $s(t)$ and $x(t)$ is defined as

$$\begin{aligned} (\omega_1, \bar{c}_1) &= K(x(t)) \\ (\omega_2, \bar{c}_2) &= K(s(t)) \\ d(K(x(t), K(s(t))) &\equiv \lambda(\log(\omega_1) - \log(\omega_2))^2 + (\bar{c}_1 - \bar{c}_2)^T(\bar{c}_1 - \bar{c}_2) \end{aligned} \tag{1.2}$$

Since the perceived pitch is approximately logarithmic in function of the fundamental frequency, the log difference is taken to express the tonal similarity of the sound. Also the perceived loudness is approximately logarithmic in function of the amplitudes which motivates the use of the Euclidean distance between the cepstrum coefficients. The parameter λ allows to control the relative importance of the tonal and spectral similarity manually. Since it is unacceptable that the model plays out of tune, this value is chosen rather high so that the tonal similarity will dominate. A disadvantage however is that this combined distance metric does not have a physical meaning.

Unfortunately, most methods that determine the fundamental frequency fail when transients (fast variation in amplitude or frequency) occur. In this case, the fundamental frequency and the spectral envelope are unreliable, resulting in a false characterization of the signal.

1.4 Modelling of Multidimensional Functions

After the feature extraction, the estimation of the control parameters can be seen as the modelling of a multidimensional function f that takes as input the signal features $K(s(t))$ and returns the control parameters

$$\bar{\mathbf{q}}(t) = f(K(s(t))) \tag{1.3}$$

The pattern recognition field offers powerful techniques to model this function from a *training set* of feature vectors containing the input and output values [4, 29]. In this case, the training is *supervised*, since both the input and the output of the system are

presented for the training. Unsupervised methods search for clusters in multidimensional data where each cluster is considered a separate class [2, 49].

In contrast to *classification* problems where one seeks to approximate the probabilities of membership to a number of classes, this is a *regression* problem since the values of $\mathbf{q}(t)$ are continuous. In general, three alternative approaches can be distinguished

- For the *parametric* approach, a specific analytical form of f is assumed containing a number of parameters which are optimized in order to fit the data set.
- By contrast, the second technique of *non-parametric* estimation does not assume any functional form and relies entirely on the data.
- The third approach, called *semi-parametric* estimation allows a very general class of functional forms, for instance neural networks, in which the number of adaptive parameters can be increased in a systematic way.

Since no mathematical form for f can be assumed, a non-parametric *k-nearest neighbors* estimation technique is used. This technique searches the most similar feature vector in the data set and returns the corresponding output. This technique is also known in the field of *machine learning* where it is called *instance-based learning* [56]. The learning procedure consists simply of storing new input-output pairs in a data set S which yields for the control parameter estimation problem

$$S = \{(\mathbf{q}(t_1), K(X(\mathbf{q}(t_1))), (\mathbf{q}(t_2), K(X(\mathbf{q}(t_2))), \ldots, (\mathbf{q}(t_N), K(X(\mathbf{q}(t_N))))\} \tag{1.4}$$

The evaluation of the function f consists simply of retrieving the feature vector $K(X(\mathbf{q}(t_i)))$ which is most similar to the features $K(s(t))$. The corresponding control parameters $\mathbf{q}(t_i)$ are then returned.

$$\tilde{\mathbf{q}}(t) = \min_{\mathbf{q}(t_i)} \; d(K(s(t)), K(X(\mathbf{q}(t_i)))) \tag{1.5}$$

Finally, these control parameters are used to for the synthesis of a simulation $\tilde{s}(t)$ of the signal $s(t)$

$$\tilde{s}(t) = X(\tilde{\mathbf{q}}(t)) \tag{1.6}$$

A disadvantage of the instance-based approach is that it requires a large number of distance computations in order to find the k-nearest neighbors. Therefore, *branch and bound search algorithms* are used that facilitate the rapid calculation of the k nearest neighbors [30, 61]. A branch and bound algorithm is a tree search algorithm that uses a *hierarchical decomposition* of the sample set of vectors. It does not compute a distance to all patterns in the data set but only to a certain node in the tree representing a subset of the sample. By using proper rules it is decided whether a vector in this node can be a nearest neighbor. If this is not the case, the complete node and all patterns belonging to it may be discarded.

For the experiments conducted in [16, 23, 20], the search algorithm described in [61] was applied. However, this search algorithm has several shortcomings. Namely, its decomposition method was not fully automated and could result in tree nodes that did not contain any vectors. Therefore, this method is not optimal. A fully automated and optimized method was developed, which is presented in chapter 2 [21, 19, 26].

1.5 Implementation

1.5.1 The Physical Model

The control parameter estimation was applied on a physical model of a trumpet [90, 93] developed at the Analysis/Synthesis team of IRCAM. Also a real-time implementation of this model was available [84].

A trumpet consists of a mouthpiece on which the player places his lips, followed by the instrument body which is basically a tube. The length of this tube can be modified by pressing the valves. The tube ends in the bell from which the outgoing pressure wave results in the perceived sound. In the physical model, the lips are modelled by a mass-spring-damper system to which a non-linearity is introduced when the lips collide. The tube is simulated by a transfer function measured from a real trumpet and a delay that depends on the tube length. Finally, the effect of the bell is modelled by a high pass filter.

The control parameters $\mathbf{q}(t)$ of the model are: the pressure in the mouth, the frequency of the lips, the damping factor of the lips and the tube length. The relationship between the tube length and the lip frequency is very important since it determines which mode of the tube is excited.

1.5.2 The Training Phase

Figure 1.1 shows an overview of the training and simulation phases of the system. The training consists of

- 1) Synthesis of the sounds that will be used in the data set.
- 2) Feature extraction from these sounds.
- 3) Concatenation of the control parameters and features. Storage of this vector in the data set.

The first data set for the inversion of the physical models was realized by controlling the real-time implementation of the model. During the real-time synthesis, the control parameters and synthesized sounds were recorded. All notes on the chromatic scale were played with a slow crescendo and diminuendo in order to have all possible intensities. Then a slow vibrato was added so that more variation in timbre could be achieved.

A second data set was obtained by *sampling* the control parameter space. Slow crescendos were obtained by augmentation the mouth pressure for every combination of lip frequency, lip damping and tube length.

1.5.3 The Simulation Phase

The simulation of a given sound $s(t)$ is achieved by the following steps:

- 4) The features $K(s(t))$ of the sound are calculated at a frame rate of 100 Hz.

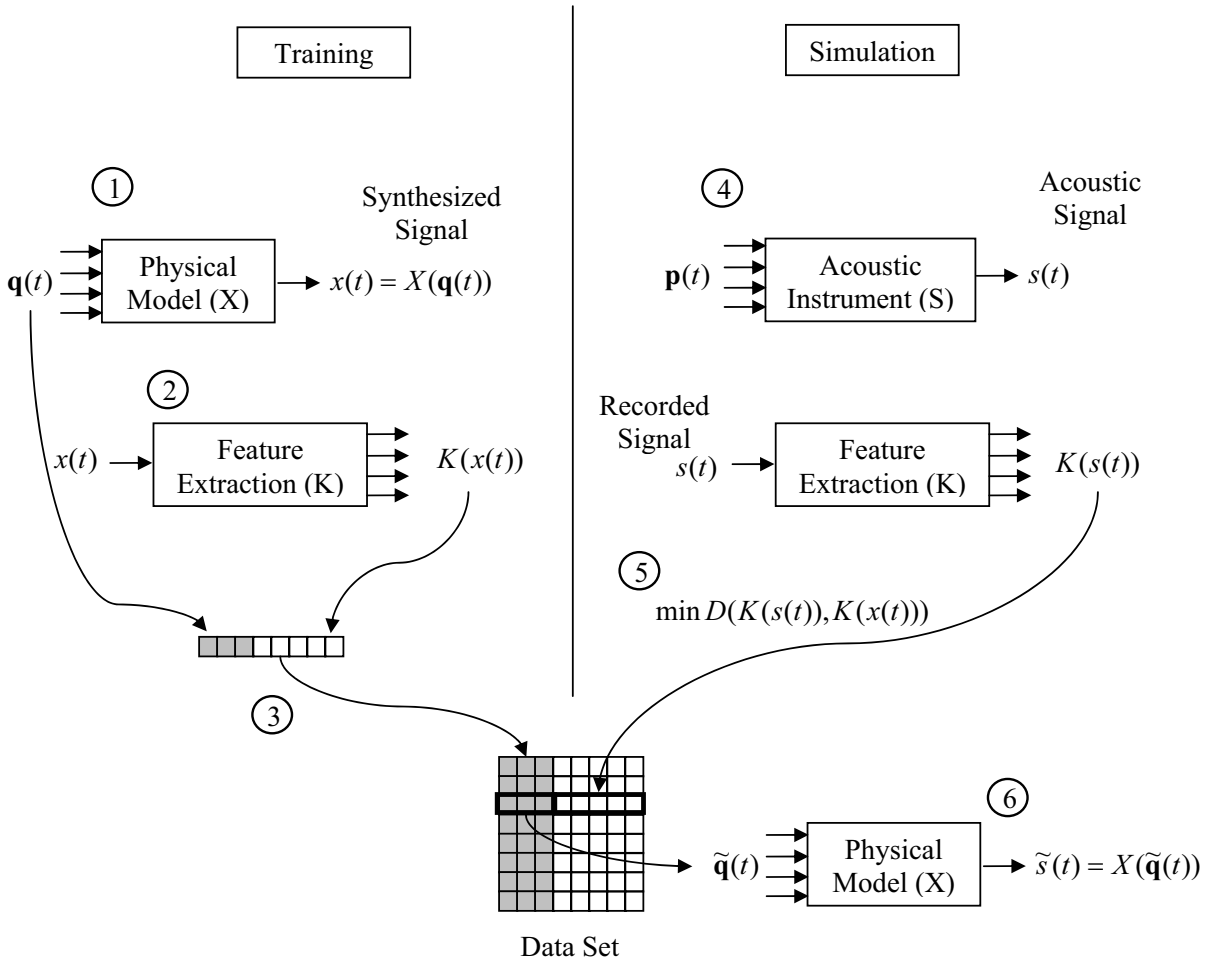


Fig. 1.1: *Non Parametric Estimation of Control Parameters.*

- 5) For each frame, the vector with the most similar characteristics $K(x(t))$ is searched in the database returning the corresponding control parameters $\mathbf{q}(t)$.
- 6) These parameters are written to file and used for the resynthesis of a sound $\tilde{s}(t)$ that is close to the original sound $s(t)$.

1.6 Results and Discussion

1.6.1 Simulations

For the first data set that was produced with the real-time implementation of the physical model, different sounds were simulated starting with sounds that were part of the training set. For these sounds the simulation was very accurate since the same exact feature vectors were available in the database. However, when attempting to simulate a sound that was not represented well in the data set several problems occurred due to sparse regions in the feature space. This sparsity can cause that sound characteristics that vary significantly around one isolated feature vector return always the same control parameters. In this case the resynthesized sound will remain stable while the original sound varies dynamically. By contrast, a small change of the signal characteristics may yield large variations in control parameters when the two closest feature vectors are very distant from each other.

Finally, recordings of an acoustic instrument were simulated. Since the training set contains only a chromatic scale of notes, and thus only a few discrete fundamental frequencies are well represented in the data set, a problem might occur when the recording of the real trumpet is not in tune with the scale that was played in the training set. In order to solve this problem a histogram for both the database and the sound file was made. The pitch feature of the original sound was adjusted slightly in order to guarantee that the feature vectors did not fall in a sparse region of the feature space. After some manual corrections of the control parameters at the transients a satisfactory simulation was obtained.

In order to avoid the sparse locations in the control parameter space, a second set was generated by a uniform sampling of this space. For the shortest tube length (no valves pressed) crescendos were generated for lip frequencies that excite the sixth mode of the tube. This was repeated for a number of values for the lip damping. A trumpet sound with a close fundamental frequency was simulated that contained dynamic variation in amplitude and a considerable vibrato. During the first simulation, it was observed that the variation in the fundamental frequency was obtained by changing the length of the tube. However, this would be impossible on a real instrument. Therefore, the length of the tube was determined first for each note. Then, when the other control parameters were determined, the only part of the data set considered was that produced by this fixed tube length.

Although the control parameter space can be sampled, this can not be realized for the feature space since the synthesizer can produce only a limited set of sounds. The dimensionality of the feature space is much larger than for control parameter space which

implies that the data lies in a lower dimensional subspace. This implies that when one wishes to simulate a sound which is not well represented in the data set, its feature vector will fall into a sparse region in the feature space. Therefore, the proposed non parametric approach is not robust and has weak generalization properties.

1.6.2 Transients

The approach that is described above assumes implicitly that the relationship between the control parameters and the sound characteristics is instantaneous or time-independent. This assumption is valid when the control parameters are varied slowly, but when the parameters are changed rapidly it takes several frames for the model to converge to a stable sound. As a result, very similar characteristics could yield very different control parameters. One solution might consist of considering consecutive frames as a single feature vector permitting to capture the dynamic evolution of the control parameters. This will result in enormous data sets since the quantity of training data needed to specify the mapping grows exponentially with the number of dimensions.

A second problem is that when transients occur, the characteristics that are estimated from the sound are very unreliable resulting in a false characterization of the signal. In this case, the control parameters cannot be retrieved by the proposed inversion method. However, experiments with the real-time implementation prove that the model is able to produce very natural transients. These transients can be detected by observing the the evolution of the fundamental frequency and the energy of the signal. Afterwards, the control parameters during the transients can be extrapolated from the context. In the case of the trumpet, two main types can be distinguished: a sudden augmentation or diminution of the pressure (onset or offset), and a sudden change in tube length and lip frequency (slur). It is assumed that during the onset, the same tube length and lip frequency is used as during the stable part of the note. During a slur, the lip frequency and tube length are changed instantaneously while the pressure is interpolated linearly. This results in quite natural sounding transients.

1.7 Conclusions

This chapter presents an initial attempt to estimate the control parameters of a physical model of a trumpet in an automatic manner. During this work, many problems were encountered which have led to the work presented in the subsequent chapters.

- **High Evaluation Cost:**

  It is commonly known that the main disadvantage of non parametric estimation techniques is their high evaluation cost. In its elementary form, the retrieval of K nearest neighbors requires the computation of the distance to all vectors that belong to the data set. When taking into account that the data sets contain about 40000 vectors and that the the features are computed at a rate of 100 Hz, this yields 4000000 distance computation to compute the parameters for each second of signal one wishes to simulate. Therefore, *branch and bound search algorithms*

were developed which allow to compute the K nearest neighbors in sublinear time. Initially the technique of Niemann and Goppert was implemented [61]. This algorithm however, did not provide an automatic decomposition, and could yield empty nodes in the search tree. Therefore, a new method based on principal component analysis was developed, which is fully automatic. This algorithm is presented in chapter 2 and is compared with the state of the art methods.

- **Respecting Physical constraints**

  When simulating a sound with vibrato, it was observed that the returned parameters attempted to simulate this by varying the tube length of the model. Evidently, this is not acceptable since this contradicts with how a real instrument is controlled. In addition, it is known that several notes can be obtained by different combinations of mode and tube length. For en entire trumpet performance, only seven tube lengths can be used. In chapter 3, the physical constraints of the physical model are discussed, and it is derived how the tube lengths can be adjusted to a given tuning frequency.

- **Unstable Features**

  When the discrete cepstrum coefficients were plot over time it was observed that these coefficients were very noisy although the sound was perceived as very stable. In chapter 4, two causes are described, being overfitting and the amplification of low amplitude variations by the log function. We propose a new method to compute the discrete Mel frequency cepstrum coefficients which was named posterior warping.

- **Conditional Estimation**

  As stated before, the dimensionality of the feature space is considerably larger then the dimensionality of the control parameter space. As a result, it will occur frequently that the feature vector computed from a sound one wishes to simulate lies in a sparse region of the data set. An alternative method consists of iteratively optimizing the control parameters with respect to the the similarity criterium.

  In chapter 5, a method is proposed which uses two distance metrics, being the tonal and spectral similarity. The parameters are determined in such a manner that the tonal similarity is optimal and that for this optimal tonal similarity, the optimal spectral similarity is obtained. This method was named *conditional optimization* since the second criterium is optimized under the condition that the optimal value for the other criterium is obtained.

- **Sinusoidal Modelling on Small Windows**

  The computation of the discrete cepstra that are used to define the spectral similarity relies on a sinusoidal modelling of the short time signal. The method which was initially used for the sinusoidal modelling computed the amplitude, phase and frequency of each sinusoidal component iteratively which requires the use of large analysis windows. In addition, this modelling is based on the assumption that

the amplitudes and frequencies are constant over the analysis frame which makes them inappropriate to track fast variations in frequency and amplitude as they occur during the transients.

Iterative optimization methods allow to use much smaller analysis windows but require a considerably higher computation cost namely $\mathcal{O}(K^2N)$ where K denotes the number of sinusoidal components and N the window length. The contribution made in chapter 6 consists of improving the computational efficiency of the amplitude estimation to $\mathcal{O}(N \log N)$. The same was realized for two frequency optimization methods which are discussed in chapter 7.

CHAPTER 2

Fast K-Nearest Neighbors Computation

2.1 Introduction and State of the Art

Searching the K nearest neighbors in a multidimensional vector space is a very common procedure in the field of pattern recognition where it is used for non parametric density estimation and classification [12, 29, 4]. However, when the number of samples is large, the computational cost of the nearest neighbor search can prohibit its practical use. Various techniques that improve the computational efficiency have been proposed and are discussed in this chapter.

2.1.1 The Basic Principle

All search algorithms require a preprocessing step in which the data set is hierarchically decomposed. This decomposition is represented by a tree where each node represents a subset of the data.

The nearest neighbor search traverses the tree in a depth first order. When the search algorithm encounters a leaf node, the distance to all vectors belonging to that node is computed and a new value for the distance to the Kth nearest neighbor is obtained. This distance is used to determine whether nodes that have not been evaluated can be excluded from the traversal. This is realized by defining a lower bound distance metric between the vector for which the nearest neighbors are searched and all vectors that belong to a given node. The exclusion of these nodes avoids distance computations for entire sets of vectors what implies a significant computational improvement.

The depth first tree traversal is implemented efficiently by using a stack on which all nodes that still need to be evaluated are stored. The nodes are traversed iteratively by popping the current node off the stack and pushing the child nodes on the stack. In Fig. 2.1, a snapshot of a depth first traversal is shown when the search has reached the

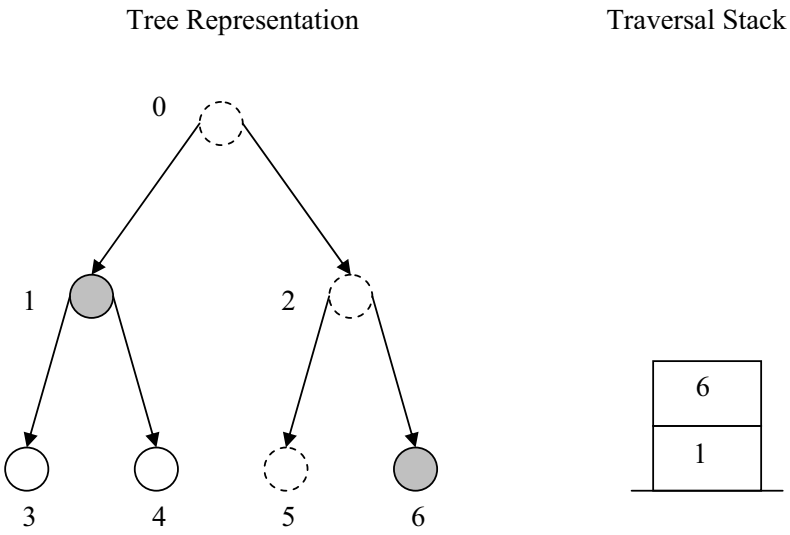


Fig. 2.1: *Snapshot of a Depth First Tree Traversal.*

first leaf node. The nodes that are represented by the dashed line have already been evaluated. The colored nodes are currently on the stack.

2.1.2 Literature

According to the type of space in which the data is represented, two groups of algorithms can be distinguished. A first group of algorithms can only be applied on data represented in a vector space and are called *branch and bound search algorithms*. The first branch and bound algorithm was developed by Fukunaga [30] and extended by Kamgar-Parsi [44]. The data set was decomposed hierarchically using a clustering algorithm. Niemann proposed to divide the sample space in hypercubes by segmenting the coordinates and adapted the distance measure between a node and a vector to this decomposition. However, the determination of these segment borders was still an open problem [61]. An automated segmentation method called *ordered partitioning* was presented in [47]. All these algorithms have a linear space complexity and a sublinear time complexity. Another interesting overview can be found in [42].

A second group of algorithms determine the nearest neighbors in a *metric space* [11]. The *approximating and eliminating search algorithm (AESA)* [94, 95, 43] has a quadratic space complexity, a linear time complexity and an average number of distance computations bounded by a constant term. Therefore, its application domain consists of problems for which the data cannot be represented in a vector space and the dissimilarity measure is computationally expensive. In [27], a probabilistic analysis is presented which

proves that the asymptotic average complexity measured by the number of dissimilarity calculations is constant. Also, a linear version of the algorithm (LAESA) was proposed that has a linear space complexity and keeps the other features unchanged [54, 55]. The TLAESA algorithm (a tree version of LAESA) applies the branch and bound algorithmic strategy on a metric space and has a sublinear time complexity at the expense of more distance computations [53]. Another sublinear AESA algorithm was developed in [96]. An overview of fast nearest neighbor search algorithms based on the approximation and elimination principle can be found in [69].

Recently, several branch and bound search algorithms were proposed that use a decomposition method based on Principal Component Analysis (PCA) [52, 21, 19, 26]. These algorithms search the nearest neighbors in a vector space where the dissimilarity between two vectors is expressed by the euclidian distance. Therefore, they belong to the first group of algorithms described previously. In the cited references, it was shown that these algorithms have a linear space complexity, an average number of distance computations bounded by a constant term and a time complexity that is very close to logarithmic for a small number of dimensions.

2.1.3 Chapter Overview

The efficiency of branch and bound search algorithms for the computation of K nearest neighbors is studied. The main aspects that influence its efficiency:

1. the decomposition method
2. the elimination rule
3. the traversal order
4. the level of decomposition

First, a *hierarchical decomposition* method is developed that separates the data set iteratively using hyperplanes (2.2). A *statistical model* of the computation cost (2.2.2), allows to define two efficiency criteria. It is shown that these criteria are optimized by choosing hyperplanes that are perpendicular to the maximal variance (2.2.3) while taking into account that each subnode should contains an equal number of vectors (2.2.4). A possible variation consists in using hyperplanes perpendicular to the axis (2.2.5).

For this decomposition to be used in the context of a branch and bound search algorithm, an appropriate *elimination rule* must be defined (2.3) which is then used to optimize the search algorithm (2.4). The number of distance computations (2.4.2) and total computation time (2.4.3) are studied in function of the number of dimensions, the number of nearest neighbors and the number of vectors.

Then, different possible traversal orders are described which can be optimized globally or locally (2.5). Also the *level of decomposition* is a user defined parameter which has a significant impact on the performance of the algorithm. A method is developed which allows to optimize the decomposition level from a few simple experiments.

The different decomposition methods, elimination rules and traversal orders yield ten different algorithms (2.7.1). The algorithms were compared for gaussian (2.7.3) and non gaussian (2.7.4) data distributions after determining the optimal decomposition level. Finally, the conclusions (2.8) and some applications are discussed (2.9).

2.2 Hierarchical Decomposition

2.2.1 Definition of the Decomposition

The decomposition of the sample is represented by a binary tree of which each node p represents a subset S_p of the total data set $S_0 = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_N\}$ with $\bar{x}_i \in \mathbb{R}^D$. Each branch denotes the decomposition of the sample according to a $D-1$ dimensional hyperplane in a D dimensional space. In general, a hyperplane can be determined by a point $\bar{\mu}_p$ contained within and a vector $\bar{v}_p$ perpendicular to it. It is then defined by the equation

$$\bar{v}_p^T(\bar{x} - \bar{\mu}_p) = 0 \tag{2.1}$$

where $\bar{x} \in \mathbb{R}^D$ and $\bar{v}_p^T$ denotes the transpose of $\bar{v}_p$. The values of $\bar{\mu}_p$ and $\bar{v}_p$ are determined in the following sections. When $\bar{v}_p$ is normalized, the absolute value of the expression $\bar{v}_p^T(\bar{x} - \bar{\mu}_p)$ yields the perpendicular distance between the vector $\bar{x}$ and the hyperplane. The sign of this expression indicates on which side of the plane the vector $\bar{x}$ lies, and therefore, to which child node it belongs. When the nodes are enumerated starting from zero, the root node S_0 represents the total data sample. The child nodes of a node p are defined by induction using

$$\begin{array}{ll} S_{2p+1} = & \{\bar{x}_i \in S_p : \bar{v}_p^T(\bar{x}_i - \bar{\mu}_p) < 0)\} \\ S_{2p+2} = & \{\bar{x}_i \in S_p : \bar{v}_p^T(\bar{x}_i - \bar{\mu}_p) \geq 0)\} \end{array} \tag{2.2}$$

implying $S_p = S_{2p+1} \cup S_{2p+2}$ and $S_{2p+1} \cap S_{2p+2} = \emptyset$.

In Fig. 2.2, an example is given of a hierarchical decomposition in two dimensions up to two levels. The plane is first divided according to the straight line $\bar{v}_0^T(\bar{x} - \bar{\mu}_0) = 0$. Vectors on the left side of this line belong to S_1, the others belong to S_2. This is continued for two levels, resulting finally in four leaf nodes and an equal number of corresponding regions in the plane. On the right of the figure, a tree representation of the decomposition is depicted.

2.2.2 A Statistical Model of the Evaluation Cost

In this subsection, a statistical model is described that expresses the average computation cost in function of the node traversal cost $\mathbf{C}_{trav}$ and the distance computation cost $\mathbf{C}_{dist}$. For any node p containing N_p vectors that is branched, the conditional probability that a child node of p is traversed can be expressed using

$$\begin{aligned} P(2p+1|p) &= \frac{N_{2p+1}}{N_p} + E_{2p+1}\frac{N_{2p+2}}{N_p} \\ P(2p+2|p) &= \frac{N_{2p+2}}{N_p} + E_{2p+2}\frac{N_{2p+1}}{N_p} \end{aligned} \tag{2.3}$$

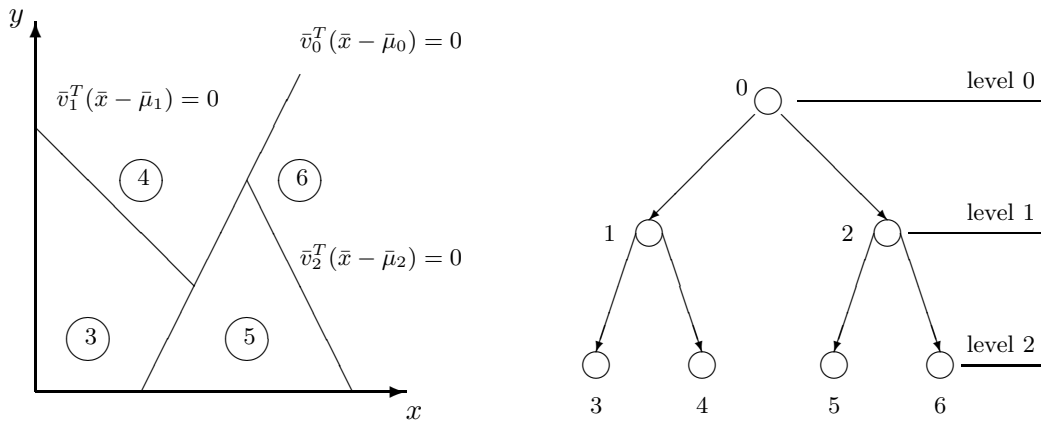


Fig. 2.2: *Example of a hierarchical decomposition in two dimensions.*

The first term in the first equation expresses the probability that $\bar{x}$ lies on the same side of the plane as the child node $2p+1$. This node will be evaluated first. The goal of the elimination rule (see section 2.3) is to avoid the evaluation of the other child node. However, it is possible that the elimination rule fails. This is expressed by the second term which is the product of the probability that $\bar{x}$ belongs to the other node $\frac{N_{2p+2}}{N_p}$ and the probability that the elimination fails E_{2p+1}.

The absolute probability $P(p)$ that a node is traversed is expressed in function of these conditional probabilities by induction

$$\begin{aligned} P(2p+1) &= P(2p+1|p)P(p) \\ P(2p+2) &= P(2p+2|p)P(p) \end{aligned} \tag{2.4}$$

with $P(0)=1$. Thus, each probability $P(p)$ is the product of all conditional probabilities over the tree path from the root node 0 to p, for example $P(3)=P(3|1)P(1|0)P(0)$.

The average evaluation cost C_p of a node p which is branched up to one level is given

by

$$\begin{aligned} C_p &= P(2p+1|p)N_{2p+1}\mathbf{C}_{dist} + \\ &\quad P(2p+2|p)N_{2p+2}\mathbf{C}_{dist} + \mathbf{C}_{trav} \\ &= \left(\frac{N_{2p+1}}{N_p} + E_{2p+1}\frac{N_{2p+2}}{N_p}\right) N_{2p+1}\mathbf{C}_{dist} + \\ &\quad \left(\frac{N_{2p+2}}{N_p} + E_{2p+2}\frac{N_{2p+1}}{N_p}\right) N_{2p+2}\mathbf{C}_{dist} + \mathbf{C}_{trav} \end{aligned} \tag{2.5}$$

Taking into account that $N_{2p+2} = N_p - N_{2p+1}$, this cost is minimized by taking the derivative $\frac{\partial C_p}{\partial N_{2p+1}}$

$$\begin{aligned} \frac{\partial C_p}{\partial N_{2p+1}} &= \frac{\partial}{\partial N_{2p+1}}\left[\left(\frac{N_{2p+1}}{N_p} + E_{2p+1}\frac{N_p - N_{2p+1}}{N_p}\right) N_{2p+1}\mathbf{C}_{trav} + \right. \\ &\quad \left.\left(\frac{N_p - N_{2p+1}}{N_p} + E_{2p+2}\frac{N_{2p+1}}{N_p}\right)(N_p - N_{2p+1})\mathbf{C}_{trav} + \mathbf{C}_{dist}\right] \\ &= \frac{\mathbf{C}_{trav}}{N_p}\left[(1 - E_{2p+1})N_{2p+1} + N_{2p+1} + E_{2p+1}(N_p - N_{2p+1})\right. \\ &\quad \left.+(-1 + E_{2p+2})(N_p - N_{2p+1}) - (N_p - N_{2p+1} + E_{2p+2}N_{2p+1}))\right] \\ &= \frac{\mathbf{C}_{trav}}{N_p}(2 - E_{2p+1} - E_{2p+2})(2N_{2p+1} - N_p) \end{aligned}$$

When this is put to zero, one obtains

$$\begin{aligned} &\frac{\partial C_p}{\partial N_{2p+1}} = 0 \\ \Rightarrow\quad &\frac{\mathbf{C}_{trav}}{N_p}(2 - E_{2p+1} - E_{2p+2})(2N_{2p+1} - N_p) = 0 \\ \Rightarrow\quad &N_{2p+1} = \frac{N_p}{2} \end{aligned}$$

This implies that the evaluation cost of node p will be minimized when each child node contains an equal number of vectors.

When a node p is further decomposed, its evaluation cost C_p is expressed in function of the cost of its children C_{2p+1} and C_{2p+2} by

$$C_p = P(2p+1|p)C_{2p+1} + P(2p+2|p)C_{2p+2} + \mathbf{C}_{trav} \tag{2.6}$$

The total computation cost C_0, starting from the root node and traversing the tree up to a level L, yields

$$C_0 = \mathbf{C}_{trav}\sum_{p=0}^{2^L-2} P(p) + \mathbf{C}_{dist}\sum_{p=2^L-1}^{2^{L+1}-2} P(p)N_p \tag{2.7}$$

The left term expresses the average number of branched nodes that is traversed, multiplied with the traversal cost. The second term, denotes the average number of distance computations, multiplied with the distance computation cost.

From this probabilistic model of the average computation cost, some important observations are already made

1. The decomposition is most efficient when the tree is balanced.

2. The goal of the elimination rule is to make E_p as small as possible in order to minimize the probability that a node is traversed. This minimizes the total computation cost as shown in equation (2.7). On the other hand, the equation shows also that the evaluation cost of the elimination rule, denoted $\mathbf{C}_{trav}$, plays an important role. An elimination rule with a larger value of E_p but with a lower evaluation cost $\mathbf{C}_{trav}$ may result in a smaller overall computation time.

3. The level of decomposition L has a strong influence on the computation time. The first term of Eq. (2.7) increases in function of it while the second term decreases. Therefore an optimal level exists, providing the trade-off between both terms. Note that this is a *user specified parameter* that has to be set manually. The optimization of this parameter is discussed further in section (2.6).

2.2.3 PCA-Based Decomposition

From the previous section was concluded that a balanced tree results in the lowest computation cost. In this section, a second efficiency criterium is proposed. Vectors that lie close to the separating hyperplane have nearest neighbors in both child nodes, which implies that none of the child nodes can be eliminated. Therefore, we wish to determine the hyperplane that minimizes the number of vectors that lie close to it. This results in the following efficiency criteria

1. Both child nodes contain an equal number of vectors.

2. The number of vectors close to the plane is minimal.

The set of vectors lying close to the hyperplane eventually results in the hyperplane itself when the number of samples approaches infinity. This implies that the second criterium is equivalent with determining the hyperplane over which the integral of the probability density function is minimal.

We examine how the efficiency criteria can be applied to a D dimensional gaussian function

$$\exp\left\{-\frac{1}{2}(\bar{x}-\bar{\mu})^T\mathbf{\Sigma}^{-1}(\bar{x}-\bar{\mu})\right\} \tag{2.8}$$

that is fit to a data set S_p where $\bar{\mu}$ is the mean vector

$$\bar{\mu} = \frac{1}{N_p}\sum_{\bar{x}_i \in S_p} \bar{x}_i \tag{2.9}$$

and $\mathbf{\Sigma}$ the $D \times D$ covariance matrix

$$\mathbf{\Sigma} = \frac{1}{N_p} \sum_{\bar{x}_i \in S_p} (\bar{x}_i - \bar{\mu})(\bar{x}_i - \bar{\mu})^T \tag{2.10}$$

Any hyperplane that contains the mean vector $\bar{\mu}$ divides the gaussian in two equal halves what directly optimizes the first criterium. In order to apply the second criterium, we wish to minimize the probability that vectors lie on the separating hyperplane. In other words, the hyperplane must be determined for which the integral of the gaussian over that plane results in the smallest value. The following equations show that this is the hyperplane perpendicular to the maximal variance of the gaussian.

The principal axes of the gaussian are given by the eigenvectors $\bar{u}_i$ of $\mathbf{\Sigma}$ satisfying

$$\mathbf{\Sigma}\bar{u}_i = \lambda_i \bar{u}_i \tag{2.11}$$

where λ_i is the variance or eigenvalue according to the eigenvector $\bar{u}_i$. The eigenvectors can be normalized in order to form a complete orthonormal set

$$\bar{u}_i^T \bar{u}_j = \delta_{ij} \tag{2.12}$$

where δ_{ij} denotes the Kronecker symbol. When a vector $\bar{x}$ is expressed as a linear combination of the eigenvectors $\bar{u}_i$

$$\bar{x} = \bar{\mu} + \sum_{k=1}^{D} \alpha_k \bar{u}_k \tag{2.13}$$

with

$$\alpha_k = (\bar{x} - \bar{\mu})^T \bar{u}_k \tag{2.14}$$

equation (2.8) can be written as

$$\exp\left\{ -\frac{1}{2} \sum_{k=1}^{D} \alpha_k \bar{u}_k^T \mathbf{\Sigma}^{-1} \sum_{l=1}^{D} \alpha_l \bar{u}_l \right\} \tag{2.15}$$

which can be simplified to

$$\exp\left\{ -\frac{1}{2} \sum_{k=1}^{D} \alpha_k^2 \lambda_k^{-1} \right\} \tag{2.16}$$

using (2.11) and (2.12).

When integrating the distribution over all axes determined by the eigenvectors, the integral decouples and yields

$$\prod_{k=1}^{D} \int \exp\left(-\frac{\alpha_k^2}{2\lambda_k} \right) d\alpha_k = \prod_{k=1}^{D} \sqrt{2\pi\lambda_k} \tag{2.17}$$

The inverse of this integral is the normalization factor that is used when a multivariate gaussian is applied as a probability density function. The integral of the gaussian distribution, using this normalization factor, over the hyperplane perpendicular to the principal component $u_{k_{max}}$ is given by

$$\frac{\prod_{k=1,k\neq k_{max}}^{D} \sqrt{2\pi\lambda_k}}{\prod_{k=1}^{D} \sqrt{2\pi\lambda_k}} = \frac{1}{\sqrt{2\pi\lambda_{k_{max}}}} \tag{2.18}$$

Since the principal component gives the direction of the maximal variance, this equation shows that the integral over the hyperplane perpendicular to the eigenvector $\bar{u}_i$ with the largest eigenvalue λ_i (i.e. the principal component) is minimal. Therefore, the decomposition according to this plane will result in the least vectors having nearest neighbors in both subsets. This is the second efficiency criterium given at the beginning of this subsection. As a result, a split of the sample according to this hyperplane is proposed of which the equation is given by $\bar{v}_p^T(\bar{x} - \bar{\mu}_p) = 0$ with

$$\bar{\mu}_p = \bar{\mu} \tag{2.19}$$

$$i_{max} = \text{argmax}_i\{\lambda_i\} \tag{2.20}$$

$$\bar{v}_p = \bar{u}_{i_{max}} \tag{2.21}$$

The efficiency of this decomposition was previously shown experimentally in [52]. However, no theoretical derivation had been provided so far. Fig. 2.3 shows the contour of a two-dimensional gaussian with its corresponding eigenvalues and eigenvectors. On the right, the separating hyperplane and the decomposition parameters of the optimal split are depicted.

2.2.4 Tree Balancing

Although it was shown theoretically that the decomposition as described above provides the optimal split for a gaussian distribution, the tree might not be balanced for real data sets. This means that for a given node one child node may contain significantly more vectors relative to the other. In order to balance the tree, the value of $\bar{\mu}_p$ is chosen to be the median in the direction of $\bar{v}_p$, instead of the mean $\bar{\mu}$. This makes the algorithm more robust when the distribution of the data set is not gaussian. The balancing is realized defining a scalar β

$$\beta = \text{median}\{\bar{v}_p^T(\bar{x}_i - \bar{\mu})\}, \quad \bar{x}_i \in S_p \tag{2.22}$$

and calculating the value of $\bar{\mu}_p$ using

$$\bar{\mu}_p = \beta\bar{v}_p + \bar{\mu} \tag{2.23}$$

Both the definition of the decomposition parameters and the balancing of the tree contribute to the efficiency of the search algorithm.

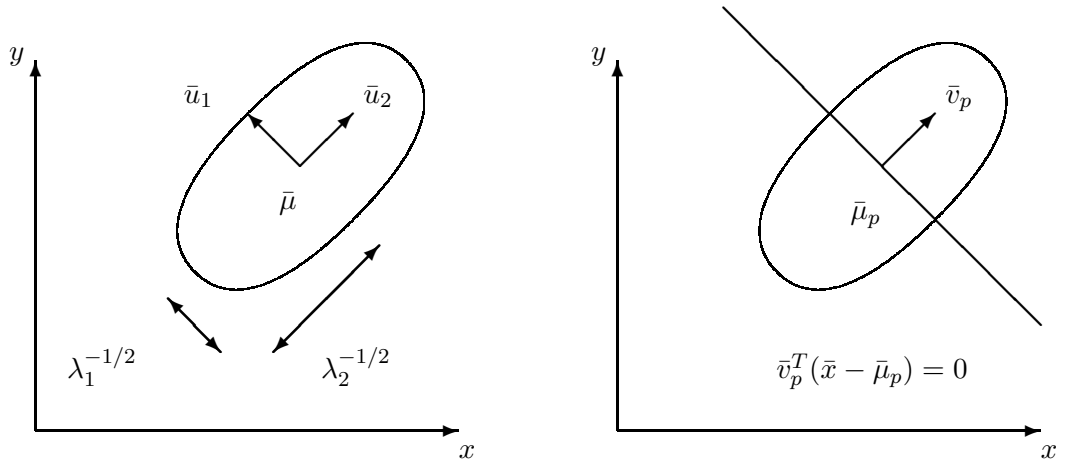


Fig. 2.3: **Left:** *Eigenvalues and Eigenvectors* **Right:** *Optimal Separating Hyperplane*

2.2.5 Axis Segmentation

In [47] and [61], the decomposition was realized with hyperplanes orthogonal to the axis of the vector space. Applying the efficiency criteria given above and using this restriction, we can define a decomposition with as parameters of a node p:

- i_p the index of the axis with the largest variance over S_p.
- s_p the median value of x_{j,i_p} with $\bar{x}_j \in S_p$.

When this decomposition is used, the elimination rule described in section 2.3.4 can be applied.

2.2.6 The Decomposition Algorithm

The decomposition algorithm is described for a data set $S_0 = \{\bar{x}_1, \ldots, \bar{x}_N\}$ up to a level $L \leq \lfloor \log_2 N \rfloor$. The algorithm organizes the vectors so that all vectors belonging to a node are grouped together. In [47], this organization is called an *ordered partition*. For each node p the index of its first vector b_p and last vector e_p are stored so that $S_p = \{\bar{x}_i \in S_0 : b_p \leq i \leq e_p\}$. The decomposition variables that are determined for each node p are

- b_p the index of the first vector of S_p
- e_p the index of the last vector of S_p
- $\bar{v}_p$ eigenvector of S_p with the largest eigenvalue

- $\bar{\mu}_p$ median vector of S_p, in the direction of $\bar{v}_p$

Note that the first two parameters must be determined for all nodes in the tree ($p = 0, \ldots, 2^{L+1} - 2$). The last two are only determined for nodes that are branched ($p = 0, \ldots, 2^L - 2$). The decomposition parameters are computed only once, and are then reused for consecutive searches. The complete decomposition algorithm is listed below.

In Fig. 2.4 and 2.5 some results of the decomposition are shown. All vectors are visualized by drawing a line from each vector to the mean vector of the node it belongs to. In Fig. 2.4 the data set consists of 2^{10} two-dimensional vectors drawn from a normal distribution with mean $\bar{0}$ and unit covariance matrix. The decomposition is realized up to the sixth level resulting in 64 nodes each containing 16 vectors. In Fig. 2.5 the same decomposition is shown for a distribution with a different covariance matrix.

2.3 Node Elimination

The branch and bound algorithm searches for the nearest neighbors of a given vector $\bar{x}$ and consists of a depth first traversal of the tree that represents the hierarchical decomposition of the data set. When a node is evaluated, it is determined whether it can contain nearest neighbors. If this is not the case, this node can be omitted from the search procedure. The rule that is used to determine whether a node can contain nearest neighbors is called the *elimination rule*. The elimination rule is adapted to the definition of the decomposition and relies on a *lower bound distance measure* $d(\bar{x}, p)$ between a vector $\bar{x}$ and a node with index p. This distance is also called the *vector-to-node distance*. By comparing $d(\bar{x}, p)$ with the distance to the Kth nearest neighbor, it is determined whether this node can be discarded from the search procedure. The correctness of the algorithm follows directly from the definition of the decomposition and the definition of an elimination rule. Different elimination rules were proposed in [30], [52] and [19].

2.3.1 D'haes Elimination Rule

In [19, 21], a vector-to-node distance $d(\bar{x}, p)$ was proposed that is defined to be 0 when p is the root node. The distances to underlying nodes of p are determined from $d(\bar{x}, p)$ using

$$\begin{array}{ll} \beta = \bar{v}_p^T(\bar{x} - \bar{\mu}_p) & \\ \texttt{if} \quad \beta < 0 & \\ \qquad d(\bar{x}, 2p+1) = d(\bar{x}, p) & \\ \qquad d(\bar{x}, 2p+2) = \max(d(\bar{x}, p), |\beta|) & \\ \texttt{else} & \\ \qquad d(\bar{x}, 2p+1) = \max(d(\bar{x}, p), |\beta|) & \\ \qquad d(\bar{x}, 2p+2) = d(\bar{x}, p) & \end{array} \tag{2.24}$$

$S_0 = \{\bar{x}_1, \ldots, \bar{x}_N\}$
$b_0 = 1$
$e_0 = N$
$p = 0$
`while` $p < 2^L - 1$
    `// Calculation of the decomposition parameters`
    $N_p = e_p - b_p + 1$
    $\bar{\mu} = \frac{1}{N_p} \sum_{k=b_p}^{e_p} \bar{x}_k$
    $\mathbf{\Sigma} = \frac{1}{N_p} \sum_{k=b_p}^{e_p} (\bar{x}_k - \bar{\mu})(\bar{x}_k - \bar{\mu})^T$
    $\bar{u}_i, \lambda_i = \text{eigenvectors}(\mathbf{\Sigma})$
    $i_{max} = \text{argmax}_i\{\lambda_i\}$
    $\bar{v}_p = \bar{u}_{i_{max}}$
    $\beta = \text{median}\{\bar{v}_p^T(\bar{x}_k - \bar{\mu}), b_p \leq k \leq e_p\}$
    $\bar{\mu}_p = \beta \bar{v}_p + \bar{\mu}$
    `// Organization of the order`
    $i = b_p$
    $j = e_p$
    `while` $j > i$
        `while` $\bar{v}_p^T(\bar{x}_i - \bar{\mu}_p) \leq 0$
            $i = i + 1$
        `end`
        `while` $\bar{v}_p^T(\bar{x}_j - \bar{\mu}_p) > 0$
            $j = j - 1$
        `end`
        `// Exchange position of` $\bar{x}_i$ `and` $\bar{x}_j$
        `if` $j > i$
            $SWAP(\bar{x}_i, \bar{x}_j)$
        `end`
    `end`
    $b_{2p+1} = b_p$
    $e_{2p+1} = j$
    $b_{2p+2} = i$
    $e_{2p+2} = e_p$
    $p = p + 1$
`end`

Algorithm 1: Hierarchical decomposition algorithm.

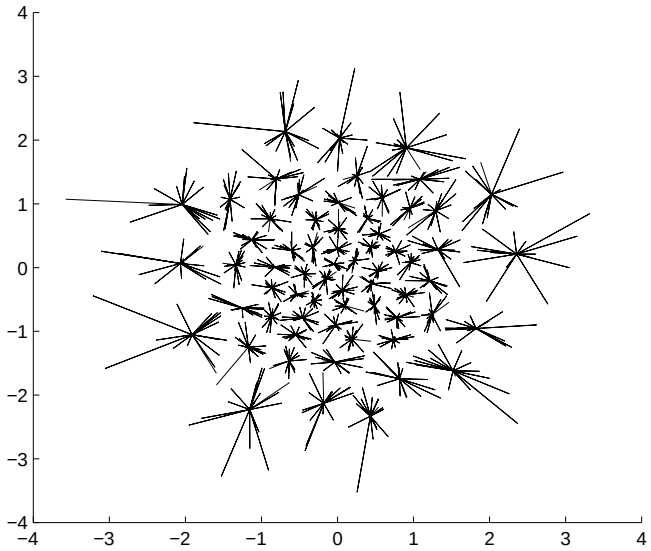


Fig. 2.4: *Decomposition of a normal distribution with unit covariance matrix*

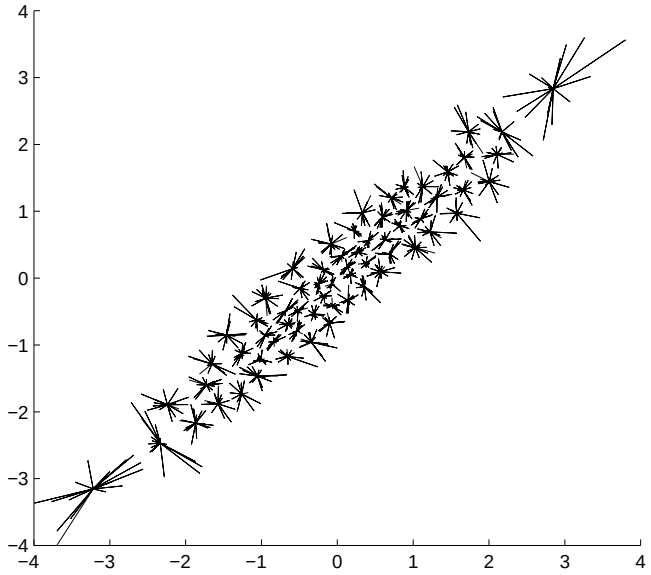
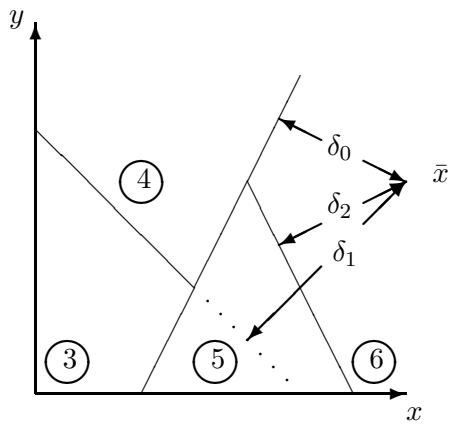


Fig. 2.5: *Decomposition of a distribution with a non unit covariance matrix.*



$$
\begin{aligned}
\delta_0 &= |\bar{v}_0^T(\bar{x}-\bar{\mu}_0)| \\
\delta_1 &= |\bar{v}_1^T(\bar{x}-\bar{\mu}_1)| \\
\delta_2 &= |\bar{v}_2^T(\bar{x}-\bar{\mu}_2)| \\
d(\bar{x},0) &= 0 \\
d(\bar{x},1) &= \max(d(\bar{x},0), |\bar{v}_0^T(\bar{x}-\bar{\mu}_0)|) \\
&= |\bar{v}_0^T(\bar{x}-\bar{\mu}_0)| = \delta_0 \\
d(\bar{x},2) &= d(\bar{x},0) = 0 \\
d(\bar{x},3) &= \max(d(\bar{x},1), |\bar{v}_1^T(\bar{x}-\bar{\mu}_1)|) \\
&= |\bar{v}_1^T(\bar{x}-\bar{\mu}_1)| = \delta_1 \\
d(\bar{x},4) &= d(\bar{x},1) = |\bar{v}_0^T(\bar{x}-\bar{\mu}_0)| = \delta_0 \\
d(\bar{x},5) &= \max(d(\bar{x},2), |\bar{v}_2^T(\bar{x}-\bar{\mu}_2)|) \\
&= |\bar{v}_2^T(\bar{x}-\bar{\mu}_2)| = \delta_2 \\
d(\bar{x},6) &= d(\bar{x},2) = d(\bar{x},0) = 0
\end{aligned}
$$

Fig. 2.6: *Illustration of the D'haes elimination rule.*

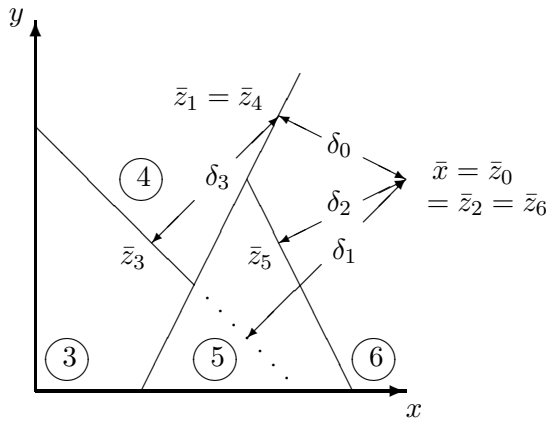
For the child node that contains the vector $\bar{x}$, the same distance is taken as the parent node. For the other child node, all vectors belonging to it have a greater distance to $\bar{x}$ than the perpendicular distance $|\bar{v}_p^T(\bar{x}-\bar{\mu}_p)|$ to the hyperplane. This distance might therefore be a valid definition for the vector-to-node distance. However, hyperplanes on previous levels might provide larger, thus more efficient, distances. Therefore, the maximum of $d(\bar{x},p)$ and $|\bar{v}_p^T(\bar{x}-\bar{\mu}_p)|$ is taken.

In order to clarify this distance measure, the decomposition depicted in Fig. 2.2 is taken and the distances from a given node $\bar{x}$ to all nodes are depicted in Fig. 2.6. Note especially the distance $d(\bar{x},3)$ where one has the choice between $|\bar{v}_0^T(\bar{x}-\bar{\mu}_0)|$ and $|\bar{v}_1^T(\bar{x}-\bar{\mu}_1)|$. Since the second distance is larger, it is more likely that it will eliminate nodes from the search and is therefore chosen to be $d(\bar{x},3)$.

During the search procedure, the currently found nearest neighbors and their distance to the vector $\bar{x}$ are stored in the variables $\bar{y}_k$ and d_k respectively with $k=1,\ldots,K$. The values of d_k are initially set at ∞. The index of the vector $\bar{y}$ that is the current Kth nearest neighbor is denoted k_{max}. When a vector is found that is closer than $\bar{y}_{k_{max}}$, it is replaced by this nearer vector. Then, it is determined which of $\bar{y}_k$ is the new Kth nearest neighbor which results in a new value for k_{max}. By the definition of the vector-to-node distance, the following elimination rule can be applied. A node p can be discarded from the search procedure if

$$d(\bar{x},p)^2 > (\bar{y}_{k_{max}}-\bar{x})^T(\bar{y}_{k_{max}}-\bar{x}) \tag{2.25}$$

since all the vectors belonging to p are further from $\bar{x}$ than $\bar{y}_{k_{max}}$.



$$
\begin{aligned}
&\delta_0 = |\bar{v}_0^T(\bar{x} - \bar{\mu}_0)| \\
&\delta_1 = |\bar{v}_1^T(\bar{x} - \bar{\mu}_1)| \\
&\delta_2 = |\bar{v}_2^T(\bar{x} - \bar{\mu}_2)| \\
&\delta_3 = |\bar{v}_1^T(\bar{z_1} - \bar{\mu}_1)| \\
&d(\bar{x}, 0) = 0 \\
&d(\bar{x}, 1) = \sqrt{d(\bar{x}, 0)^2 + \delta_0^2} = \delta_0 \\
&d(\bar{x}, 2) = d(\bar{x}, 0) = 0 \\
&d(\bar{x}, 3) = \sqrt{d(\bar{x}, 1)^2 + \delta_3^2} = \sqrt{\delta_0^2 + \delta_3^2} \\
&d(\bar{x}, 4) = d(\bar{x}, 1) = \delta_0 \\
&d(\bar{x}, 5) = \sqrt{d(\bar{x}, 2)^2 + \delta_2^2} = \delta_2 \\
&d(\bar{x}, 6) = d(\bar{x}, 2) = d(\bar{x}, 0) = 0 \\
&\bar{z_1} = x - \delta_0 \bar{v_0} \\
&\bar{z_3} = z_1 - \delta_3 \bar{v_1} \\
&\bar{z_5} = x - \delta_2 \bar{v_2}
\end{aligned}
$$

Fig. 2.7: *Illustration of the McNames elimination rule.*

2.3.2 McNames Elimination Rule

In [52], another elimination rule is used which projects $\bar{x}$ iteratively on the hyperplane. It is also defined inductively from $d(\bar{x}, p)$ and $\bar{z}_p$. The values of $d(\bar{x}, 0)$ and $\bar{z}_0$ are initially set to 0 and $\bar{x}$ respectively. Its definition is given by

$$
\begin{aligned}
&\beta = \bar{v}_p^T(\bar{z}_p - \bar{\mu}_p) \\
&\texttt{if} \quad \beta < 0 \\
&\qquad d(\bar{x}, 2p+1) = d(\bar{x}, p) \\
&\qquad \bar{z}_{2p+1} = \bar{z}_p \\
&\qquad d(\bar{x}, 2p+2) = \sqrt{d(\bar{x}, p)^2 + \beta^2} \\
&\qquad \bar{z}_{2p+2} = \bar{z}_p - \beta \bar{v}_p \\
&\texttt{else} \\
&\qquad d(\bar{x}, 2p+1) = \sqrt{d(\bar{x}, p)^2 + \beta^2} \\
&\qquad \bar{z}_{2p+1} = \bar{z}_p - \beta \bar{v}_p \\
&\qquad d(\bar{x}, 2p+2) = d(\bar{x}, p) \\
&\qquad \bar{z}_{2p+2} = \bar{z}_p
\end{aligned}
\tag{2.26}
$$

In Fig. 2.7, the McNames lower bound distance is illustrated.

2.3.3 Fukunaga Elimination Rule

In order to apply the Fukunaga elimination rule [30], an additional decomposition parameter must be computed. For each node p, the distance for each vector to the median

vector $\bar{\mu}_p$ is computed and only the largest value r_p is retained.

$$r_p = \max_{\bar{x}_i \in S_p} \left(\sqrt{(\bar{x}_i - \mu_p)^T (\bar{x}_i - \mu_p)} \right) \tag{2.27}$$

Thus all vectors $\bar{x}_i$ belonging to a node S_p lie in a hypersphere with mean $\bar{\mu}_p$ and radius r_p. When d_{kmax} denotes the distance to the Kth nearest neighbor, the following condition

$$r_p + d_{kmax} < \sqrt{(\bar{x} - \bar{\mu}_p)^T (\bar{x} - \bar{\mu}_p)} \tag{2.28}$$

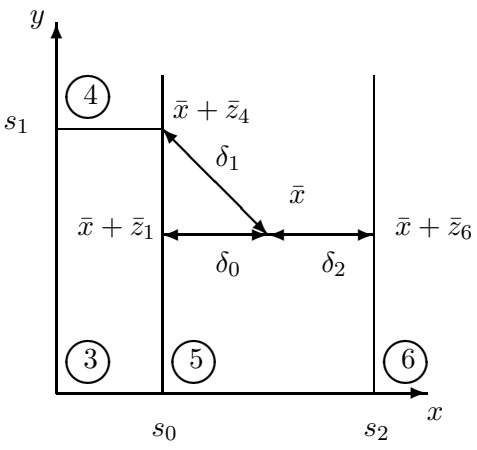
expresses that the hypersphere cannot contain nearest neighbors of $\bar{x}$. This follows directly from the triangle inequality [30]. Therefore, node p can be eliminated.

2.3.4 Kim and Park Elimination Rule

Using the decomposition method orthogonal to the axis that was described in section 2.2.5, another elimination rule can be applied which is defined by

$$\begin{array}{ll}
\beta = x_{i_p} - s_p & \\
\texttt{if} & \beta \leq 0 \\
& d(\bar{x}, 2p+1) = d(\bar{x}, p) \\
& \bar{z}_{2p+1} = \bar{z}_p \\
& d(\bar{x}, 2p+2) = \sqrt{d(\bar{x}, p)^2 - z_{p,i_p}^2 + \beta^2} \\
& \bar{z}_{2p+2} = \bar{z}_p \\
& z_{2p+2,i_p} = -\beta \\
\texttt{else} & \\
& d(\bar{x}, 2p+1) = \sqrt{d(\bar{x}, p)^2 - z_{p,i_p}^2 + \beta^2} \\
& \bar{z}_{2p+1} = \bar{z}_p \\
& z_{2p+1,i_p} = -\beta \\
& d(\bar{x}, 2p+2) = d(\bar{x}, p) \\
& \bar{z}_{2p+2} = \bar{z}_p
\end{array} \tag{2.29}$$

In this case, $\bar{z}_0$ and $d(\bar{x}, 0)$ are initialized as $\bar{0}$ and 0, respectively. An example is given in Fig. 2.8.



$$\delta_0 = d(\bar{x}, 1)$$
$$\delta_1 = d(\bar{x}, 4)$$
$$\delta_2 = d(\bar{x}, 6)$$
$$\bar{z}_0 = \bar{z}_2 = \bar{z}_5 = \bar{0}$$
$$\bar{z}_1 = \bar{z}_3 = (s_0 - \bar{x}_1, 0)$$
$$\bar{z}_4 = (s_0 - x_1, s_1 - x_2)$$
$$\bar{z}_6 = (s_2 - x_1, 0)$$
$$d(\bar{x}, 0) = d(\bar{x}, 2) = d(\bar{x}, 5) = 0$$
$$d(\bar{x}, 1) = d(\bar{x}, 3) = |s_0 - x_1|$$
$$d(\bar{x}, 4) = \sqrt{(s_0 - x_1)^2 + (s_1 - x_2)^2}$$
$$d(\bar{x}, 6) = |s_2 - x_1|$$
$$i_0 = 1,\ i_1 = 2,\ i_2 = 1$$

Fig. 2.8: *Illustration of the Kim and Park elimination rule.*

2.4 The Search Algorithm

2.4.1 Outline of the Algorithm

The tree that represents the decomposition of the data sample is traversed in a depth first order, which can be implemented efficiently using a stack s which is addressed by an index t. On this stack, the node index p of nodes that still need to be evaluated are stored together with the data that is needed for the evaluation of the elimination rule. This data is needed to compute the vector-to-node distances that are defined by induction. For the D'haes elimination rule this data consists of $d(\bar{x}, p)$. For the McNames elimination rule and Kim and Park elimination rule, an additional vector $\bar{z}_p$ is needed. The Fukunaga elimination rule is the only rule that does not require any extra data on the stack since it is not defined by induction.

The search is started by pushing the root node on the stack. When a node is evaluated, it is popped of the stack and the elimination rule is evaluated. When the elimination rule indicates that the node cannot contain nearest neighbors, the following node on the stack is evaluated. If not, two cases can be distinguished. If the node is a leaf node ($p \geq 2^L - 1$) the vectors in $S_p = \{\bar{x}_{b_p}, \ldots, \bar{x}_{e_p}\}$ are searched which means that their distance to $\bar{x}$ is calculated and compared with $d_{k_{max}}$. If the node is branched, the child nodes and their distances are pushed on the stack. The algorithm terminates when the stack is empty ($t < 0$) which indicates that the entire tree was traversed and the K nearest neighbors found.

A complete search algorithm using the D'haes elimination rule (see section 2.3.1) is listed below. The K nearest neighbors of a vector $\bar{x}$ are determined from a data set S_0 using a decomposition level L.

$d_1, \ldots, d_K = \infty$
$S_0 = \{\bar{x}_1, \ldots, \bar{x}_N\}$
$k_{max} = 1$
$s_0 = 0$ // push the root node
$s_1 = 0$ // push the distance
$t = 1$
while $t \geq 0$
    $dxp = s_t$ // pop distance
    $p = s_{t-1}$ // pop node
    $t = t - 2$
    if $dxp < d_{k_{max}}$ // elimination rule
        if $p \geq 2^L - 1$ // p is a leaf node
            $i = b_p$
            while $i \leq e_p$
                $dxx = (\bar{x} - \bar{x}_i)^T(\bar{x} - \bar{x}_i)$
                if $dxx < d_{k_{max}}$
                    $\bar{y}_{k_{max}} = \bar{x}_i$
                    $d_{k_{max}} = dxx$
                    $k_{max} = \max \arg_k \{d_k\}$
                end
                $i = i + 1$
            end
        else // p is a branched node
            if $\bar{v}_p(\bar{x} - \bar{\mu}_p) < 0$
                $s_{t+1} = 2p + 2$
                $s_{t+2} = \max(dxp, (\bar{v}_p^T(\bar{x} - \bar{\mu}_p))^2)$
                $s_{t+3} = 2p + 1$
                $s_{t+4} = dxp$
                $t = t + 4$
            else
                $s_{t+1} = 2p + 1$
                $s_{t+2} = \max(dxp, (\bar{v}_p^T(\bar{x} - \bar{\mu}_p))^2)$
                $s_{t+3} = 2p + 2$
                $s_{t+4} = dxp$
                $t = t + 4$
            end
        end
    end
end

Algorithm 2: Search Algorithm.

D	K	Number of vectors in data set N					
		2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}
2	1	4.5	4.7	4.7	4.9	4.7	4.7
	2	7.7	7.9	8.0	7.9	8.0	7.9
	4	13.5	13.8	13.6	13.6	13.5	13.5
	8	23.7	23.8	23.8	23.9	23.8	24.0
4	1	34	35	37	39	39	41
	2	51	55	58	62	62	63
	4	79	87	93	98	101	102
	8	121	136	149	157	163	168
8	1	371	544	739	968	1197	1515
	2	495	736	1025	1380	1789	2210
	4	608	954	1388	1915	2534	3203
	8	726	1185	1785	2567	3443	4483

Tab. 2.1: *Average number of distance computations*

2.4.2 Number of distance computations

The behavior of the algorithm was studied by means of experiments on artificial data. Prototype sets were produced from a D-dimensional normal probability distribution with mean 0 and unit covariance matrix. Each result was obtained from the average of 2^{10} experiments. The performance of the algorithm was studied with respect to different values of

- D the dimensionality of the vector space
- N the number of vectors in the data set
- K the number of nearest neighbors that are searched
- L the level of decomposition that is used for the search

In table 2.1 the average number of vector-to-vector distance computations is given. From this table, one can observe that the average number of distance computations is in general very small. The nearest neighbor in a 2-dimensional space was obtained after 4.7 distance computations. Other works report results of 46 [30] and 165 [44] average distance computations. An interesting overview is given in [94]. The number of distance computations tends to be independent of the number of prototypes (for a dimensionality of $D \leq 4$). This property was also observed from the AESA algorithm and its derivatives.

2.4.3 Total Computation Time

In addition to these distance computations, the search algorithm also spends time calculating the vector-to-node distance and traversing the tree. If the cost of the distance

computation is very high relative to this extra effort, this could be neglected. However, when observing the average calculation time of the algorithm for increasing levels of decomposition, the total calculation time decreases fast at the lower levels, obtains a minimum and increases towards the highest level. Every extra level of decomposition reduces the number of distance computations but increases the traversal cost. If this extra cost exceeds the reduction in distance calculation time, an additional level of decomposition will increase the total search time. This implies that there is an optimal level L_{opt} for which the total computation time is minimal. Using this optimal level of decomposition, the average computation time was determined in function of the number of vectors N. Results for K being 1, 2, 4 and 8 are shown for different dimensionalities D in figures 2.9 to 2.11.

2.5 Locally Versus Globally Optimized Traversal Order

As can be observed from the search algorithm, the node with the smallest value for $d(\bar{x}, p)$ is pushed on the stack last so that it is evaluated first. This optimizes *locally* the traversal order of the search. However, the distance to nodes at higher levels might be smaller and more interesting to evaluate first. This can be realized by sorting the nodes stored in s by their lower bound distance $d(\bar{x}, p)$ after they are pushed on the stack. The stack s is then an ordered list where closest nodes are stored on top. Doing so, a *globally* optimized traversal order can be realized. Evidently, this sorting will only imply an efficiency gain when its computation cost is fairly low.

An example is given in Fig. 2.12 where the globally optimized traversal reduces the number of distance computations. The figure shows a hierarchic decomposition of a data set from which the nearest neighbors of a vector $\bar{x}$ are determined. Since the vector lies in section 5 of the decomposition the nodes 0 and 2 are traversed first while the nodes 1 and 6 are pushed on the stack for later evaluation. The locally optimized traversal will evaluate the node 6 before node 3 although the lower bound distance to 3 is significantly smaller. By comparing the lower bound distances to the nodes it is determined that it is more efficient when node 1 is traversed first which will eventually lead to node 3.

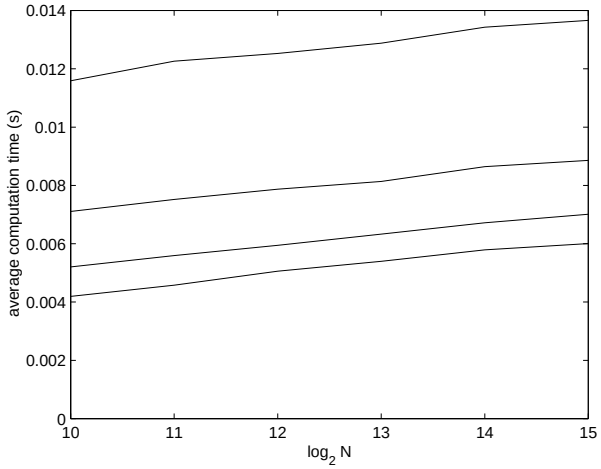


Fig. 2.9: *Average computation time for* $D = 2$

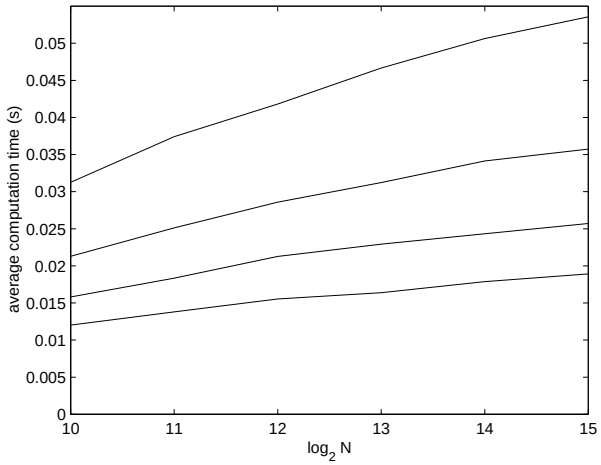


Fig. 2.10: *Average computation time for* $D = 4$

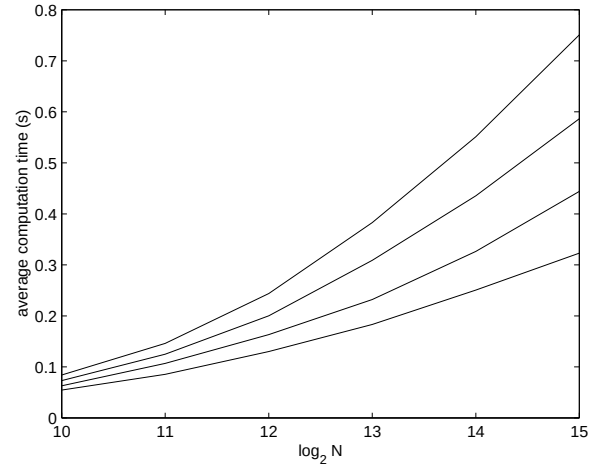


Fig. 2.11: *Average computation time for* $D = 8$

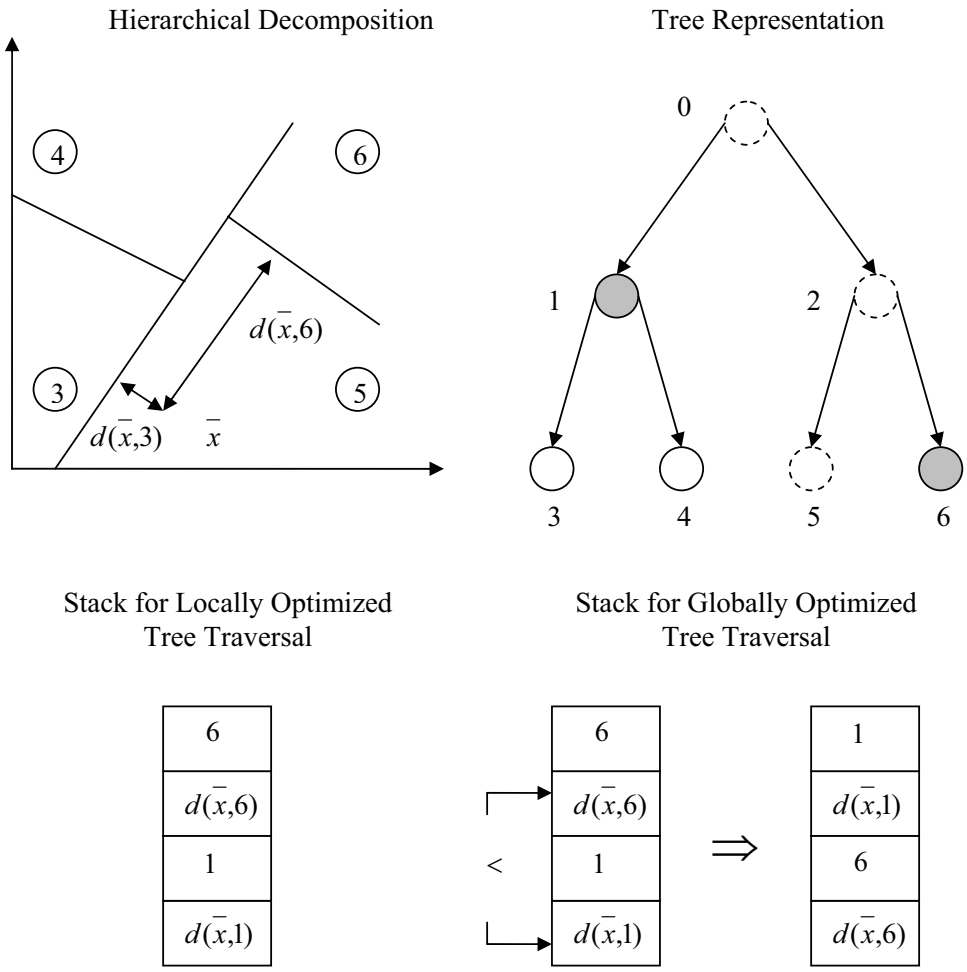


Fig. 2.12: *Locally versus Globally Optimized Traversal*

2.6 Optimization of the Decomposition Level

Although the goal one wishes to achieve is the minimization of the total calculation time, most articles only report the average number of distance computations. This allows to compare different search algorithms but does not take into account the overhead for the tree traversal and vector-to-node distance computation. The distance computation cost decreases up to the maximal level of decomposition L_{max}, being $\lfloor \log_2 N \rfloor$. By contrast, the total node traversal cost increases in function of L. Therefore, there is an optimal level of decomposition L_{opt} which minimizes the total computation time, as expressed by equation (2.7).

An estimation technique is proposed that determines the optimal level of decomposition L_{opt} from a few simple experiments. When doing I experiments for different levels of decomposition L_i with $i = 1, \ldots, I$, a set of total traversal cost values $C_{0,i}$ is obtained. When the search is executed, a counter t_p is incremented each time a node p is traversed. This counter is only incremented when the elimination of the node fails. The conditional and absolute probabilities are determined using

$$P(2p+1|p) = \frac{t_{2p+1}}{t_p} \tag{2.30}$$

$$P(p) = \frac{t_p}{t_0} \tag{2.31}$$

from which the total number of traversed branched nodes T_{trav} and distance computations T_{dist} can be computed for each experiment i yielding

$$\begin{aligned} T_{trav,i} &= \sum_{p=1}^{2^{L_i}-2} \frac{t_p}{t_0} \\ T_{dist,i} &= \sum_{p=2^{L_i}-1}^{2^{L_i+1}-2} \frac{t_p N_p}{t_0} \end{aligned}$$

From Eq. (2.7), the following error function can be derived

$$\chi(\mathbf{C}_{trav}, \mathbf{C}_{dist}) = \sum_{i=1}^{I} \left(C_{0,i} - \mathbf{C}_{trav} T_{trav,i} + \mathbf{C}_{dist} T_{dist,i}\right)^2 \tag{2.32}$$

The minimization of this function by putting the partial derivatives with respect to $\mathbf{C}_{dist}$ and $\mathbf{C}_{trav}$ to zeros yields

$$\begin{bmatrix} \sum_{i=1}^{I} T_{trav,i}^2 & \sum_{i=1}^{I} T_{trav,i} T_{dist,i} \\ \sum_{i=1}^{I} T_{trav,i} T_{dist,i} & \sum_{i=1}^{I} T_{dist,i}^2 \end{bmatrix} \begin{bmatrix} \mathbf{C}_{trav} \\ \mathbf{C}_{dist} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{I} T_{trav,i} C_{0,i} \\ \sum_{i=1}^{I} T_{dist,i} C_{0,i} \end{bmatrix} \tag{2.33}$$

The solution of these equations yields the least squares estimate for the traversal cost and distance computation cost which are denoted $\tilde{\mathbf{C}}_{trav}$ and $\tilde{\mathbf{C}}_{dist}$ respectively. With

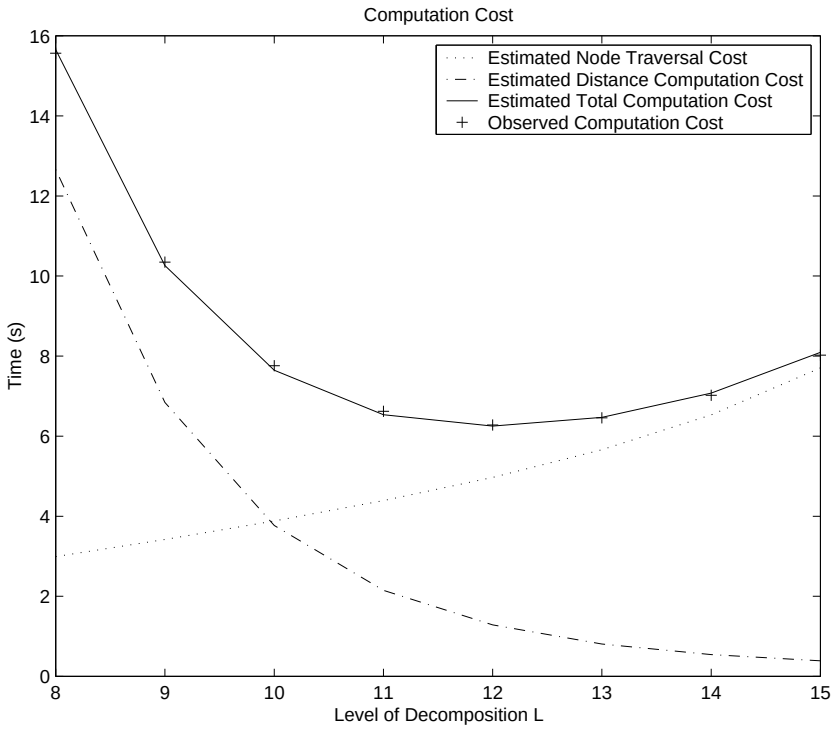


Fig. 2.13: *Computation time (seconds) in function of the level of decomposition for* $D = 2$ *and* $K = 1$

these values and the absolute probabilities $P(p)$, the estimated computation time $\tilde{C}_{0,L}$ is determined for each decomposition level L

$$\tilde{C}_{0,L} = \tilde{\mathbf{C}}_{trav} \sum_{p=0}^{2^L-2} P(p) + \tilde{\mathbf{C}}_{dist} \sum_{p=2^L-1}^{2^{L+1}-2} P(p) N_p \tag{2.34}$$

from which the optimal level L_{opt} can be derived

$$L_{opt} = \arg\min_{L} \tilde{C}_{0,L} \tag{2.35}$$

In addition, the total node traversal cost and distance computation cost can be determined separately. In figure 2.13, the computation time is plot for 2^{10} searches of the nearest neighbor from a two-dimensional uniform data set containing 2^{15} vectors. This figure shows how the traversal cost increases and the distance computation cost decreases in function of the level of decomposition. In this example, a minimum is obtained at level 12.

Since Eq. (2.33) determines two unknown variables, minimum two experiments are required. One of the experiments must use the maximal level L_{max} of decomposition so that the traversal probabilities for all nodes are known.

2.7 A Comparative Evaluation of PCA-Based Search Algorithms

2.7.1 PCA-Based Algorithms

So far, different decomposition methods, elimination rules and traversal orders were described which can be combined in order to obtain different branch and bound search algorithms. The decomposition methods are:

1. decomposition orthogonal to the maximal variance (2.2.3)
2. decomposition orthogonal to the axis with the maximal variance (2.2.5)

The elimination rules are:

1. D'haes elimination rule (2.3.1)
2. McNames elimination rule (2.3.2)
3. Fukunage elimination rule (2.3.3)
4. Kim and Park elimination rule (2.3.4)

The traversal order can be optimized

1. locally
2. globally

Not every combination of these aspects of the search algorithm is possible. For example, the Kim and Park elimination rule can only be applied when the decomposition is orthogonal to the axis. One can also combine different elimination rules, however not all combinations are useful. For example, the McNames elimination rule will always yield a larger value for the lower bound distance $d(\bar{x}, p)$ than the D'haes elimination rule. Therefore, the combination of the two rules will only increase the node traversal cost without decreasing the number of distance computations.

In table 2.2, a number of valid and useful combinations of the different aspects of the search algorithms are given resulting in ten algorithms that will be evaluated in the next sections.

2.7.2 Experiments

The behavior of the algorithm is studied by means of experiments on artificial data. Prototype sets were produced from a D-dimensional normal probability distribution with mean 0 and unit covariance matrix. The performance of the algorithm is studied with respect to different values of the following parameters

- D the dimensionality of the vector space
- N the number of vectors in the data set

Nr.	Decomposition		Elimination				Order	
	1	2	1	2	3	4	1	2
1	√		√				√	
2	√		√					√
3	√		√		√		√	
4	√		√		√			√
5	√			√			√	
6	√			√				√
7	√			√	√		√	
8	√			√	√			√
9		√				√	√	
10		√				√		√

Tab. 2.2: *Different types of search algorithms*

- K the number of nearest neighbors that are searched
- L the level of decomposition that is used for the search

Since L is a value that can be set by the user, we optimize this value first using the statistical model of the total computation cost that was described previously in section 2.6.

2.7.3 Results for Gaussian Distributions

The efficiency of the algorithms was tested on artificial data sets of 2^{15} vectors drawn from a uniform distribution. In the following tables the total computation time is given for 2^{10} experiments and using the optimal level of decomposition L_{opt}. The total computation cost over all experiments is

$$C_0 = \mathbf{C}_{trav} T_{trav} + \mathbf{C}_{dist} T_{dist} \tag{2.36}$$

where T_{trav} and T_{dist} denote now the total number of traversed nodes and the total number of distance computations.

In table (2.3) and (2.4) the total computation time for the experiment C_0 is given for all algorithms. In addition, the values of T_{trav}, $\mathbf{C}_{trav}$, T_{dist} and $\mathbf{C}_{dist}$ are given. From this data, the contradiction between the evaluation based on the number of distance computations and the total computation time is clearly shown. For example, for algorithm 1 (D'haes elimination rule and locally optimized traversal) the highest number of traversed nodes and distance computations is observed. However, for $D = 2$ still the lowest computation time is achieved due to the low node traversal cost $\mathbf{C}_{trav}$. The total computation time for 2^{10} exhaustive searches of the nearest neighbor from the two-dimensional data set containing 2^{15} nodes was 2407 seconds. Note that the Fukunaga

	L_{opt}	C_0	T_{trav}	$\mathbf{C}_{trav}$	T_{dist}	$\mathbf{C}_{dist}$
1	12	6.31	14325	34.4×10^{-5}	16216	7.83×10^{-5}
2	12	8.96	14291	52.8×10^{-5}	15912	7.88×10^{-5}
3	12	7.71	14265	44.8×10^{-5}	16160	7.82×10^{-5}
4	12	11.0	14231	67.1×10^{-5}	15856	8.14×10^{-5}
5	12	8.95	14289	49.6×10^{-5}	15992	8.24×10^{-5}
6	12	11.4	14254	69.4×10^{-5}	15688	8.26×10^{-5}
7	12	10.0	14223	60.0×10^{-5}	15920	8.23×10^{-5}
8	12	12.8	14188	80.0×10^{-5}	15616	8.25×10^{-5}
9	12	8.05	14293	47.7×10^{-5}	15400	7.79×10^{-5}
10	12	11.6	14101	77.2×10^{-5}	14864	7.24×10^{-5}

Tab. 2.3: *Total computation time for $D = 2$ and $K = 1$ for all algorithms.*

elimination rule was not applied alone because its performance was significantly slower. For $K = 1$ and $D = 2$, a computation time of 80.7 seconds was obtained.

Algorithms that incorporate the Fukunaga elimination rule (algorithms 3,4,7 and 8), eliminate nodes more efficiently but have a slightly increased node traversal cost. Due to this increased traversal cost, no gain is achieved for $D = 2$. For $D = 8$ however, a significant improvement is realized. The same conclusion can be drawn for the comparison between the D'haes elimination rule (algorithms 1-4) and the McNames elimination rule (algorithms 5-8). For $D = 2$ the D'haes elimination rule outperforms the other algorithms due to its low node traversal cost. The more efficient elimination capabilities of the other elimination rules dominate for data sets with a higher number of dimensions. The global optimization of the traversal order improves the elimination power of the algorithm. However, due to the significant increase of the node traversal cost, no gain in total computation time was achieved. Finally, it is interesting to note that also the Kim and Park elimination rule (algorithm 9) performs very well.

Table 2.5 shows the computation time for different values of K. The total computation cost increases in a sublinear way in function of K.

2.7.4 Results for Other Data Sets

After performing tests on gaussian distributions, the experiment was repeated in the two-dimensional case for two other data sets:

- A *correlated gaussian distribution* with eigenvalues of the covariance matrix being $\lambda_1 = 1$ and $\lambda_2 = 4$.

- A *mixture of four gaussian distributions* realized by first drawing four points from a uniform distribution with variance one. Then, for each of these points a gaussian cluster with variance $\frac{1}{10}$ was drawn.

	L_{opt}	C_0	T_{trav}	$\mathbf{C}_{trav}$	T_{dist}	$\mathbf{C}_{dist}$
1	11	332	307053	34.4×10^{-5}	3142880	7.21×10^{-5}
2	10	370	180196	66.9×10^{-5}	3581952	7.08×10^{-5}
3	11	124	134118	54.0×10^{-5}	716720	7.23×10^{-5}
4	10	159	91924	91.5×10^{-5}	1097376	7.08×10^{-5}
5	11	201	191189	43.5×10^{-5}	1628992	7.22×10^{-5}
6	10	245	120090	99.6×10^{-5}	1960672	7.08×10^{-5}
7	11	123	120262	62.9×10^{-5}	666336	7.18×10^{-5}
8	10	159	81905	118×10^{-5}	988320	6.89×10^{-5}
9	11	190	184627	44.3×10^{-5}	1474272	7.52×10^{-5}
10	10	256	118018	140×10^{-5}	1817600	6.14×10^{-5}

Tab. 2.4: *Total computation time for $D = 8$ and $K = 1$ for all algorithms.*

D	2				4				8			
K	1	2	4	8	1	2	4	8	1	2	4	8
1	6.31	7.45	9.37	14.1	19.6	26.2	35.8	53.5	332	456	616	814
2	8.96	10.3	12.3	16.8	24.2	30.9	40.9	59.3	370	504	676	872
3	7.71	9.01	11.1	16.1	20.1	25.7	34.2	49.6	124	168	228	309
4	11.0	11.9	14.2	19.0	24.8	30.8	40.0	56.3	159	219	294	403
5	8.95	9.52	11.5	16.3	20.6	26.2	34.6	62.3	201	266	350	456
6	11.4	12.3	14.4	18.9	24.9	30.9	39.9	91.9	245	318	405	527
7	10.0	11.1	13.2	18.1	21.4	26.8	35.1	58.0	123	168	228	310
8	12.8	13.6	15.8	20.5	25.7	31.5	40.2	92.3	159	222	297	400
9	8.05	10.3	12.5	17.9	23.0	29.3	38.7	54.9	190	253	339	463
10	11.6	13.2	16.2	22.0	36.9	46.2	59.4	81.0	256	327	422	551

Tab. 2.5: *Computation time for different values for D and K*

normal	correlated	clustered
6.31	6.21	6.26
8.96	8.68	8.68
7.71	7.64	7.67
11.0	10.76	10.78
8.95	8.89	8.92
11.4	11.15	11.13
10.0	9.95	9.98
12.8	12.53	12.56
8.05	7.86	7.90
11.6	11.27	11.3

Tab. 2.6: *Computation time for $D = 2$ and $K = 1$ for different data sets.*

The results of these tests are listed in table 2.6 from which it is observed that the computation time is slightly but consistently lower than for the uniformly distributed gaussian. This can be explained by the second efficiency criterium that was expressed in section 2.2.3. Since the decomposition divides the data orthogonal to the maximal variance, the correlated gaussian will have less vectors lying close to the hyperplane. Also for the clustered data, the hyperplanes will pass through very sparse regions in the feature space which implies less vectors close to the plane and a lower computation time.

2.8 Conclusions

In this chapter, branch and bound search algorithms are evaluated that use a PCA-based decomposition. These algorithms were initially proposed in [52] and [19] where it was shown that they have a linear space complexity, a number of distance computations bounded by a constant term and a sublinear time complexity. The time complexity was even logarithmic for a small number of dimensions $D \leq 4$.

By using a probabilistic model that expresses the total computation cost in function of the tree traversal cost and the distance computation cost, two efficiency criteria were proposed. It was shown that for a gaussian distribution these criteria were optimized by using a separating hyperplane orthogonal to the largest variance and passing through the median vector. This provides a strong theoretical motivation for the proposed decomposition method.

Ten different algorithms were developed by combining different decomposition methods, elimination rules and traversal orders. In order to compare the efficiency of these algorithms, the optimal decomposition level was determined using a probabilistic model of the computation cost. The performance for all decomposition levels can be estimated from only two experiments, and then the most efficient level can be selected.

When comparing the results of different algorithms, it was concluded that contradictory results were obtained depending on the chosen efficiency criterium. Many results

reported in the cited references are based on the number of distance computations while the real goal one wishes to achieve is the lowest computation cost. For a low number of dimensions, the differences between the number of traversed nodes was quite small and the evaluation cost of a single node was the predominant factor. For a high number of dimensions, the efficiency of the elimination rule was more important than the node evaluation cost. The incorporation of the Fukunaga elimination rule resulted in a significant gain when the number of dimensions was large. The globally optimized traversal order introduced too much overhead and failed to realize a lower computation cost.

As stated in [52], the elimination rules become less effective when the number of dimensions increases. Fortunately, real high dimensional data sets are often highly dependent which makes that they can be represented in a lower dimensional vector space. The strength of our decomposition method is that it adapts itself automatically to the distribution of the vectors and takes into account local correlations of the data. Interestingly, lower computation costs were obtained for correlated and clustered data.

2.9 Musical Applications

Although this chapter focused on some very technical details of branch and bound search algorithms for K nearest neighbors computation, a lot of musical applications can benefit from these methods. Since the method is applicable to any non parametric classification technique it can for instance be used for instrument identification and classification [6, 67]. Another strongly upcoming field is content based music information retrieval (MIR) [28, 65, 13] and audio information retrieval (AIR) [81, 1]. A set of multidimensional search algorithms for music information was discussed in [41]. Most applications use cepstrum based features to define the similarity metric [28, 81, 24]. In the field of audio coding these techniques can contribute to the fast retrieval of the most similar dictionary element when a matching pursuit technique is applied [50, 36, 40].

CHAPTER 3

Physical Model of a Trumpet and its Constraints

In this chapter, a physical model of a trumpet is described. Although this model clearly defines the mechanical and acoustical phenomena that are perceptually relevant, additional *constraints* must be imposed on the control parameters. In contrast with the model where the tube length can be varied continuously, only seven different tube lengths can be obtained with a real instrument. By studying the physical model and its implementation, different relationships between the control parameters and signal characteristics are identified. These relationships are then used to obtain the best set of tube lengths with respect to a given tuning frequency. In addition, it is known that several notes can be obtained by using different combinations of tube length and lip frequency. However, a player knows exactly which mode and fingering must be played in order to obtain a desired note. This *prior knowledge* will be included in the estimation method described in chapter 6.

3.1 Introduction

Physical modelling consists of describing the mechanical and acoustical phenomena that take place in a musical instrument in terms of a system of equations and in solving (numerically) these equations to obtain the sound output. The sound signal is computed from a set of time-varying control parameters that correspond with the gestures of the player.

In the Analysis/Synthesis team of IRCAM, a physical model of a trumpet was developed [91, 75, 74, 90, 92, 93]. Also, a real-time implementation of the model was provided that allowed to control it with an adapted instrument-like interface (sax MIDI, Yamaha WX7) [84]. Helie [38] proposed to invert the equations on which the model

is based, resulting in an algorithm that automatically determines the lip frequency and damping factor from a synthesized sound for which the tube length and mouth pressure of the player are known. In chapter 1, a non parametric estimation technique was proposed based on nearest neighbor classification using fast nearest neighbor search algorithms. A disadvantage of this approach was that no constraints were imposed on the control parameters of the physical model resulting in control parameter sequences that were not physically acceptable. For example, when simulating a sound with vibrato the control parameters tried to simulate this by varying the tube length. This contradicts the control of a real instrument where the tube length is fixed for each note.

3.2 Physical Model of a Trumpet

3.2.1 Basic Functioning of a Trumpet

The physical model which is described below was developed by Christophe Vergez in [90]. We describe a somewhat simplified version and discuss its implementation.

A trumpet is a brass instrument consisting of a mouth piece, a resonating body with three valves and a bell. The body of the instrument consists mainly of a tube of which the length can be varied by pressing the valves. This change in tube length results in the fact that the instrument body resonates differently. The player excites the instrument by contracting his lips and augmenting the pressure in his mouth. When the force exerted by the mouth pressure is large enough, the lips are pushed open. As a result the air is released, and the force exerted by the lips make that the lips close again. This movement is repeated periodically and is nonlinear because of the collision between the lower and upper lip.

The outgoing volume flow from the lips results in an outgoing pressure wave which propagates through the instrument. At the end of the instrument, a part of this wave is reflected while a second part of the wave is transmitted into the acoustic environment and results in the perceived sound. This results in an incoming that this system is nonlinear because of the lip collision and has delayed feedback because of the reflected wave.

3.2.2 The Lips: a Non-Linear Mass-Spring-Damper System

It is known that mainly the upper lip of the trumpet player oscillates while the lower lip moves very little. The upper lip is modelled by a parallelepipedic mass attached to a damped spring. The position of the mass is indicated by $x(t)$ while m, r and k denote the mass, damping and stiffness of the spring. This system is depicted in Figure 3.1. The equilibrium between all forces that are exerted on this mass, when the lips are opened ($x(t) > 0$), results in the following equation

$$m\ddot{x}(t) + r\dot{x}(t) + kx(t) = A\cos(\alpha)(P_M(t) - p(t)) \tag{3.1}$$

The left hand side of the equation is the differential equation that describes the dynamic behavior of the mass, while the right hand side expresses the external forces. These

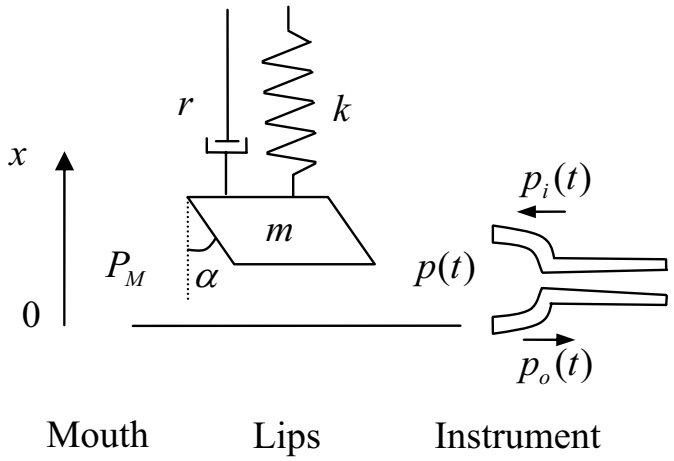


Fig. 3.1: *Nonlinear mass-spring-damper system.*

external forces are the result of the pressures that are exerted on the left a and right surfaces of the mass. This surface is denoted A from which follows that $A\cos(\alpha)p(t)$ denotes the force component in the x direction. The pressure that is exerted by the player $P_M(t)$ pushes the lips open, i.e. in the direction of the x axis. The pressure on the side of the instrument $p(t)$ results in a movement of the lips in the inverse direction.

The behavior of this system changes when the lips collide which is the case when $x(t) < 0$. The closure of the lips is modelled by changing the spring characteristics. The stiffness is increased with a factor 4 and the damping with a factor 5 [90].

$$m\ddot{x}(t) + 5r\dot{x}(t) + 4kx(t) = A\cos(\alpha)(p_s(t) - p(t)) \tag{3.2}$$

This sudden change introduces a nonlinearity to the oscillation.

3.2.3 Wave Propagation in the Body of the Instrument

The pressure in the body of the instrument at a position z at a time t is denoted $p(z,t)$. When the propagation in the instrument is linear and plane, one can write $p(z,t)$ as the sum of an incoming wave $p_i(t)$ and an outgoing wave $p_o(t)$ yielding

$$p(z,t) = p_o(t - \frac{z}{c_0}) + p_i(t + \frac{z}{c_0}) \tag{3.3}$$

where c_0 denotes the propagation speed of sound in air. For the volume flow one obtains

$$u(z,t) = u_o(t - \frac{z}{c_0}) + u_i(t + \frac{z}{c_0}) \tag{3.4}$$

For a progressive plane wave, the acoustic impedance $Z_c(z)$ expresses the ratio of the pressure over the volume flow, at a given point z

$$Z_c(z) = \frac{p_o(t)}{u_o(t)} = -\frac{p_i(t)}{u_i(t)} \tag{3.5}$$

yielding

$$p(z,t) = Z_c(z)(u_o(t - \frac{z}{c_0}) - u_i(t + \frac{z}{c_0})) \tag{3.6}$$

It is known that

$$Z_c(z) = \frac{\rho_0 c_0}{A(z)} \tag{3.7}$$

where ρ_0 denotes the density of the air, and $A(z)$ the surface at position z.

3.2.4 The Linear Response of the Body

When the body of the instrument is considered to be a linear resonator, the incoming pressure wave can be written as the outgoing pressure wave convoluted with the impulse response of the instruments body.

$$p_i(t) = (h_\lambda * p_o)(t) \tag{3.8}$$

This impulse response h_λ was measured on a real instrument [10] and was then simplified for computational reasons [90]. When looking at this impulse response which is depicted in Figure 3.2, the following parts of the reflection function h_λ are observed

- h_1, the direct response at the mouthpiece
- a delay of λ zeros, corresponding with the cylindrical part of the tube where there is almost no reflection
- h_2, the reflection after travelling back and forth the instrument body

An example of the total reflection function h_λ is shown in figure 3.2. By changing the number of zeros between these two reflections, different tube lengths are simulated which imitates the effect of the valves of the trumpet [91].

In the implementation that was provided in [84], the number of zeros λ was is computed from the tube length control parameter P_T by

$$\lambda = \lfloor \frac{F_s}{2P_T} \rfloor \tag{3.9}$$

The interval between the two maxima in the reflection function can be computed approximatively by adding the number of samples from the peaks to the zeros and is denoted

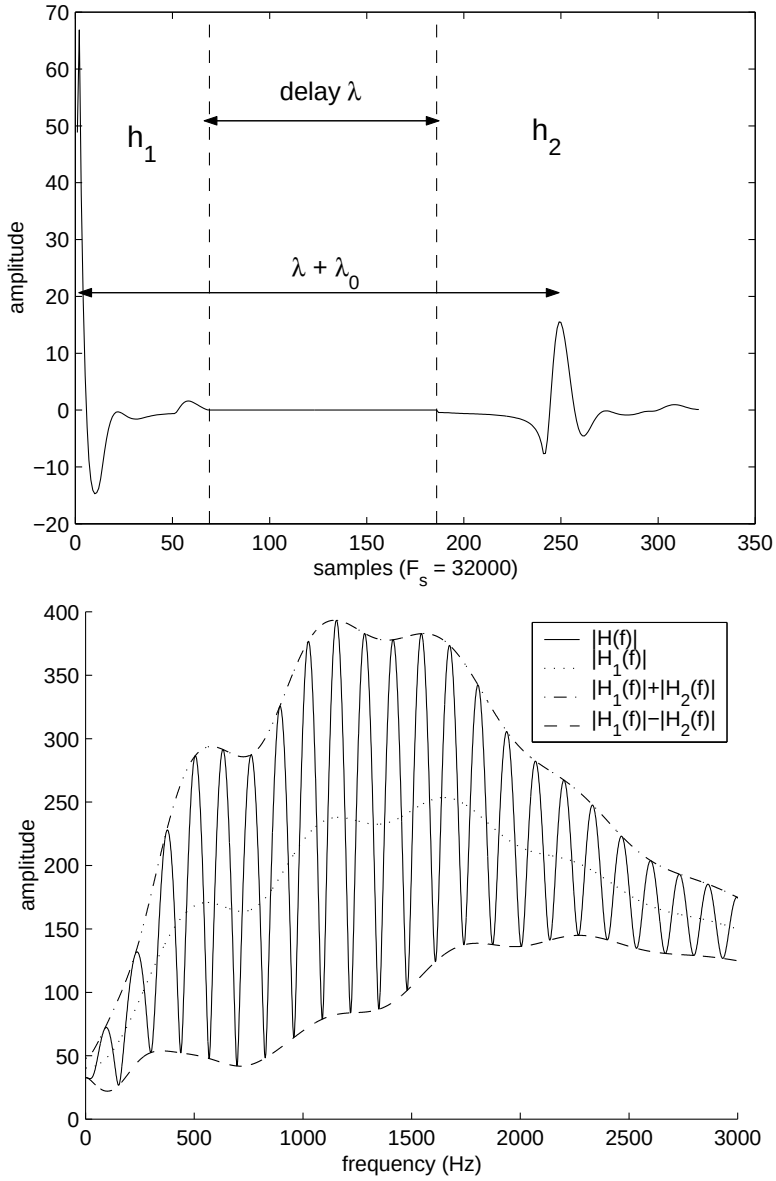


Fig. 3.2: *Reflection function of the instrument body in the time and frequency domain.*

λ_0. This implies that the total time for the wave to run back and forth the instrument body is given by

$$\tau = \frac{\lambda + \lambda_0}{F_s} \tag{3.10}$$

Therefore the resonance frequencies of the tube tube will be multiples of

$$f_\tau = \frac{F_s}{\lambda + \lambda_0} \tag{3.11}$$

which are depicted in figure 3.2. When the total reflection function is expressed as

$$h_{\lambda,n} = h_{1,n} + h_{2,n-\tau} \tag{3.12}$$

its fourier transform yields

$$H_\lambda(f) = H_1(f) + H_2(f)e^{-2\pi i f\tau} \tag{3.13}$$

Maxima and minima of $|H_\lambda(f)|$ are obtained when $H_1(f)$ and $H_2(f)$ are in phase or antiphase respectively, implying that

$$|H_1(f)| - |H_2(f)| \leq |H_\lambda(f)| \leq |H_1(f)| + |H_2(f)| \tag{3.14}$$

This shows that the bounds of $|H_\lambda(f)|$ are independent of λ. Therefore, the expression $|H_1(f)| + |H_2(f)|$ denotes the *spectral envelope* of $|H_\lambda(f)|$ as shown figure 3.2. Their phase difference is mainly due to the modulator term $e^{-2\pi i f\tau}$ which determines the position of the resonances which are approximately equally spaced with an interval f_τ.

3.2.5 Nonlinear Acoustic Coupling

The coupling between the lips and the instrument body is realized by applying the stationary Bernouilli theorem. This theorem expresses the relationship between the pressure and flow velocity inside the mouth (P_M and v_M) and leaving the lips ($p(t)$ and $v(t)$) resulting in

$$P_M + \frac{1}{2}\rho v_M^2 = p(t) + \frac{1}{2}\rho v^2(t) \tag{3.15}$$

The mouth is considered a reservoir with a constant pressure P_M in which the flow velocity can be neglected, meaning $v_M = 0$. Under the hypothesis of non compressibility, the volume flow is expressed by

$$u(t) = v(t)A(t) \tag{3.16}$$

When the lip opening is assumed to be rectangular, the surface can be written as

$$A(t) = x(t)l \tag{3.17}$$

yielding

$$Z_c(t) = \frac{\rho_0 c_0}{x(t)l} \tag{3.18}$$

where l is the width of the opening. In the case that $P_M > p(t)$, the volume flow can now be written using

$$u(t) = l\sqrt{\frac{2}{\rho}}x(t)\sqrt{(P_M - p(t))} \tag{3.19}$$

In the case that $P_M < p(t)$ a volume flow is realized going inside the mouth yielding

$$u(t) = -l\sqrt{\frac{2}{\rho}}x(t)\sqrt{(-P_M + p(t))} \tag{3.20}$$

These two equations can be written as a single equation

$$u(t) = l\sqrt{\frac{2}{\rho}}x(t)sgn(p_M - p(t))\sqrt{|P_M - p(t)|} \tag{3.21}$$

where $sgn(P_M - p(t))$ returns 1 when $P_M - p(t)$ is positive and -1 in the opposite case. Denoting the pressure and volume flow at the lips, $z = 0$, by $p(0,t)$ and $u(0,t)$ results in

$$\begin{cases} u(0,t) = & l\sqrt{\frac{2}{\rho}}x(t)\sqrt{(P_M - p(0,t))} \\ u(0,t) = & \frac{1}{Z_c}(p(0,t) - 2p_i(0,t)) \end{cases} \tag{3.22}$$

follows that

$$\begin{aligned} l\sqrt{\frac{2}{\rho}}x(t)\sqrt{(P_M - p(0,t))} &= \frac{1}{Z_c(t)}(p(0,t) - 2p_i(0,t)) \\ Z_c l\sqrt{\frac{2}{\rho}}x(t)\sqrt{(P_M - p(0,t))} &= p(0,t) - 2p_i(0,t) \end{aligned}$$

when using a constant $C = Z_c l\sqrt{\frac{2}{q}}$, one obtains

$$C^2x(t)^2(P_M - p(0,t)) = p^2(0,t) - 4p_i^2(0,t)p(0,t) + 4p_i^2(0,t) \tag{3.23}$$

or

$$p^2(0,t) + (C^2x(t)^2 - 4p_i(0,t))p(0,t) + 4p_i^2(0,t) - C^2x(t)^2P_M = 0 \tag{3.24}$$

This equation can be solved $p(0,t)$, yielding

$$p(0,t) = 2p_i(0,t) - \frac{1}{2}Cx(t)\left(Cx(t) - \sqrt{(C^2x(t)^2) + 4(P_M - 2p_i(t,0))}\right) \tag{3.25}$$

The current pressure at the lips $p(t)$ is now expressed in function of the incoming pressure p_i, the mouth pressure P_M and the lip position $x(t)$.

3.2.6 A Discrete Model of a Damped Oscillator

The differential equation for the lips, Eq. (3.1) can be implemented in a discrete way by using an Euler scheme or a recursive filter.

Euler Scheme

When discretizing the continuous equation

$$m\ddot{x}(t) + r\dot{x}(t) + kx(t) = F \tag{3.26}$$

using a right hand Euler scheme, one obtains

$$m\frac{x_n - 2x_{n-1} + x_{n-2}}{(\Delta t)^2} + r\frac{x_n - x_{n-1}}{\Delta t} + kx_n = F_n \tag{3.27}$$

where Δt denotes the sampling interval. The value of x_n can be derived in function of previous values and the external forces by

$$x_n = \zeta_1 x_{n-1} + \zeta_2 x_{n-2} + \zeta_3 F_n \tag{3.28}$$

with

$$\begin{aligned} \zeta_1 &= \left(2 + \frac{r}{m}\Delta t\right)\zeta_2 \\ \zeta_2 &= \frac{1}{\frac{k}{m}\Delta t^2 + \frac{r}{m}\Delta t + 1} \\ \zeta_3 &= \frac{(\Delta t)^2}{m}\zeta_2 \end{aligned}$$

in which the physical characteristics of the oscillator in free movement appear, namely

- the eigen frequency $\omega = \sqrt{\frac{k}{m}}$
- the damping factor $\rho = \frac{r}{2m}$

An inconvenience of the Euler scheme is that when an oscillator is modelled which is not damped, meaning that $r = 0$, the discrete oscillator becomes

$$x_n = \frac{2}{1 + \frac{k}{m}(\Delta t)^2}x_{n-1} + \frac{-1}{1 + \frac{k}{m}(\Delta t)^2}x_{n-2} \tag{3.29}$$

For an oscillator which is not damped, the coefficient for x_{n-2} should result in -1. Therefore the discrete case of an oscillator that is not damped results in a damped oscillator.

Recursive Filter

The behavior of a mass-spring-damper system on which no external force is exerted can also be described by an recursive filter with two complex conjugate poles being p and p^*. The frequency response $H(z)$ is then given by

$$H(z) = \frac{1}{(1 - pz^{-1})(1 - p^* z^{-1})} \tag{3.30}$$

When these poles are written as $p = \rho \exp(2\pi i\omega)$, the inverse z-transform yields

$$x_n = 2\rho \cos(\omega) x_{n-1} - \rho^2 x_{n-2} \tag{3.31}$$

By extension of Eq. (3.28), the influence of the external force is included using

$$x_n = \zeta_1 x_{n-1} + \zeta_2 x_{n-2} + \zeta_3 F_n \tag{3.32}$$

with

$$\begin{aligned} \zeta_1 &= 2\rho \cos(\omega) \\ \zeta_2 &= -\rho^2 \\ \zeta_3 &= -\zeta_2 \frac{(\Delta t)^2}{m} \end{aligned}$$

3.2.7 The Complete Trumpet Synthesizer

When taking into account all mechanical and acoustical properties of the instrument and their couplings the following trumpet synthesis algorithm is obtained.

$\lambda = \lfloor \frac{F_s}{2P_T} \rfloor$ `// Delay computation`
$p_{i,n} = \sum_{k=1}^{K} p_{o,n-k} h_{\lambda,k}$ `// Body Resonance`
`if` $x_n > 0$ `// Lips opened`
    $p_n = 2p_{i,n} - \frac{1}{2} C x_n (C x_n - \sqrt{(C x_n)^2 + 4|P_M - 2p_{i,n}|})$
    $u_n = (p_n - 2p_{i,n})/Z_c$
    $x_{n+1} = 2P_D \cos(2\pi P_L/F_s) x_n - P_D^2 x_{n-1} + \frac{1}{mF_s^2}(P_M - p_n)$
`else // Lips closed`
    $p_n = 2p_{i,n}$
    $u_n = 0$
    $x_{n+1} = 2P_D \cos(2\pi(2P_L)/F_s) x_n - P_D^2 x_{n-1} + \frac{1}{mF_s^2}(P_M - p_n)$
`end`
$p_{o,n} = \frac{1}{2}(p_n + Z_c u_n)$

Algorithm 1: Trumpet synthesizer

In the first two lines, the number of zeros λ is computed which is used to simulate the desired tube length and the incoming pressure wave is computed by convolution with the outgoing wave. According to the lip position which can be opened or closed, the new values of the pressure p_n, the volume flow u_n and the lip position x_n are computed. This results finally in the outgoing pressure wave $p_{o,n}$.

To sum up, the control parameters are

- the mouth pressure P_M
- the lip frequency P_L
- the lip damping P_D
- the tube length P_T

The damping factor was generally kept fixed at a value $P_D = 0.999$.

3.3 Physical Constraints and Prior Knowledge

As stated in the introduction, the constraint that must be imposed is that a constant tube length must be used for each note. In contrast with the physical model where the length can be varied continuously, a real trumpet can only obtain seven different tube lengths. With the three valves, eight combinations can be obtained of which two result in the same tube length. In table 3.1 each column corresponds with a tube length and each row with a mode, resulting in a given note. As can be seen from the table, some notes can be obtained with different tube lengths. For example G4 can be obtained by exciting the sixth mode of tube length 1, the seventh mode of tube length 4 or the eighth mode of tube length 6.

When controlling the physical model of the trumpet, the following constraint must be taken into account:

> **Constraint 1:** "When a trumpet player interprets a score he chooses a correct finger position and excites the correct mode of the tube in order to obtain the desired note."

In other words, the player uses a mapping from the note he wishes to play to a mode and tube length couple. For simplicity, we will assume that the same combination is always used. In reality, other combinations can sometimes be preferred but this is considered beyond the scope of this article.

A second factor that influences the obtained fundamental frequency is the tuning of the instrument. The tuning valve adjusts all the tube lengths in order to correspond with a given reference frequency f_{ref}. Typically A3 corresponds with 440 Hz. This leads to a second constraint:

> **Constraint 2:** "Given a reference frequency, a set of seven tube lengths must be determined for the control of an entire trumpet performance"

	tube length						
mode N	1	2	3	4	5	6	7
1	C2	B1	B♭1	A1	A♭1	G1	F♯1
2	C3	B2	B♭2	A2	A♭2	G1	F♯2
3	G3	F♯3	F3	E3	E♭3	D3	C♯3
4	C4	B3	B♭3	A3	A♭3	G3	F♯3
5	E4	E♭4	D4	C♯4	C4	B4	B♭4
6	G4	F♯4	F4	E4	E♭4	D4	C♯4
7	B♭4	A4	A♭4	G4	F♯4	F4	E4
8	C5	B4	B♭4	A4	A♭4	G4	F♯4

Tab. 3.1: *Notes obtained by exciting different modes of a given tube length.*

3.4 Excitation of the Modes

3.4.1 Resonance Phenomena of the Physical Model

When the model produces a stable periodic sound, each period consists of an interval where the lips are opened and an interval where the lips are closed. We derive the pressure value p_n for large opening of the lips, $x_n \gg 0$. When expressing the square root as a Taylor expansion and taking the limit for x_n going to infinity one obtains

$$\begin{aligned}
p_n &= 2p_{i,n} - \frac{1}{2}Cx_n\left(Cx_n - \sqrt{(Cx_n)^2 + 4(P_M - 2p_{i,n})}\right) \\
&= 2p_{i,n} - \frac{1}{2}(Cx_n)^2\left(1 - \sqrt{1 + \frac{4|P_M - 2p_{i,n}|}{(Cx_n)^2}}\right) \\
&= 2p_{i,n} - \frac{1}{2}(Cx_n)^2\left(1 - \left[1 + \frac{1}{2}\frac{4(P_M - 2p_{i,n})}{(Cx_n)^2} - \frac{3}{2^3 2!}\left(\frac{4(P_M - 2p_{i,n})}{(Cx_n)^2}\right)^2 + \ldots\right]\right) \\
&= 2p_{i,n} + (P_M - 2p_{i,n}) \\
&= P_M
\end{aligned} \tag{3.33}$$

This means that p_n approaches P_M when x_n is large. An example is given in figure 3.3. In this case the value of the mouth pressure P_M is 5000 Pa which is exactly the value that is obtained for p_n when the lips are opened ($x \gg 0$). For a large opening, the state variables take the values

$$\begin{aligned}
p_n &= P_M \\
u_n &= (P_M - 2p_i)/Z_c \\
p_{o,n} &= P_M - p_{i,n} \\
P_M - p_n &= 0
\end{aligned} \tag{3.34}$$

Since the term $\frac{1}{mF_s^2}(P_M - p_n)$ expresses the external force that is exerted on the lips, it follows that the lips oscillate freely when they are largely opened. When $x < 0$, which

means when the lips are closed, we obtain

$$\begin{aligned} p_n &= 2p_{i,n} \\ u_n &= 0 \\ p_{o,n} &= p_{i,n} \\ P_M - p_n &= P_M - 2p_{i,n} \end{aligned} \tag{3.35}$$

indicating that now an external force is exerted on the lips is expressed by

$$\frac{1}{mF_s^2}(P_M - 2p_{i,n}) \tag{3.36}$$

For small positive values of x_n, transition values for these state values are obtained. From this computation, it can be concluded that these state variables have the same period as the lip period. Therefore, when the fundamental frequency is measured from the sound signal produced by the physical model (this is the high pass filtered outgoing pressure $p_{0,n}$), the periodicity of all the state variables is known. A second conclusion that is drawn, is that the lips are excited by an external force, essentially when they are closed. Since the strength of excitation force is dependent on $P_M - 2p_{i,n}$, the conditions are examined for which the excitation is the strongest, meaning that the resonance is maximal. For a fixed value for P_M this will be obtained for a negative value of $p_{i,n}$ with a maximal absolute value. Since $p_{i,n}$ is calculated from a convolution of the outgoing wave $p_{o,n}$ with the reflection function of the body $h_{\lambda,n}$ the maximal strength is obtained when the period of p_o is a multiple of the resonance frequency f_τ of the tube. In figure 3.4 an example is shown for the fourth mode of a fixed tube length. Both peaks of $h_{\lambda,n}$ coincide with negative values of p_o which results in a negative value for p_i with a maximal absolute value. In addition, the fact that four periods of p_o correspond with twice the tube length (the reflection function is measured at the mouthpiece) implies that the fourth mode is excited. Thus, for a fixed tube length, the trumpet model will resonate the strongest when the produced sound is a multiple of the first tube mode, f_τ. This implies that the amplitude of x_n is the highest when $f_0 = Nf_\tau$, where N is the mode index.

3.4.2 Relationship with Lip Frequency P_L

It would be interesting to know which lip frequency is used in order to obtain this optimal resonance. This is difficult since the lip frequency does not correspond directly with the produced fundamental frequency. When the lips are opened, they oscillate freely, implying that the time interval that they are opened is $\frac{1}{2P_L}$. When the lips are closed, P_L is doubled, resulting in an interval of $\frac{1}{4P_L}$ when no external force is exerted. As a result, the total period has a length of of $\frac{3}{4P_L}$ and a corresponding frequency of $\frac{4P_L}{3}$. We will examine whether this is still a good approximation for the fundamental frequency even when the external force is exerted. At the bottom of figure 3.3 the lip position is shown in function of time.

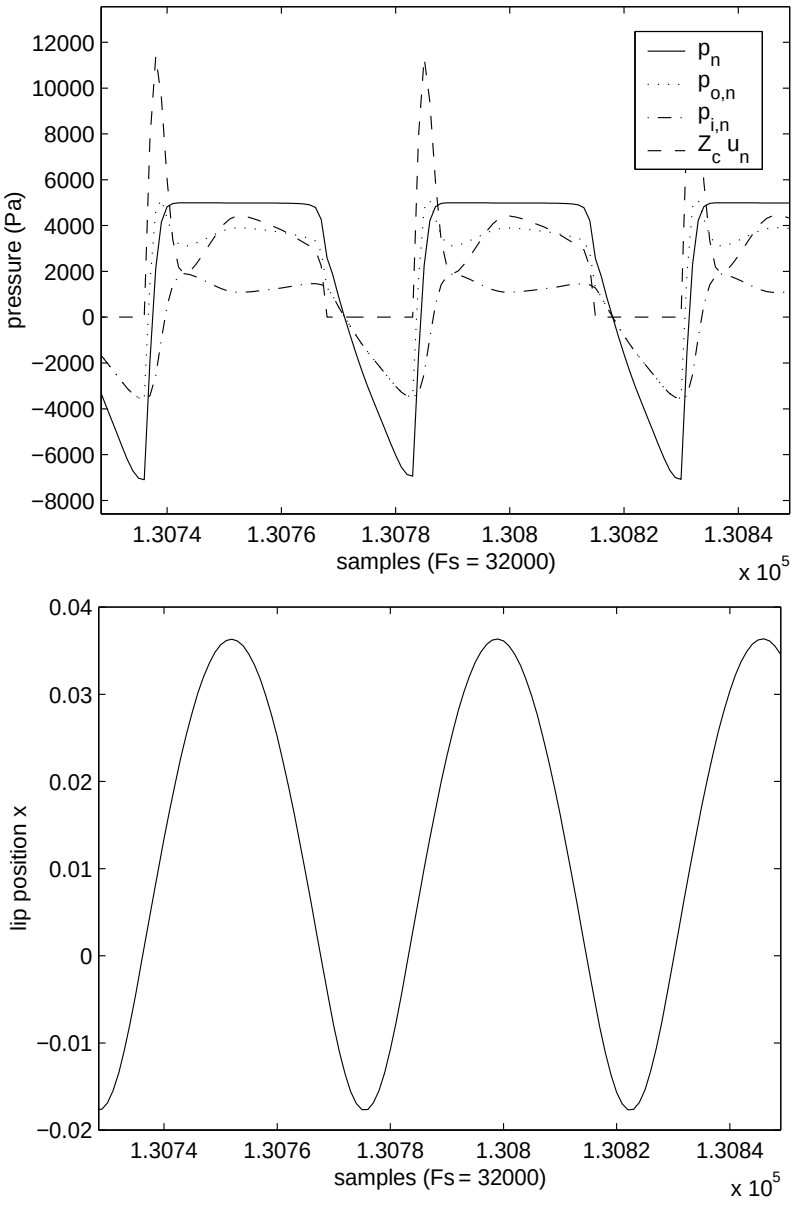


Fig. 3.3: *Periodicity of lip position and state variables.*

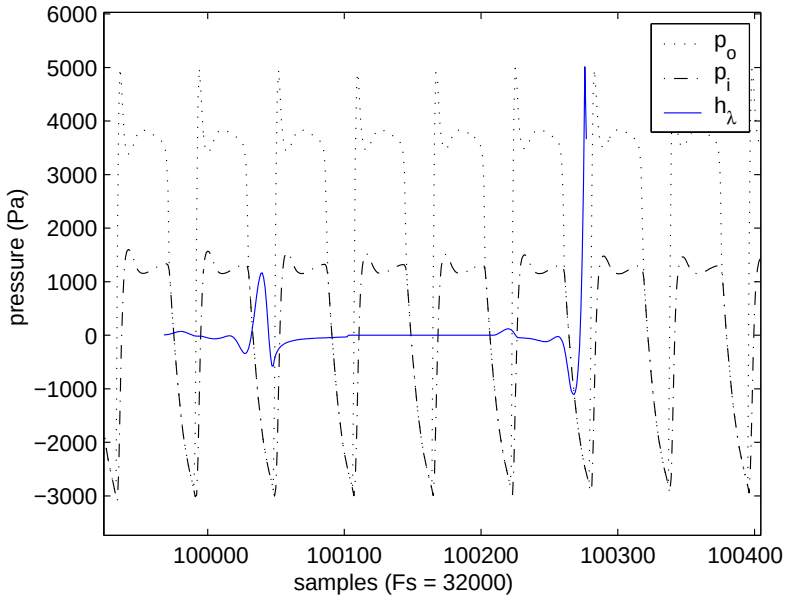


Fig. 3.4: *Pressure waves and reflection function.*

3.5 Tube Length Determination

3.5.1 Validation Experiment

The computations in the previous sections can be validated by executing the following experiment. By varying the lip frequency from its lowest to its highest value, the maximal resonances of the lip position x_n can be observed. If the fundamental frequencies at these maximal resonances are multiples of f_τ, and the corresponding lip frequency is $\frac{3}{4}P_L$, then the previous reasoning is validated.

The experiment was performed for values of P_T being 150 and 70. F_s and λ_0 had the values 32000 and 128 respectively resulting in the f_τ-values of 136.7 and 89.9. From the results listed in table 3.2, the following conclusions can be drawn:

- For the high modes it is observed that f_τ is very close to f_0/N, meaning that the fundamental frequencies for the maximal resonances are very close to multiples of f_τ.
- The ratio $\frac{P_L}{f_0}$ seems to be a constant very close to $\frac{3}{4}$
- These approximations are less accurate for the lower modes. In this case, the actual fundamental frequency is higher then the estimated value.

In the case of maximal resonance, the following relationships can be deduced for the physical model.

$$f_0 = N f_\tau \tag{3.37}$$

P_T	N	f_0	f_0/N	P_L	P_L/f_0
70	10	899	89.9	673.7	0.749
	9	809	89.8	605.4	0.748
	8	720	90.0	538.5	0.748
	7	630	90.1	471.3	0.748
	6	537	91.2	404	0.752
	5	463	92.6	344	0.743
	4	364	91.0	271	0.745
	3	276	92.0	205	0.743
	2	185	92.5	137	0.741
150	9	1231	136.7	922	0.749
	8	1094	136.7	819	0.749
	7	955	136.4	715	0.749
	6	818	136.3	614	0.751
	5	633	136.3	511	0.751
	4	551	137.7	412	0.748
	3	414	138	308	0.744
	2	277	138.5	207	0.747

Tab. 3.2: *Observations for different tube lengths.*

$$f_0 = \frac{4}{3} P_L \tag{3.38}$$

This corresponds with the first constraint that was expressed in section 2. When a player wishes to play a given note, a valve position must be chosen corresponding with a resonance frequency f_τ so that the desired fundamental frequency is a multiple of f_τ. In order to excite the correct mode N, the correct value of P_L must be used according to equation (3.38). This corresponds with the control of a real instrument and yields therefore an extra validation of the physical model.

3.5.2 Instrument Tuning

As indicated in table 3.1, seven different tube lengths are used in order to obtain all notes. Therefore, we wish to determine seven values for P_T that correspond with these tube lengths. The frequency f_0 of a note is calculated from a note index I and a reference frequency f_{ref} in the following manner

$$f_0 = f_{ref} 2^{\frac{I}{12}} \tag{3.39}$$

Taking 440 Hz as the reference frequency, meaning that $I = 0$ corresponds with the medium A, the fundamental frequencies of all notes are calculated. Knowing which notes are played using the fourth mode ($N = 4$), the values of f_τ and P_T can be deduced using equations (3.9) (neglecting the floor operator) and (3.11) as shown in table 3.3. This satisfies the second constraint that was imposed.

note	I	f_0	$f_\tau = \frac{f_0}{4}$	λ	P_T
C4	3	523.3	130.8	116.6	137.2
B3	2	493.9	123.5	131.2	122.0
B♭3	1	466.2	116.5	146.6	109.2
A3	0	440	110	162.9	98.2
A♭3	-1	415.3	103.8	180.2	88.8
G3	-2	392.0	98.0	198.5	80.6
F♯3	-3	370	92.5	218.0	73.4

Tab. 3.3: *Determination of P_T.*

$\hat{\lambda}$	f_τ	f_0	I	$P_{T,min}$	$P_{T,max}$
117	130.6	522.4	2.97	135.6	136.7
131	123.6	494.2	2.01	121.3	122.1
147	116.4	465.5	0.97	108.2	108.8
163	110.0	439.9	-0.01	97.6	98.1
180	103.9	415.6	-0.99	88.4	88.9
199	97.6	391.4	-2.02	80.1	80.5
218	92.5	370	-3.00	73.1	73.3

Tab. 3.4: *Determination of P_T for $\hat{\lambda}$.*

Furthermore, it must be taken into account that the value of λ is an integer value. Therefore, the integer value $\hat{\lambda}$ value closest to λ is used in order to recalculate the obtained fundamental frequencies to determine wether this is admissible. This results in an interval $P_L \in [P_{L,min}, P_{L,max}]$ for which all values result in the same value of $\hat{\lambda}$. The results are shown in table 3.4. The recalculation of I shows that the floor function for the computation of λ introduces a maximal deviation of three percent of a half tone.

Using appropriate combinations of tube length and lip frequency, as given in table 3.1, an ascending chromatic scale was synthesized. Figure 3.5 shows the note index computed from the fundamental frequency, according to equation (3.39). This figure confirms the previously drawn conclusion that the estimated f_0 is less accurate for lower modes.

3.6 Further work and conclusions

This paper describes the physical constraints that must be imposed for the control of a physical model of a trumpet. A real trumpet only uses seven tube lengths while the tube length parameter P_T of the physical model is continuous. Although the physical model used in this article is quite simplified, no significant changes in the synthesized sounds were observed. By a detailed study of the implementation of this physical model

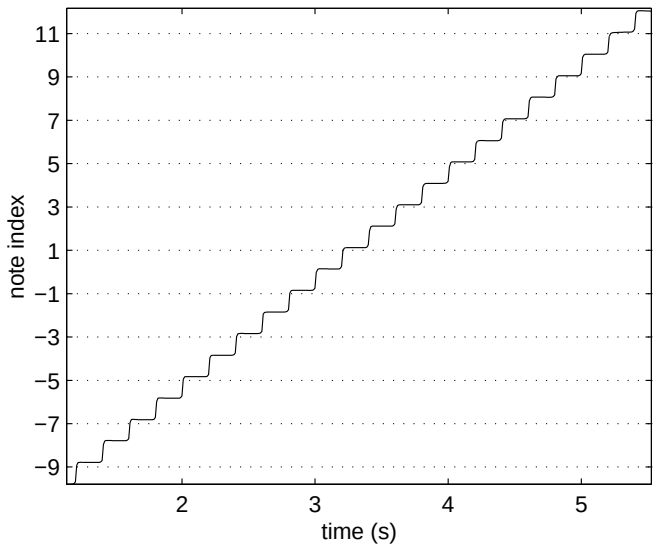


Fig. 3.5: *Note index of a chromatic scale played by the physical model.*

some very simple and approximative relationships between the fundamental frequency and the control parameters were identified. These relationships were then used in order to determine a set of seven tube lengths with respect to a given tuning frequency f_{ref}.

In addition, a number of parameters had to be determined manually. Due to the simplification of the reflection function h_λ additional filters were used to amplify the lower modes [91]. The resonance frequency of these filters was at the second and third mode of the tube and the amplitude of the filter was adapted manually in order to obtain a strength comparable with higher modes.

CHAPTER 4

Discrete Cepstrum Coefficients as Perceptual Features

4.1 Chapter Overview

Cepstrum coefficients are widely used as features for both speech and music. For the representation of the spectral envelope of quasi-periodic sounds, the discrete cepstrum was developed, which is computed from sinusoidal peaks in the short time spectrum. In this chapter, an attempt is made to combine elements from the pattern recognition, psychoacoustics and signal processing field to achieve a distance metric that expresses the perceptual similarity between the spectral envelopes of two signals.

After discussing the applications in which cepstrum coefficients play an important role (4.2), the definition of the discrete cepstrum and its computation is discussed (4.3). It is shown that the estimation problem is ill-posed and that this can result in *overfitting*. In order to take into account the frequency resolution of the human auditory system, the envelope is defined in function of the Mel frequency instead of the linear frequency (4.4). Current techniques that avoid overfitting are presented, including a new method which was named *posterior warping*. This technique computes the Mel frequency coefficients directly from the linear scale coefficients. In addition, the coefficients were observed to be very sensitive to noise which is amplified enormously by the log function. This problem was avoided succesfully using a threshold.

4.2 Introduction

In its elementary form, the real cepstrum of a signal is defined as the inverse fourier transform of the log magnitude spectrum. In practical recognition applications however, they are rarely used as features in this form. In the case of speech recognition for example,

a filter bank is applied of which the center frequency of each bank is scaled according to the Mel scale. This scale takes into account the frequency resolution properties of the human ear. The inverse fourier transform of the log output of this filter bank yields the *Mel Frequency Cepstrum Coefficients (MFCC)*. Various other cepstrum like coefficients have been proposed and it is believed that further improvement in the front-end of a speech recognition system, i.e. the feature extraction, can be achieved [37, 57].

Also in the music domain, cepstrum coefficients have been extensively used in numerous applications such as the retrieval of similar audio tracks [1], instrument identification [6], content based audio retrieval [28, 81], synthesis [78], and they are currently investigated for automated estimation of control parameters for musical synthesis algorithms [23, 26].

In this work, the characterization of the *spectral envelope* of a nearly periodic sound is studied. The spectral envelope is a function of the frequency that matches the amplitudes of the individual partials in the spectrum. This captures an important aspect of the timbre since it is generally accepted that the relative strength of the amplitudes of the partials allows to distinguish musical instruments and spoken language vowels. However, a strong abstraction is still made and not all perceptually relevant features of the timbre are captured. For example, the noise component is not taken into account and the roughness is often diminished when the analysis window is taken too large. Furthermore, the estimation of the partials is often not accurate at transients.

Different representations of the spectral envelopes have been proposed such as *linear prediction coefficients (LPC)*, the cepstrum and the *discrete cepstrum*. The discrete cepstrum was originally proposed by Gallas and Rodet [31, 32] and later, a regularized version was developed by Cappé and Oudot [7, 9]. In the work of Schwarz [78], different spectral envelope representations were studied and compared. There, it was shown that the discrete cepstrum is more suitable for the representation of nearly periodic sounds than LPC or the cepstrum since for these last two methods the envelopes follow the individual peaks in the spectrum. This problem is illustrated in Figure 4.1.

4.3 Discrete Cepstrum

4.3.1 Definition and Computation

P discrete cepstrum coefficients c_p, with $p = 0, \ldots, P-1$ define a magnitude envelope $|H(\omega)|$ of the form

$$|H(\omega)| = \exp\left(c_0 + 2\sum_{p=1}^{P-1} c_p \cos(p\omega)\right) \tag{4.1}$$

$$c_p = \frac{1}{2\pi}\int_{-\pi}^{\pi} \log(|H(\omega)|)e^{i\omega p}d\omega \tag{4.2}$$

Since the inverse fourier transform of the log amplitude yields again the coefficients c_p, this definition corresponds with the classic cepstrum definition. Contrary to the classic

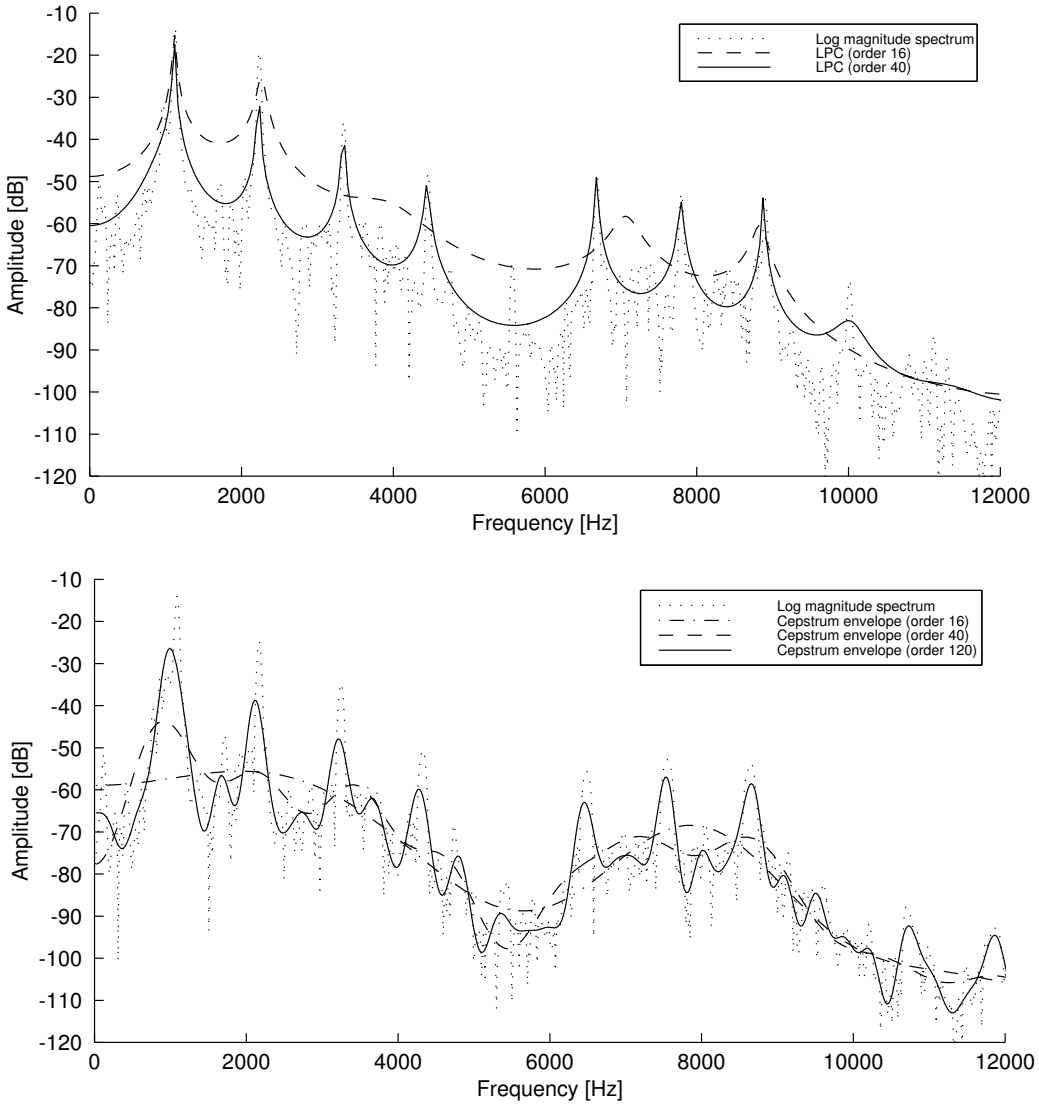


Fig. 4.1: *Spectral Envelopes using Linear Prediction Coefficients and the cepstrum (courtesy of D. Schwarz)*

cepstrum which is computed directly from the spectrum, the discrete coefficients are matched with the amplitudes of sinusoidal components. These amplitudes are computed by applying a sinusoidal analysis [71, 17, 80]. A spectrum of this form can be described by a set of partials at frequencies ω_k with amplitudes $\hat{X}_k$ $(k = 1, \ldots, K)$. This is written as

$$X(\omega) = \sum_{k=1}^{K} \hat{X}_k \delta(\omega - \omega_k) \tag{4.3}$$

where $\delta(\omega)$ denotes the Dirac delta distribution. The estimation of the coefficients c_p is realized by minimizing the square log difference between the amplitude envelopes $|H(\omega)|$ and $|X(\omega)|$. This equation defines $|X(\omega)|$ only at the peak frequencies ω_k from which the following square error function $\chi(\mathbf{c})$ can be derived in function of the cepstrum coefficients $\mathbf{c}$.

$$\begin{aligned}
\chi(\mathbf{c}) &= \sum_{k=1}^{K} \Big(\log(|H(\omega_k)|) - \log(\hat{X}_k) \Big)^2 \\
&= \sum_{k=1}^{K} \left(c_0 + 2 \sum_{p=1}^{P-1} c_p \cos(p\omega_k) - \log(\hat{X}_k) \right)^2 \\
&= \sum_{k=1}^{K} \left(\sum_{p=0}^{P-1} (2 - \delta_{p0}) c_p \cos(p\omega_k) - \log(\hat{X}_k) \right)^2
\end{aligned} \tag{4.4}$$

this error function is minimized by putting all partial derivatives $\frac{\chi(\mathbf{c})}{c_q}$ for each q to zero resulting in

$$\sum_{k=1}^{K} \left(\sum_{p=1}^{P-1} (2 - \delta_{p0}) c_p \cos(p\omega_k) - \log(\hat{X}_k) \right) \cos(q\omega_k) = 0 \tag{4.5}$$

which can be written in the following matrix form

$$\mathbf{Bc} = \mathbf{b} \tag{4.6}$$

with

$$\begin{aligned}
B_{q,p} &= \sum_{k=0}^{K-1} (2 - \delta_{p0}) \cos(p\omega_k) \cos(q\omega_k) \\
b_q &= \sum_{k=0}^{K-1} \log(\hat{X}_k) \cos(q\omega_k)
\end{aligned} \tag{4.7}$$

A different matrix notation which is frequently used is the following

$$\begin{aligned}
\chi(\mathbf{c}) &= \sum_{k=1}^{K} \left(\sum_{p=0}^{P-1} (2 - \delta_{p0}) c_p \cos(p\omega_k) - \log(\hat{X}_k) \right)^2 \\
&= (\mathbf{Mc} - \mathbf{a})^T (\mathbf{Mc} - \mathbf{a})
\end{aligned} \tag{4.8}$$

with

$$\mathbf{M} = \begin{bmatrix} 1 & 2\cos(\omega_1) & \dots & 2\cos(P\omega_1) \\ \vdots & & & \vdots \\ 1 & 2\cos(\omega_K) & \dots & 2\cos(P\omega_K) \end{bmatrix}$$

$$\mathbf{c} = \begin{bmatrix} c_0 \\ \vdots \\ c_P \end{bmatrix} \tag{4.9}$$

$$\mathbf{a} = \begin{bmatrix} \log(\hat{X}_1) \\ \vdots \\ \log(\hat{X}_K) \end{bmatrix} \tag{4.10}$$

In this case, the error minimization yields,

$$\begin{aligned} & \tfrac{\partial \chi(\mathbf{c})}{\partial \mathbf{c}} = 0 \\ \Rightarrow \quad & \mathbf{M}^T\mathbf{M}\mathbf{c} - \mathbf{M}^T\mathbf{a} = 0 \\ \Rightarrow \quad & \mathbf{c} = (\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T\mathbf{a} \end{aligned} \tag{4.11}$$

4.3.2 Overfitting

Since the cepstrum coefficients are computed from a linear set of equations, the computation of P coefficients requires minimum an equal number of detected peaks. If not, a singular matrix will be obtained for $\mathbf{A}$ in the set of equations. Although the error function defined in Eq. (4.11) can be minimized exactly to zero when the number of coefficients equals the number of peaks, it is not desired to do so. As can be seen from Fig. 4.2, *overfitting* occurs when the number of coefficients equals the number of peaks. This problem illustrates the ill-posed nature of the estimation problem by which is meant that the error function is only defined at the peaks while implicitly a certain smoothness is desired. This can easily be avoided by lowering the number of coefficients. However, when to few coefficients are used, a low pass filtered envelope is obtained that fails to match the peaks accurately.

Obviously, for a sound with a lower pitch, more peaks will be detected in the same interval, and as a consequence more coefficients are needed to match them accurately. Note that when the peaks are positioned exactly at multiples of $\frac{\pi}{K}$, with K being the number of peaks, the estimation of the cepstrum coefficients is equivalent to a discrete inverse fourier transform which implies no information loss. Therefore, the number of cepstrum coefficients is scaled with the ratio of the frequency interval over which the envelope is estimated and the fundamental frequency. Doing so, an accurate matching with the peaks is obtained, while overfitting is avoiding successfully.

4.3.3 Envelope Similarity

Since the perceived loudness of a human listener is approximately logarithmic in function of the amplitude (as in the dB scale), the integral over the square difference between

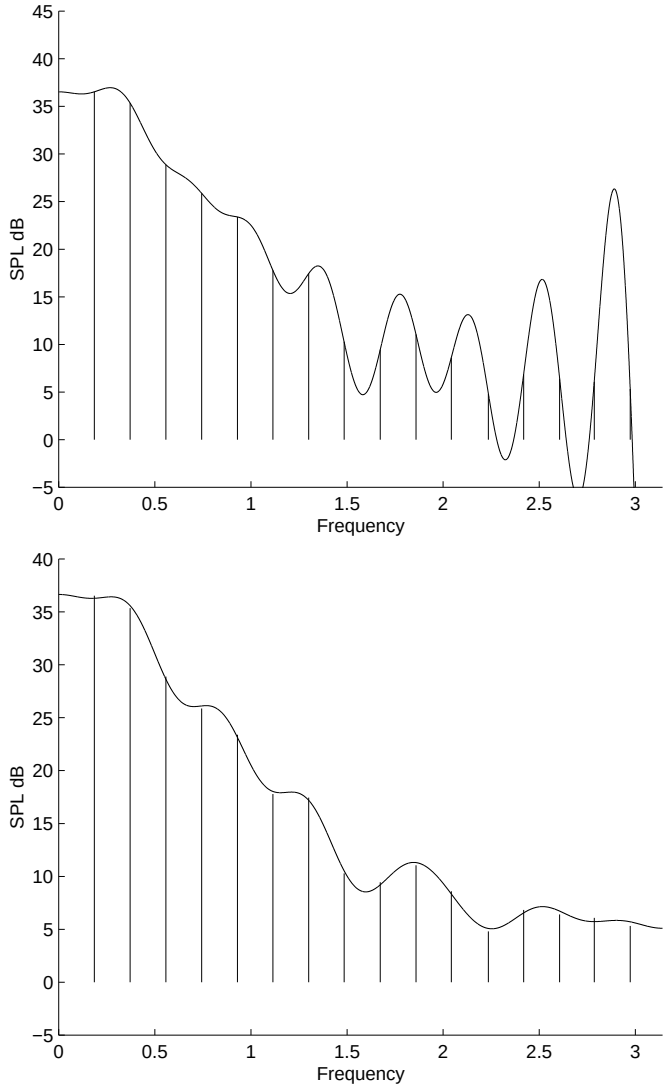


Fig. 4.2: *Spectral envelope estimation over a range of 15000 Hz for a trumpet sound with* $f_0 = 886Hz$ *using* 17 *and* 14 *cepstrum coefficients respectively.*

the log envelopes can be used as a similarity criterium. The following computation shows that this similarity is equivalent to the euclidean distance between the cepstrum coefficients.

Consider two spectral envelopes

$$|H_1(\omega)| = \exp\left(\sum_{p=1}^{P-1}(2-\delta_{0p})c_p^1\cos(p\omega)\right) \tag{4.12}$$

and

$$|H_2(\omega)| = \exp\left(\sum_{p=1}^{P-1}(2-\delta_{0p})c_p^2\cos(p\omega)\right) \tag{4.13}$$

The integral over the square difference between the log envelopes is written as

$$\begin{aligned}
& \frac{1}{2\pi}\int_{-\pi}^{\pi}\left(\log(|H_1(\omega)|)-\log(|H_2(\omega)|)\right)^2 d\omega \\
= & \frac{1}{2\pi}\int_{-\pi}^{\pi}\left(\sum_{p=1}^{P-1}(2-\delta_{p0})(c_p^1-c_p^2)\cos(p\omega)\right)^2 d\omega \\
= & \sum_{p=1}^{P-1}\sum_{q=1}^{P-1}(2-\delta_{op})(2-\delta_{oq})(c_p^1-c_p^2)(c_q^1-c_q^2)\frac{1}{2\pi}\int_{-\pi}^{\pi}\cos(p\omega)\cos(q\omega)d\omega \\
= & \sum_{p=1}^{P-1}\sum_{q=1}^{P-1}(2-\delta_{op})(2-\delta_{oq})(c_p^1-c_p^2)(c_q^1-c_q^2) \\
& \qquad \frac{1}{2\pi}\int_{-\pi}^{\pi}\cos((p+q)\omega)+\cos((p-q)\omega)d\omega \\
= & \sum_{p=0}^{P-1}(c_p^1-c_p^2)^2 \qquad (4.14) \\
= & (\mathbf{c}^1-\mathbf{c}^2)^T(\mathbf{c}^1-\mathbf{c}^2) \qquad (4.15)
\end{aligned}$$

which is the Euclidean distance between two vectors $\mathbf{c}^1$ and $\mathbf{c}^2$ as illustrated by Figure 4.3. This provides a theoretic motivation to represent the log envelopes as points in a vector space where each axis corresponds with a cepstrum coefficient.

4.4 Discrete Mel Frequency Cepstrum Coefficients

4.4.1 Mel Scale

As stated in the introduction, most speech recognition systems use in their front end *Mel-Frequency Cepstrum Coefficients* as features, computed from the output of a filterbank. The center of each filterbank is scaled according to the Mel scale which takes into account the frequency resolution of the human auditory system. Recently, a technique was

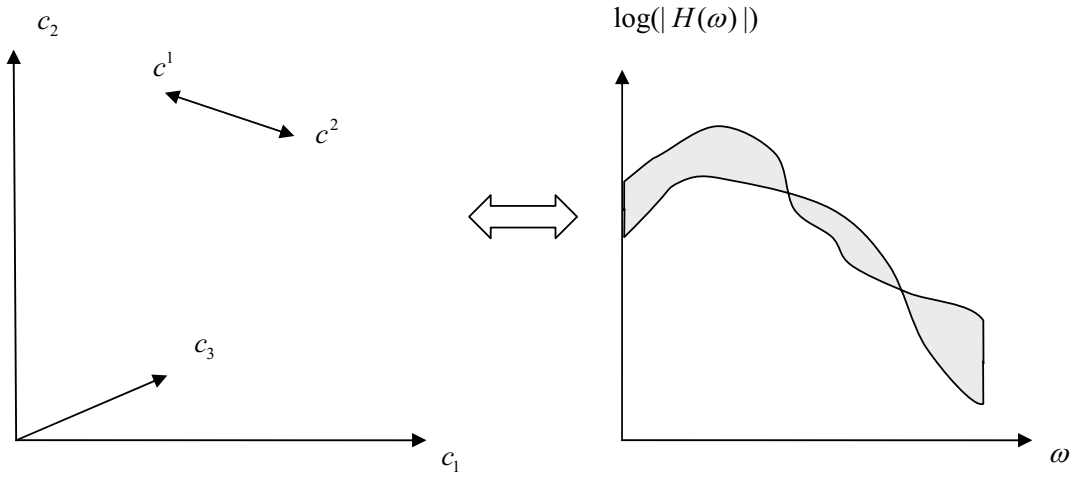


Fig. 4.3: *Spectral Envelope Similarity*

proposed that computes the warped coefficients directly on the power spectrum without the use of the filterbank [57]. The function which computes the Mel scale frequency from the linear frequency is called the *Mel frequency warping function* g and is defined as

$$g(\omega) = 2595 \log\left(1 + \frac{\omega f_s}{2\pi 700 Hz}\right) \tag{4.16}$$

This function is normalized over an interval $[0, \pi]$ using the normalization factor $\frac{\pi}{g(\pi)} g(\omega)$ resulting in the normalized warping fucntion $\bar{g}(\omega)$ defined as

$$\bar{g}(\omega) = \alpha(f_s) \log\left(1 + \omega\beta(f_s)\right) \tag{4.17}$$

with

$$\begin{aligned} \alpha(f_s) &= \frac{\pi}{\log(1 + \frac{f_s}{2 \cdot 700 Hz})} \\ \beta(f_s) &= \frac{f_s}{2\pi 700 Hz} \end{aligned} \tag{4.18}$$

where f_s denotes the sampling frequency. The warping function $\bar{g}(\omega) : [0, \pi] \rightarrow [0, \pi]$ is monotone and invertible, converting a linear scale frequency ω to a Mel scale frequency $\bar{\omega}$. Its inverse function is given by

$$\bar{g}^{-1}(\bar{\omega}) = \frac{1}{\beta}\left(\exp\left(\frac{\bar{\omega}}{\alpha}\right) - 1\right) \tag{4.19}$$

4.4.2 Discrete Mel Frequency Cepstrum Coefficients

Analogue to the MFCC's used in speech, the *discrete MFCC's* were proposed in [31, 33, 32]. Instead of computing the envelope over the linear scale, the peaks were first warped on the Mel-scale using $\bar{g}(\omega)$. Then, the discrete cepstrum is computed over the warped peaks.

However, repositioning the partials results in large gaps in the low frequency band where there are no observations and the envelope is unconstrained. As a result, the high frequency peaks dominate the estimation resulting consistently in overfitted envelopes. The solution of Galas and Rodet consisted in introducing to each observation a cluster of neighboring points which yields satisfying results in many cases but increases the numerical complexity and depends on the initial choice and number of points.

4.4.3 Regularized Mel-Scale Discrete Cepstrum

Cappé introduced a regularization factor to control the smoothness of the envelope by using a modified least squares criterion [7, 8, 9]. This means that an additional term is introduced to the error function which yields a large error for fast variations in the envelope and a small error for slow variations. This regularization function is denoted $R\left[\log(|H(\omega)|)\right]$ and is weighted by a regularization parameter λ. This parameter controls the relative importance of the regularization function and allows to control the trade-off between the exactness and the smoothness of the envelope.

The modified error criterium is written as

$$\chi(\mathbf{c}) \;=\; \sum_{k=1}^{K} \left(\log(|H(\omega_k)|) - \log(|X(\omega_k)|)\right)^2 + \lambda R\left[\log(|H(\omega)|)\right]$$

A possible choice for the penalty function R is to take the integral over the square of the first derivative [8, 9, 7] which is written as

$$\begin{aligned}
& R\left[\log(|H(\omega)|)\right] \\
= \;& \frac{1}{2\pi}\int_{-\pi}^{\pi}\left[\frac{\partial}{\partial\omega}\log(|H(\omega)|)\right]^2 d\omega \\
= \;& \frac{1}{2\pi}\int_{-\pi}^{\pi}\left[\sum_{p=0}^{P-1} -(2-\delta_o p)pc_p\sin(p\omega)\right]^2 d\omega \\
= \;& \frac{1}{2\pi}\sum_{p=0}^{P-1}\sum_{q=0}^{P-1}(2-\delta_{0q})(2-\delta_{0p})pqc_pc_q\int_{-\pi}^{\pi}\sin(p\omega)\sin(q\omega)d\omega \\
= \;& \sum_{p=0}^{P-1} p^2c_p^2 \\
= \;& \mathbf{c}^T\mathbf{P}\mathbf{c}
\end{aligned} \tag{4.20}$$

where $\mathbf{P}$ denotes a matrix of which the diagonal is $[0, 1^2, 2^2, \ldots (P-1)^2]$ and of which all other elements are zero. Analogue to Eq. (4.11), Eq. (4.20) can be written as

$$(\mathbf{Mc} - \mathbf{a})^T(\mathbf{Mc} - \mathbf{a}) + \lambda \mathbf{c}^T\mathbf{Pc} \tag{4.21}$$

for which the minimization with respect to $\mathbf{c}$ yields

$$\begin{aligned} & \mathbf{M}^T(\mathbf{Mc} - \mathbf{a}) + \lambda\mathbf{Pc} = 0 \\ \Rightarrow \quad & (\mathbf{M}^T\mathbf{M} + \lambda\mathbf{P})\mathbf{c} = \mathbf{M}^T\mathbf{a} \\ \Rightarrow \quad & \mathbf{c} = (\mathbf{M}^T\mathbf{M} + \lambda\mathbf{P})^{-1}\mathbf{M}^T\mathbf{a} \end{aligned} \tag{4.22}$$

4.4.4 Posterior Warping

The techniques described in the previous section convert the peaks to the Mel frequency scale before the envelope is estimated. This is named *prior warping*. In addition, the envelope depends on the regularization parameters λ that needs to be set manually by the user and has a large influence on the exactness and smoothness of the fit. As stated in section 4.3.2, it is rather easy to obtain a spectral envelope on the linear scale that is at the same time accurate and smooth by adapting the number of cepstrum coefficients to the number of spectral peaks. This led to the idea of computing the Mel scale cepstrum coefficients a posteriori. This means that first, the linear scale cepstrum coefficients $\mathbf{c}$ are computed from which directly the the Mel scale cepstrum coefficients $\mathbf{d}$ are derived, as shown in Figure 4.4.

The transformation from $\mathbf{c}$ to $\mathbf{d}$ follows directly from their definition. In [57], a warped cosine transform was proposed which allows to compute the warped cepstrum directly on the power spectrum. In this section a method is proposed that does the same for the discrete cepstrum [24].

By defining a spectral envelope over the Mel frequency scale using coefficients $\mathbf{d}$, one obtains

$$|G(\bar{\omega})| = \exp\left(d_0 + 2\sum_{p=1}^{P-1} d_p \cos(p\bar{\omega})\right) \tag{4.23}$$

We wish to determine the coefficients $\mathbf{d}$ of G by transforming H to the Mel scale, which is expressed as

$$|G(\bar{\omega})| = |H(\bar{g}^{-1}(\bar{\omega}))| \tag{4.24}$$

When applying the definition of the discrete cepstrum one obtains

$$\begin{aligned} d_k &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \log(|H(\bar{g}^{-1}(\bar{\omega})|)e^{j\bar{\omega}k}d\bar{\omega} \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \left(\sum_{p=0}^{P-1} c_p(2-\delta_{0p})\cos(p\bar{g}^{-1}(\bar{\omega}))\right) e^{j\bar{\omega}k}d\bar{\omega} \\ &= \sum_{p=0}^{P-1} c_p \frac{(2-\delta_{0p})}{2\pi}\int_{-\pi}^{\pi} \cos\left(p\bar{g}^{-1}(\bar{\omega})\right)\ \ e^{j\bar{\omega}k}d\bar{\omega} \\ &= \sum_{p=1}^{P-1} c_p \frac{(2-\delta_{0p})}{\pi}\int_{0}^{\pi} \cos\left(p\bar{g}^{-1}(\bar{\omega})\right)\ \cos(\bar{\omega}k)d\bar{\omega} \end{aligned} \tag{4.25}$$

where δ_{0p} denotes the Kronecker symbol.

Eq. (4.25) shows that the warped cepstrum coefficients d_k can be computed directly by taking a linear combination of the cepstrum coefficients over the linear scale c_k. This can be written as a matrix product

$$\mathbf{d} = \mathbf{A}\mathbf{c} \tag{4.26}$$

with

$$A_{k+1,l+1} = \frac{(2-\delta_{0l})}{\pi}\int_{0}^{\pi} \cos\left(l\bar{g}^{-1}(\bar{\omega})\right)\ \cos(k\bar{\omega})d\bar{\omega} \tag{4.27}$$

where k and l range from 0 to $K-1$. The integral can be approximated by a discrete sum over N samples for a one sided spectrum, resulting in

$$A_{k+1,l+1} \approx \frac{(2-\delta_{0l})}{N}\sum_{n=0}^{N-1} \cos\left(l\bar{g}^{-1}\left(\frac{\pi n}{N}\right)\right)\cos\left(\frac{\pi n k}{N}\right) \tag{4.28}$$

The similarity of two Mel Frequency spectral envelopes can be computed directly from the linear scale coefficients $\mathbf{c}_1$ and $\mathbf{c}_2$ using

$$\begin{aligned} & (\mathbf{d}_1-\mathbf{d}_2)^T(\mathbf{d}_1-\mathbf{d}_2) \\ = \ & (\mathbf{A}\mathbf{c}_1-\mathbf{A}\mathbf{c}_2)^T(\mathbf{A}\mathbf{c}_1-\mathbf{A}\mathbf{c}_2) \\ = \ & (\mathbf{c}_1-\mathbf{c}_2)^T\mathbf{A}^T\mathbf{A}(\mathbf{c}_1-\mathbf{c}_2) \end{aligned} \tag{4.29}$$

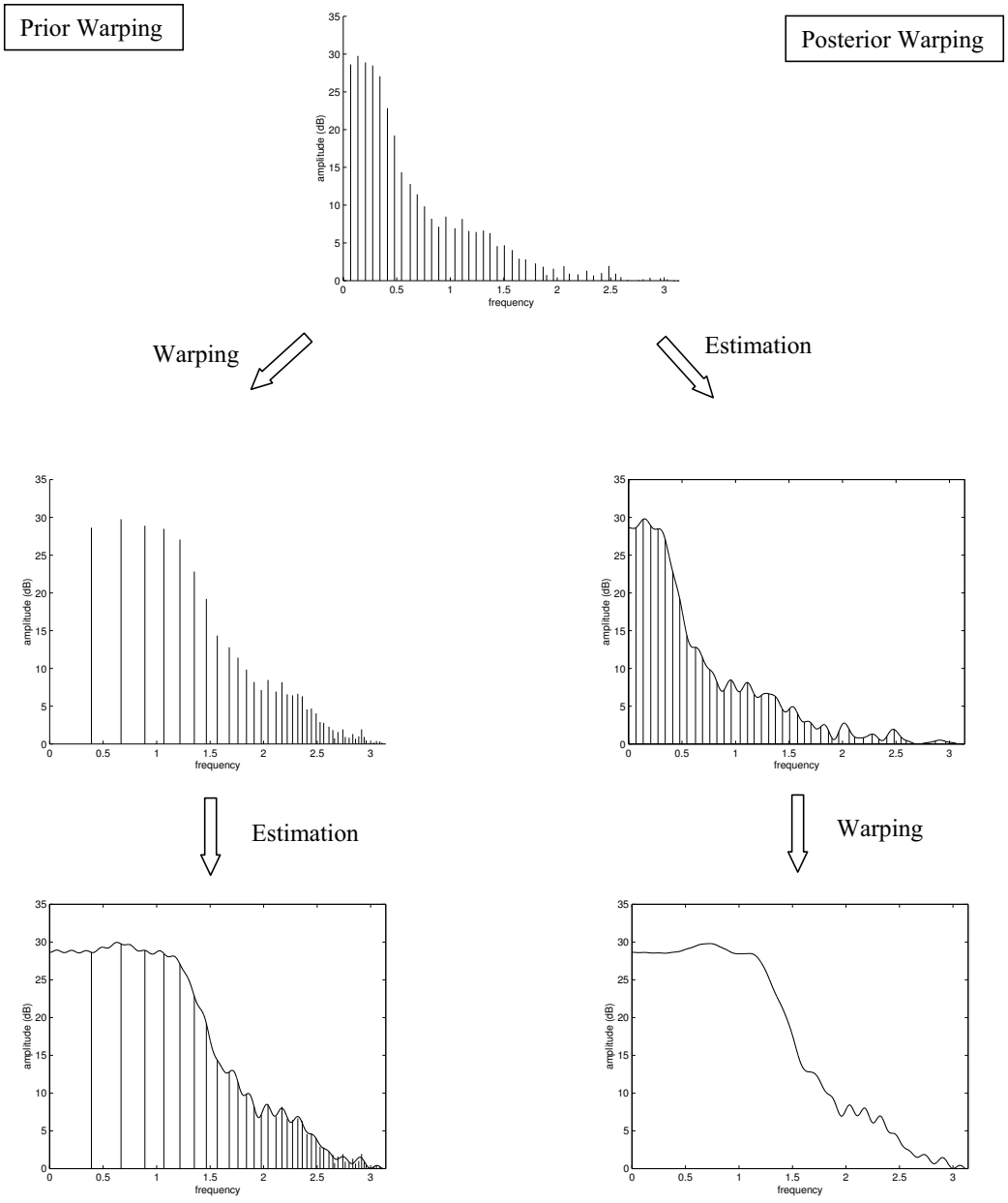


Fig. 4.4: *Prior warping versus posterior warping.*

4.4.5 Analytic Solution for the Mel Scale Warping Matrix

The Mel scale warping matrix can be computed analytically from

$$\begin{aligned} A_{k+1,l+1} &= \frac{(2-\delta_{0l})}{\pi}\int_0^{\pi}\cos(l\bar{g}^{-1}(\bar{\omega}))\cos(\bar{\omega}k)d\bar{\omega} \\ &= \frac{(2-\delta_{0l})}{\pi}\int_0^{\pi}\cos\left(l\frac{1}{\beta}\left(\exp\left(\frac{\bar{\omega}}{\alpha}\right)-1\right)\right)\cos\left(\frac{\bar{\omega}}{\alpha}\alpha k\right)d\bar{\omega} \end{aligned} \tag{4.30}$$

In the case that $l=0$, the expression yields 1 when $k=0$ and 0 when $k>1$. Therefore, the first column of $\mathbf{A}$ consists only of zeros and its first element is 1. For the other cases, $\exp(\frac{\bar{\omega}}{\alpha})$ is substituted by x what implies that $\bar{\omega}=\alpha\log(x)$ and $d\bar{\omega}=\alpha\frac{dx}{x}$ resulting in

$$\begin{aligned} &\frac{(2-\delta_{0l})}{2\pi}\int_1^{\exp\left(\frac{\pi}{\alpha}\right)}\cos\left(\frac{l}{\beta}(x-1)\right)\left(x^{j\alpha k}+x^{-j\alpha k}\right)\alpha\frac{dx}{x} \\ = &\frac{\alpha(2-\delta_{0l})}{2\pi}\int_1^{\exp\left(\frac{\pi}{\alpha}\right)}\cos\left(\frac{l}{\beta}(x-1)\right)\left(x^{j\alpha k-1}+x^{-j\alpha k-1}\right)dx \\ = &\frac{\alpha(2-\delta_{0l})}{4\pi}\int_1^{\exp\left(\frac{\pi}{\alpha}\right)}\left(\exp\left(\frac{jl}{\beta}(x-1)\right)+\exp\left(-\frac{jl}{\beta}(x-1)\right)\right) \\ &\qquad\left(x^{j\alpha k-1}+x^{-j\alpha k-1}\right)dx \\ \equiv &\ \zeta(k,l)+\zeta(k,-l)+\zeta(-k,l)+\zeta(-k,-l) \end{aligned}$$

with

$$\begin{aligned} \zeta(k,l) &= \frac{\alpha(2-\delta_{0l})}{4\pi}\exp\left(\frac{-jl}{\beta}\right)\int_1^{\exp\frac{\pi}{\alpha}}\exp\left(\frac{jl}{\beta}x\right)x^{j\alpha k-1}dx \\ &= \frac{\alpha(2-\delta_{0l})}{4\pi}\exp\left(-\frac{jl}{\beta}\right)\left(-\frac{jl}{\beta}\right)^{-j\alpha k} \\ &\qquad\left[\Gamma(j\alpha k,-\frac{jl}{\beta})-\Gamma(j\alpha k,-\frac{jl}{\beta}\exp\left(\frac{\pi}{\alpha}\right)\right] \end{aligned} \tag{4.31}$$

where $\Gamma(a,x)$ denotes the incomplete gamma function defined as

$$\Gamma(a,x)=\int_t^{\infty}x^{a-1}e^{-x}dx \tag{4.32}$$

An example of a 5×5 posterior warping matrix for an interval from 0 to 15000 Hz is given below. It was computed using the analytically exact solution for $\mathbf{A}$ given in Eq.(4.31) with values of α and β computed using Eq. (4.18) with f_s being 30000.

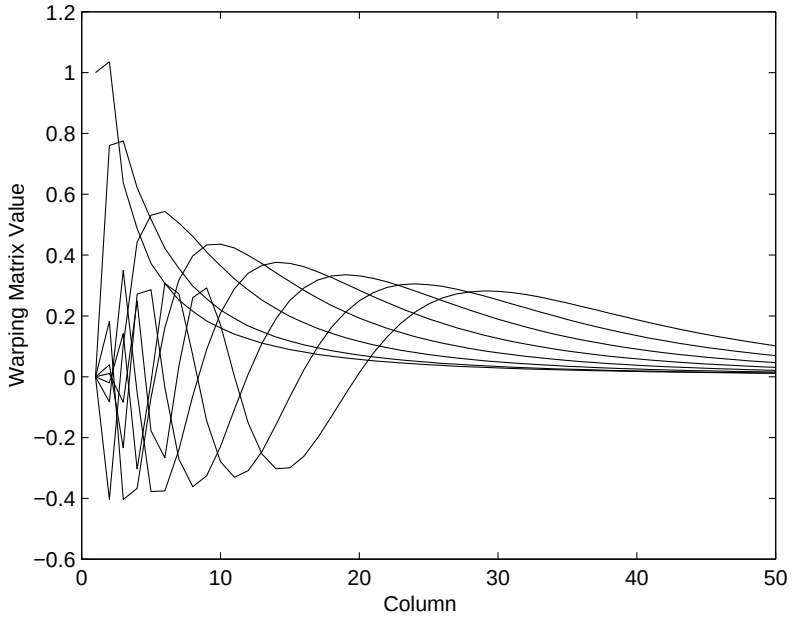


Fig. 4.5: *Warping matrix*

$$\begin{bmatrix} 1 & 1.0354 & 0.6374 & 0.4872 & 0.3727 \\ 0 & 0.7602 & 0.7755 & 0.6217 & 0.5180 \\ 0 & -0.4033 & 0.1050 & 0.4431 & 0.5315 \\ 0 & 0.1820 & -0.4038 & -0.3673 & -0.0690 \\ 0 & -0.0827 & 0.3494 & -0.0521 & -0.3775 \end{bmatrix}$$

Figure 4.5 shows the first eight rows of a 50×50 warping matrix

4.5 Examples

In Fig. 4.2, it was shown that spectral envelopes can be obtained on the linear scale that are at the same time accurate and smooth just by reducing the number of coefficients. Figure 4.6 and 4.7 show Mel scale envelopes of the same spectrum computed by the regularized discrete cepstrum. These figures show that in the case that $\lambda = 0.1$, a smooth envelope is obtained but it fails to match the peaks accurately in the high frequency band. Increasing λ does not seem to solve the matching accuracy and introduces overfitting in the lower frequency band of the envelope. The posterior warped version shown in Fig. 4.8, is at the same time very smooth and very accurate.

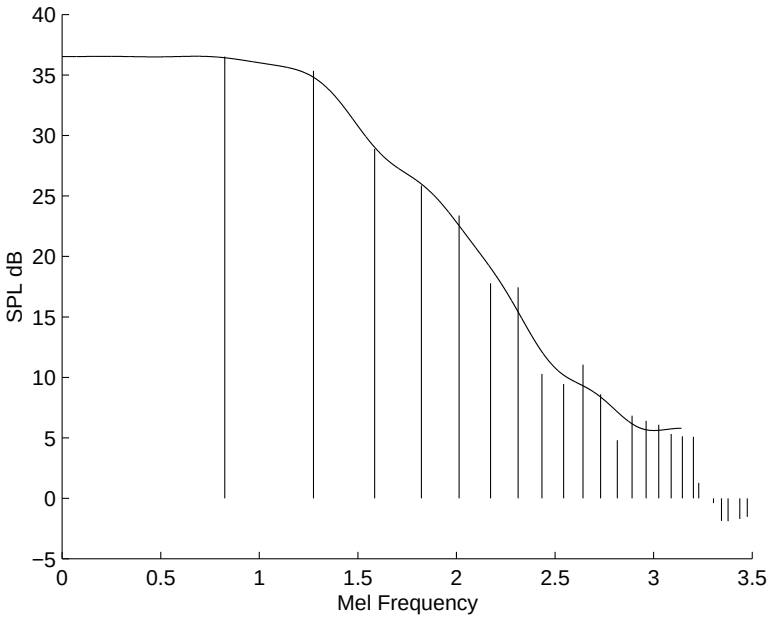


Fig. 4.6: *Regularized discrete cepstrum using 40 cepstrum coefficients with* $\lambda = 0.1$.

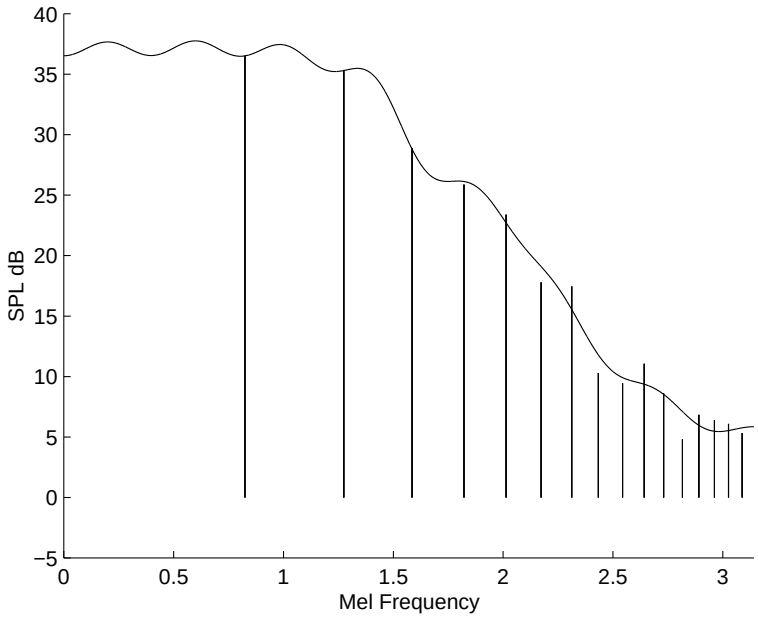


Fig. 4.7: *Regularized discrete cepstrum using 40 cepstrum coefficients with* $\lambda = 0.01$

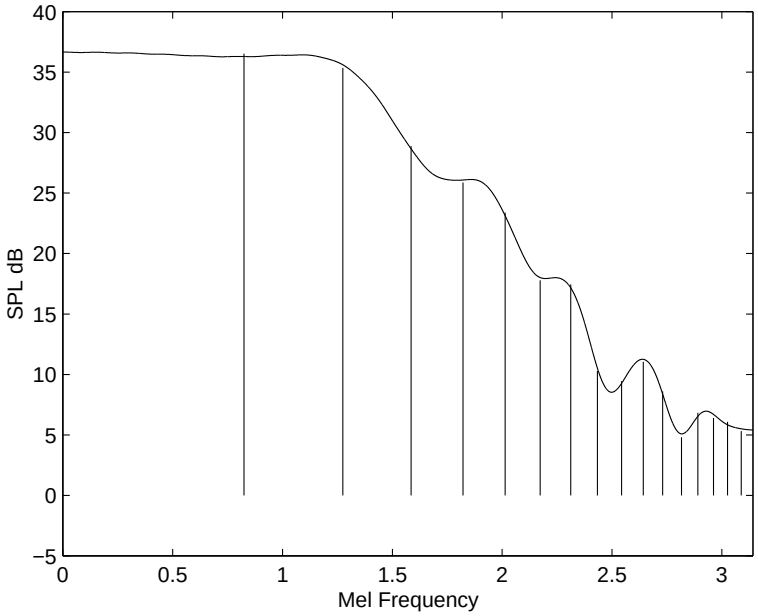


Fig. 4.8: *Discrete cepstrum using 40 cepstrum coefficients computed from 14 cepstrum coefficients on the linear scale using posterior warping.*

4.6 Stability and Perceptual Relevance

The goal of the regularization and posterior warping methods in the previous sections is to avoid overfitting. Because of the ill-posed nature of the estimation problem, very similar spectra can have very different envelopes since the values between the peaks are not defined. As a result, the distance metric given by Eq. 4.29 can yield large values although the envelopes are very similar.

When overfitting was avoided successfully, the discrete cepstra were observed to be very noisy when they were plot in time. This was even the case when the sound was perceived as being very stable. An example is provided in figure 4.9.

The cause of this problem is clarified in Fig. 4.10 where the envelopes of consecutive frames are plot. On the left hand side of the figure it is shown that the envelope of the lower frequency band is very stable over consecutive frames while considerable differences are shown in the high frequency band. These differences come from very small amplitude differences which are amplified enormously by the log function which approaches $-\infty$ when the amplitude approaches zero. The absolute threshold in quiet, represented by the dashed line suggests that these variations are not perceived by a human listener. The cepstrum coefficients on the right hand side of the image are clearly influenced by the variations in the high frequency band. Therefrom must be concluded that this variation in the cepstrum coefficients does not correspond with the perceived variation in timbre. This compromises the cepstrum based distance metric that was proposed previously.

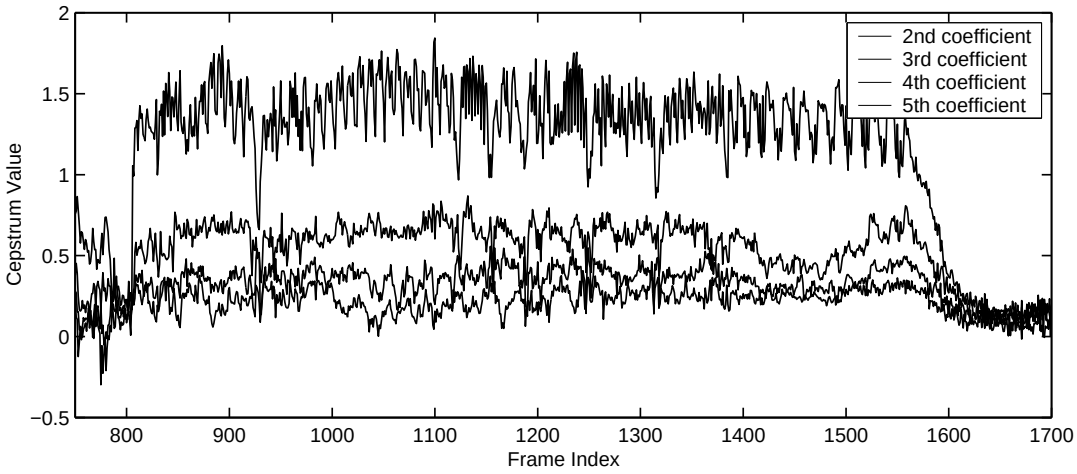


Fig. 4.9: *Discrete Mel frequency cepstrum coefficients over time.*

The stability of the cepstrum coefficients was improved by using a lower bound threshold on the amplitude of the peaks. By replacing amplitudes that were below the threshold with the threshold itself, the influence of these noisy low amplitude partials was significantly reduced. Since most of these partials are situated in the high frequency band, a second method consists in estimating the envelope over a limited spectral band. However, when only the frequency band in synthesized, the perceived quality deteriorates significantly. Fig. 4.11 shows that the linear scale discrete cepstrum coefficients are very noisy what makes them difficult to interpret. When the lower bound threshold is applied, the features become much more stable. One can clearly observe the silence at the beginning and end of the excerpt, the transients between different notes and the periodic variations in the features from frame 1000 to 1100 which is the result of vibrato.

4.7 Conclusions and Further Work

It commonly known that discrete cepstrum coefficients are more appropriate to represent the spectral envelope of quasi periodic signals then linear prediction coefficients or cepstrum coefficients. An additional advantage is that a resynthesis can be obtained from these coefficients which allows to verify to what extent the coefficients represent the sound. This is a property that does not hold for MFCC's.

However, using discrete ceptrum coefficients as perceptual features appeared to be surprising difficult because of overfitting and noise sensitivity. Overfitting is harder to avoid on the Mel scale since the peaks are positioned far apart in the low frequency band which results in large intervals for which the envelope is unconstrained. Several approaches to overfitting have been proposed previously. The technique of cloud smoothing consists of introducing additional control points manually at regions where the envelope is unconstrained. A second method is regularization which controls the trade-off between the exactness and smoothness of the fit by a parameter that has to be

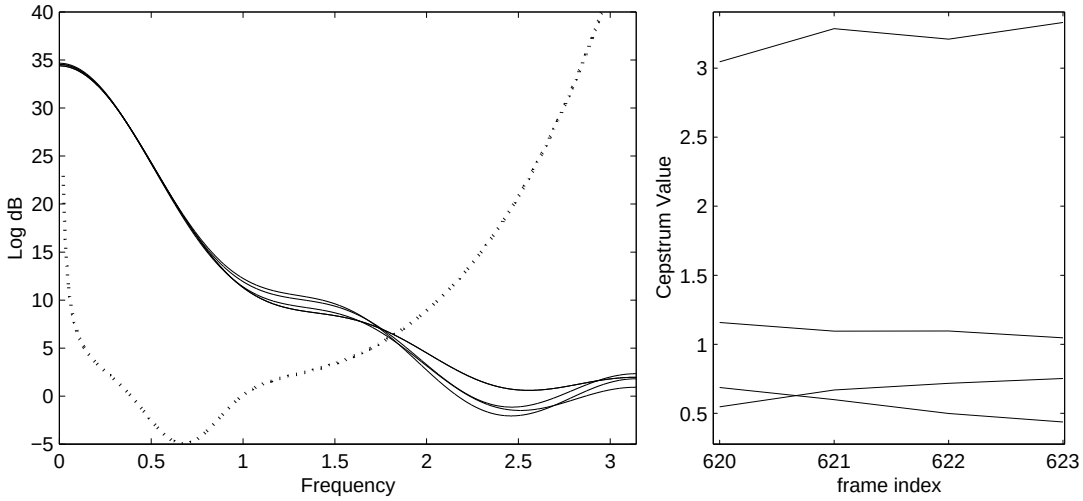


Fig. 4.10: *Spectral envelopes and cepstrum coefficients for consecutive time frames.*

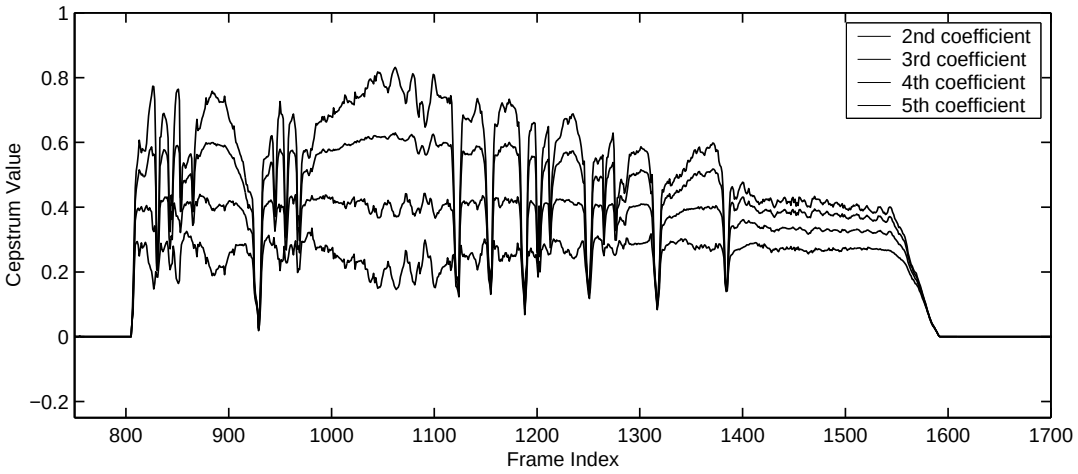


Fig. 4.11: *Stabilized discrete cepstrum.*

set manually. Since it is much easier to obtain a fit that is at the same time accurate and smooth on the linear scale, a method was proposed that converts the linear scale coefficients to the Mel scale coefficients. Since the warping is executed after the envelope estimation, this technique is called *posterior warping* and is computed by a simple matrix multiplication.

Even when the overfitting problem was avoided succesfully, the coefficients were still varying in a very random manner over time. It is shown that these variations come from low amplitude partials in the high frequency band which have a very low signal to noise ratio. This noise sensitivity is amplified enormously by the log function influencing all discrete cepstrum coefficients and was avoided by using a threshold.

Several improvements to this cepstrum based distance metric can be imagined. At the moment, only the frequency resolution of the human auditory system is taken into account by expressing the envelope on the Mel scale. Other important properties such as masking phenomena are not yet taken into account. Another improvement might consist of replacing the log function by a psychoacoustic function which expresses the perceived loudness in function of the amplitude and frequency.

CHAPTER 5

A Conditional Estimation technique for Determining the Control Parameters

5.1 Chapter Overview

A new estimation technique is proposed which computes the control parameters of a physical model of a trumpet in order to simulate a recording of a real instrument. This method takes into account the physical constraints of the instrument and the prior knowledge about how a player controls a trumpet (6.2). This is realized during the design of the data set (6.4) and guarantees that these constraints are respected. Then, an estimation procedure minimizes two perceptual similarity criteria in function of the control parameters. The first criterium expresses the difference of the spectral envelopes and the second one the difference in fundamental frequency (6.3). An optimization technique is proposed that yields an optimal solution for the fundamental frequency, and a conditional suboptimal solution for the spectral envelope (6.5). A robust implementation of the technique was developed for which it is shown that the estimated parameters are unique and that the optimization does not suffer from local minima (6.6). Successful simulations of a trumpet phrase are obtained by the method (6.7).

5.2 Taking into Account Physical Constraints and Prior Knowledge

In chapter 3, the physical constraints are defined for the physical model of the trumpet. Its control parameters $\bar{P}$ consists of the pressure in the mouth P_M, the lip frequency P_L, the tube length P_T and damping factor of the lips P_D. The conditions were derived for which the maximal resonance for this non linear system with delayed feedback was obtained [22]. It was shown that the resonance frequency f_τ of the tube controlled with

a the parameter P_T could be computed using

$$f_\tau = \frac{F_s}{\lfloor \frac{F_s}{2P_T} \rfloor + \lambda_0} \tag{5.1}$$

with F_s being the sampling frequency and λ_0 being a constant. By keeping all parameters constant and varying the the lip frequency it was observed that a maximal resonance was obtained at multiples of f_τ with a value of P_L being three fourth of the frequency yielding

$$P_L = \frac{3}{4} N f_\tau \tag{5.2}$$

where N is an integer value expressing the mode index. Equations (5.1) and (5.2) express the relationship between P_L and P_T for which the resonance is maximal.

An important physical constraint is that the tube length of the instrument must remain constant for each note. In addition, only seven different tube lengths can be used for an entire trumpet performance. Seven tube lengths were determined such that the optimal resonance was achieved for notes tuned to a frequency f_{ref} of 440 Hz. This constraint was not respected by the instance-based approach in chapter 1 [23].

For each note, the player chooses a combination of tube length and mode in order to obtain the desired frequency. The choice of this combination is the *prior knowledge* that a player uses when he plays the instrument and is modelled by the series $P_{T,i}$ and N_i. When the fundamental frequency f is expressed in half tones using

$$I = 12 \log_2 \left(\frac{f}{f_{ref}} \right) + 16 \tag{5.3}$$

then index of the series i corresponds with the rounded value of I. The value 16 is added to make the index of the lowest trumpet note (low $F\sharp$) correspond to 1.

Evidently, the player does not always excite the tube at the exact frequency for which the resonance is maximal. For instance when vibrato is played, the lip frequency varies periodically. To express this deviation we introduce a real valued parameter $\Delta N \in [-0.5, 0.5]$. In order to play a given note with index i when using a deviation ΔN, the lip frequency is computed by

$$P_L = \frac{3}{4} f_{\tau,i} (N_i + \Delta N) \tag{5.4}$$

where $f_{\tau,i}$ is computed from the control parameter $P_{T,i}$ using Eq. (5.1).

5.3 Distance Metrics

The perceived distance between two short time spectra is defined by two components. To express the perceptual similarity in timbre, the difference between the log spectral envelopes was used. This envelope was expressed in terms of *Mel Frequency Discrete*

Cepstrum Coefficients (MFDCC) [7, 8, 32, 78]. In this work a stabilized version was used, computed with *posterior warping* and a *lower bound threshold* [26]. An elegant property of the discrete MFDCC's is that the log difference between the Mel scale spectral envelopes is equivalent with the Euclidean distance between the cepstrum coefficients. In other words, when two spectral envelopes are considered, defined by two cepstrum vectors $\bar{c}_1$ and $\bar{c}_2$ respectively, the spectral similarity D_1 is given by

$$D_1(\bar{c}_1, \bar{c}_2) = (\bar{c}_1 - \bar{c}_2)^T(\bar{c}_1 - \bar{c}_2) \tag{5.5}$$

The square difference of the log fundamental frequency yields the second distance metric.

$$D_2(f_1, f_2) = (\log(f_1) - \log(f_2))^2 \tag{5.6}$$

One can imagine to use a weighted combination of D_1 and D_2, but since the physical meaning of such a combined distance metric is questionable and it is not known how these weights should be determined, we choose to keep the criteria separated.

5.4 Data Set Design and Feature Extraction

The data set was designed by using a fixed set of seven tube lengths that were optimized for a given tuning frequency of 440 Hz (see [22]). This automatically imposes the physical constraints of the acoustic instrument. For every note, and for a range of values of ΔN from -0.06 to 0.06 in steps of 0.01, crescendos were synthesized by varying the pressure P_M slowly from the 0 Pa to 30000 Pa. This data set design guarantees that all the intensities for each note are available and that a variation in fundamental frequency can be realized for the synthesis of vibrato. These are all the elements that are needed to simulate an expressive trumpet performance.

After an additive analysis of the synthesized sounds [71], the discrete MFDCC's and fundamental frequencies were computed. The extraction of the discrete MFDCC's is represented formally by a 2 dimensional function of the control parameters for a given note i as

$$\mathcal{C}^i(\Delta N, P_M) \tag{5.7}$$

and the estimation of the fundamental frequency as

$$\mathcal{F}^i(\Delta N, P_M) \tag{5.8}$$

5.5 Estimation

Given a vector of cepstrum coefficients $\bar{c}$ and a fundamental frequency f computed from a short time window of a recorded sound, the goal of the estimation consists of determining the values of ΔN and P_M which minimize

$$D_1\left(\bar{c}, \mathcal{C}^i(\Delta N, P_M)\right) \tag{5.9}$$

and

$$D_2\left(f, \mathcal{F}^i(\Delta N, P_M)\right) \tag{5.10}$$

Many parameter optimization techniques exist [4], but this case is quite particular since there are two criteria that need to be optimized.

The first step consists of classifying the fundamental frequency to the best note index i. This index is computed by taking the rounded value of I obtained from Eq. 5.3 and identifies the data that will be used for a given note. It follows directly from the inner working of the physical model that the pressure in the mouth P_M has the largest influence on the spectral envelope while ΔN has a predominant influence on fundamental frequency. Therefore, we propose an estimation procedure that consists of two steps. In each step one distance metric is optimized. First, the mouth pressure is optimized for each ΔN with respect to D_1. This results in a function $\mathcal{P}^i(\Delta N; \bar{c})$ yielding the mouth pressure P_M in function of ΔN for which D_1 is minimized given $\bar{c}$ and i. Note that this yields a *conditional optimum*, since it yields the best value of P_M for a given ΔN.

$$\mathcal{P}^i(\Delta N; \bar{c}) = \arg\min_{P_M} D_1\left(\bar{c}, \mathcal{C}^i(\Delta N, P_M)\right) \tag{5.11}$$

Inserting Eq. (5.11) in Eq. (5.10), the distance criterium D_2 only depends on ΔN for a given f and $\bar{c}$. When the value ΔN^* is determined which minimizes the second distance metric D_2 for a given f, the estimation procedure is completed.

$$\Delta N^* = \arg\min_{\Delta N} D_2\left(f, \mathcal{F}^i(\Delta N, \mathcal{P}^i(\Delta N; \bar{c}))\right) \tag{5.12}$$

The condition for this method to work is that an optimal solution for ΔN can retrieved for every mouth pressure value P_M with respect to D_1. Without the additional criterium D_2, this would yield a large set of $(\Delta N, P_M)$ for the desired frequency f. The second criterium allows to select one single value of all these solutions.

Still, the control parameter values need to be computed from ΔN^* and i using

$$P_M^* = \mathcal{P}^i(\Delta N^*; \bar{c}) \tag{5.13}$$

$$P_L^* = \frac{3}{4} f_{\tau,i}(N_i + \Delta N^*) \tag{5.14}$$

$$P_T^* = P_{T,i} \tag{5.15}$$

It is important to note, that the optimization with respect to D_2 is optimal for a given f. By contrast, the value P_M^* only optimizes D_1 in a suboptimal way for $\bar{c}$. This is justified by the fact that small differences in spectral envelope, or timbre, are less disturbing than deviations of the fundamental frequency.

5.6 Implementation

5.6.1 Estimation Example

In the previous section, the estimation procedure was described in terms of continuous functions. Since no parametric or analytic forms of these functions are available they

are practically realized by piecewise linear functions. Instead of repeating the entire derivation for these discrete sampled functions, a practical example is described for a given $\bar{c}$ and f. In this example, f has a value of 786,65 Hz for which Eq. (5.3) yields a value of 26,0588. This implies that $i = 26$, meaning that the 26th data set will be used (see Eq. (5.7) and (5.8)). This data set corresponds with a high G which has been produced by exciting the sixth mode of the tube with a length corresponding with the fingering where all valves are released. This is how the prior knowledge is taken into account and the physical constraints are imposed.

Fig.5.1 shows a plot of $D_1\left(\bar{c}, \mathcal{C}^{26}(\Delta N, P_M)\right)$ in function of P_M for different values of ΔN. One can observe that for each ΔN a global optimum is available but that the error function is quite noisy. In order to make an accurate and robust estimate of the minimum, D_1 is modelled locally by a quadratic approximation for each ΔN that is fit to the observed values by a least squares procedure. This results in

$$D_1\left(\bar{c}, \mathcal{C}^i(\Delta N, P_M)\right) \simeq a(\Delta N)P_M^2 + b(\Delta N)P_M + c(\Delta N) \tag{5.16}$$

In order to realize $\mathcal{P}^i(\Delta N; \bar{c})$ as defined in Eq.(5.11), the minimum of the fit is taken for each ΔN value yielding.

$$\mathcal{P}^i(\Delta N; \bar{c}) = \frac{-b(\Delta N)}{2a(\Delta N)} \tag{5.17}$$

The piecewise linear realization of this function is given on the left side of in Fig. 5.2. Now, the corresponding values of the fundamental frequencies can be retrieved from the data set which was expressed in the previous section by

$$\mathcal{F}^i(\Delta N, \mathcal{P}^i(\Delta N; \bar{c})) \tag{5.18}$$

In Fig. 5.2, the inverse of this function was plot, since we wish to determine ΔN from the given f. This was expressed by Eq. (5.12) and is realized by evaluating the inverse piecewise linear function. The result is depicted by the dashed line in the figure. In this example, the value of f expressed in half tones was $26,0588$, and yielded a value of $\Delta N^* = 0.0082$. When $\mathcal{P}^i(\Delta N; \bar{c})$ is evaluated, a value of P_M was obtained being 13756. Fig. 5.3 shows that ΔN^* yields a suboptimal value with respect to the spectral envelope similarity D_1.

5.6.2 Conclusion

It is shown that for each ΔN a global optimum can be found that can be determined in a robust manner using a local quadratic approximation [4]. This motivates the function $\mathcal{P}$ that expresses the optimal P_M in function of ΔN with respect to D_1. The function $\mathcal{F}^i(\Delta N, \mathcal{P}^i(\Delta N; \bar{c}))$, modelled by a piecewise linear function, was observed to be increasing monotonously. Evidently, its inverse function is also increasing monotonous and therefore a unique solution of ΔN^* is obtained for a given f. Also, the function $\mathcal{P}$ returns a single value of P_M^*. This implies that the obtained solution is unique and that the optimization technique does not suffer from local minima.

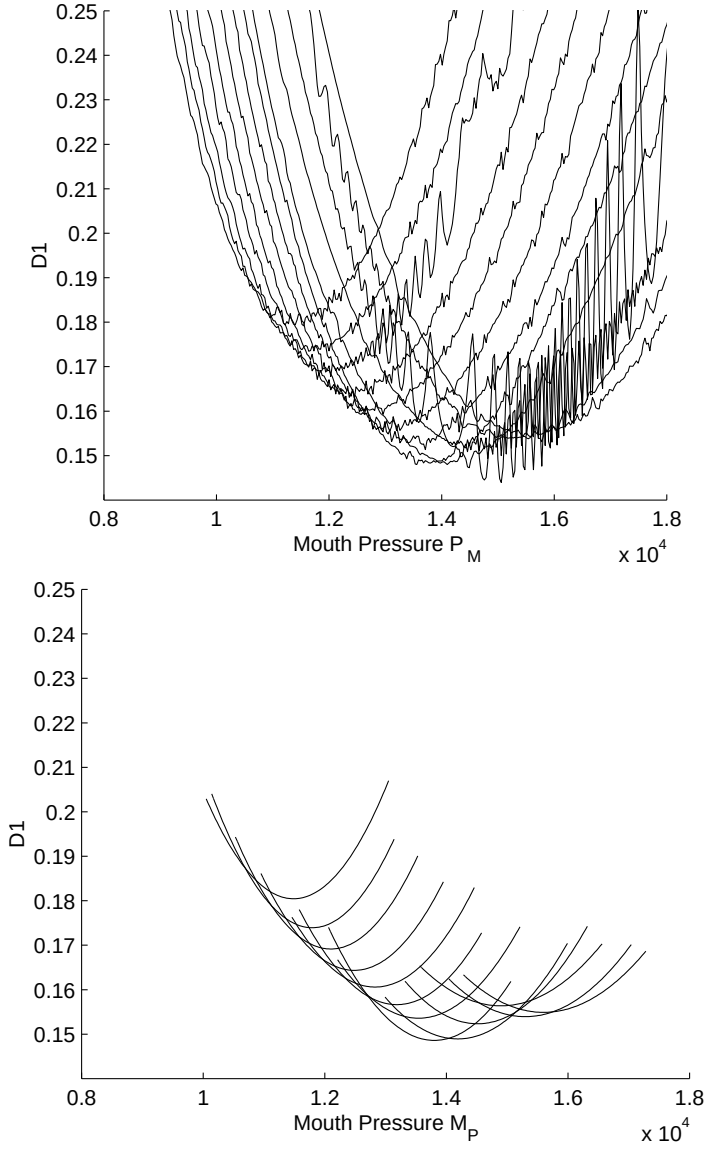


Fig. 5.1: *Spectral similarity in function of the mouth pressure for different ΔN values.* **Left:** *Raw data.* **Right:** *Local quadratic approximation.*

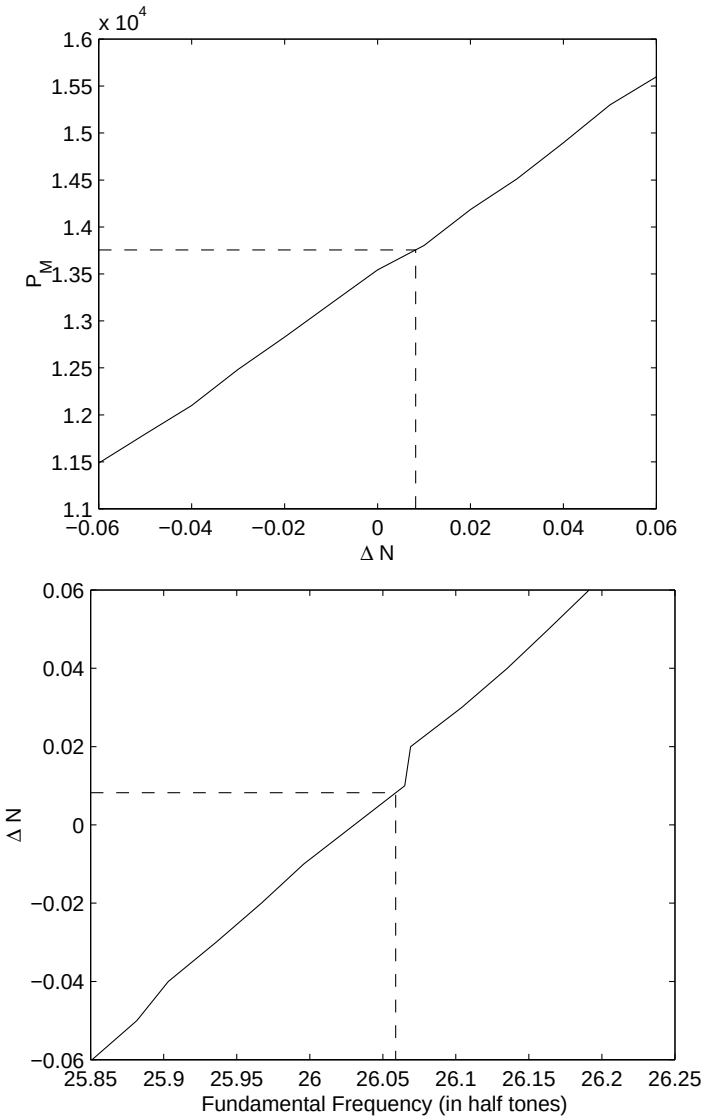


Fig. 5.2: **Left:** *Piecewise linear function denoting* $\mathcal{P}^i(\Delta N; \bar{c})$. **Right:** *Inverted piecewise linear function of* $\mathcal{F}^i(\Delta N, \mathcal{P}^i(\Delta N; \bar{c}))$

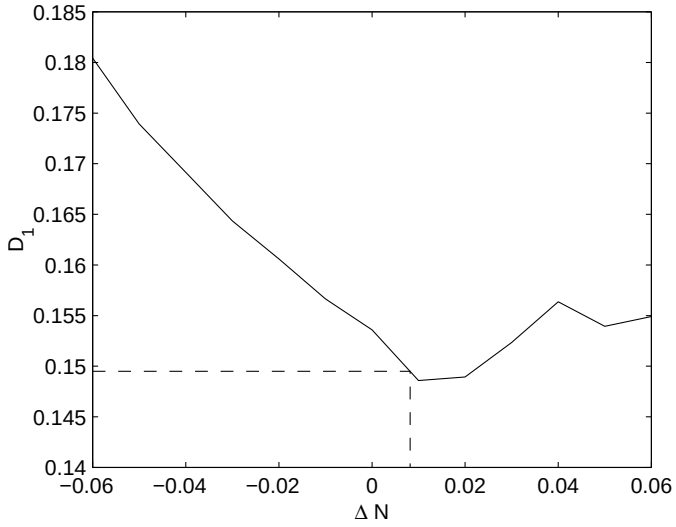


Fig. 5.3: *Suboptimal value for D_1 in function of ΔN*

In the example, it is shown that an exact solution with respect to D_2 was obtained for f using the inverse piecewise linear function. By contrast, the retrieved value of ΔN did not globally optimize D_1, and only a suboptimal solution was obtained. We name this the *conditional optimum* with respect to D_1, since it yields the optimal value of P_M, *given the condition* that D_2 is optimized first for a given f. The motivation of this optimization is the fact that the accuracy of the fundamental frequency has a higher priority than the optimization of the spectral envelope.

5.7 Results

In Fig. 5.4, the results are shown for a musical trumpet phrase. The phrase contains long notes with vibrato, slurred notes and attacked notes which were all simulated successfully. The top figure shows the original signal. The other figures show the estimated control parameters. From these figures, one observes how the mouth pressure follows the amplitude envelopes of the sounds while the lip frequency follows the melodic line of the excerpt including the vibrato at the end of the long sustained notes.

5.7.1 Posterior Tuning

During the derivation in the previous sections, a fixed set of seven tube lengths was assumed with respect to a given tuning frequency. This implied a unique solution for the mouth pressure and lip frequency. When tuning of the instrument is allowed, a solution will be obtained for every possible tuning frequency. Evidently, this tuning allows only slight variations in tube length, since large variations imply that the modes of the tube will fail to correspond with the desired note frequencies.

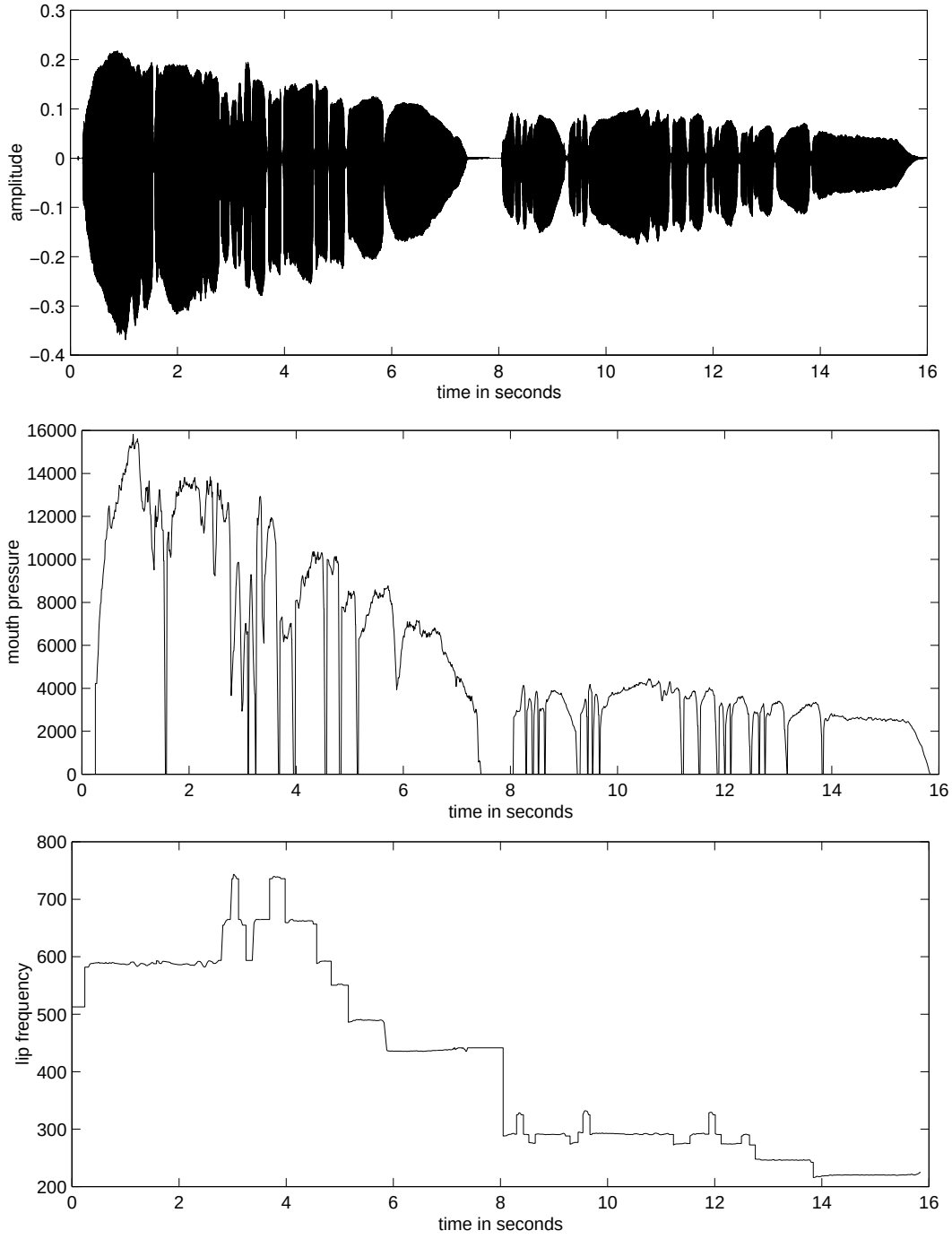


Fig. 5.4: **Top:** *original signal.* **Middle:** *estimated mouth pressure.* **Bottom:** *estimated lip frequency.*

When the control parameters were estimated for a given sound, the value of the fundamental frequency was slightly adapted so that the median frequency of the note corresponded with the median frequency of data set. This was done in order to guarantee that the frequency range of the desired sound was available. However, this results in a simulation which is tuned slightly different than the original sound. This tuning can be compensated a posteriori using the following method. When the tube length and lip frequency are changed in manner so that the ratio

$$\frac{P_L}{f_\tau} = \frac{3}{4}(N_i + \Delta N) \tag{5.19}$$

remains constant, no variation in timbre is perceived. In addition, the expression

$$\Delta f_0 \equiv f_0 - (N + \Delta N) f_\tau \tag{5.20}$$

was observed to be nearly identical for every tube length. Therefore, $N + \Delta N$ and Δf_0 are kept constant while f_τ is adapted in order to be tuned to the desired frequency f_0'. The new tube length f_τ' is then obtained by taking the median value of

$$\frac{f_0' - \Delta f_0}{N + \Delta N} \tag{5.21}$$

for all notes that are played with this specific tube length. The new lip frequency values are finally obtained using $P_L' = \frac{3}{4}(N_i + \Delta N) f_\tau'$.

Fig. 5.5 shows that without posterior tuning, a systematic tuning deviation is obtained between the resynthesis and the original sound. When the tuning is applied, the matching is shown to be very accurate.

5.7.2 Transient Handling and Attack Improvement

The features that are extracted in section 5.4 implicitly assume that the signal is deterministic and stable during the windowed time frame. In the case of transients, this assumption does not hold implying that the feature extraction fails and the parameter estimation technique cannot be applied. However, a transient must always be considered in its context since it is the transition between two stable parts. Otherwise, we would speak of noise instead of a transient. In the case of the trumpet, the onset, offset and slur are the types of transients that can be distinguished. Therefore, a manual annotation of the sound was realized dividing the sound in silence, stable sound, onset, offset and slur. For the onset and offset, the same lip frequency and tube length were taken as for the preceding and consecutive stable part respectively. In the case of the slur, the response function of the tube was cross-faded between two different tube lengths and the lip frequency and mouth pressure were interpolated linearly.

In addition, a problem was observed at the attack being that the relationship between the control parameters and signal features was not instantaneous. In the case of a sustained sound, the lips open and close regularly. Fig. 5.6 shows that in the case of an attack, the lips are pushed open by the pressure in the mouth. Then, when the outgoing

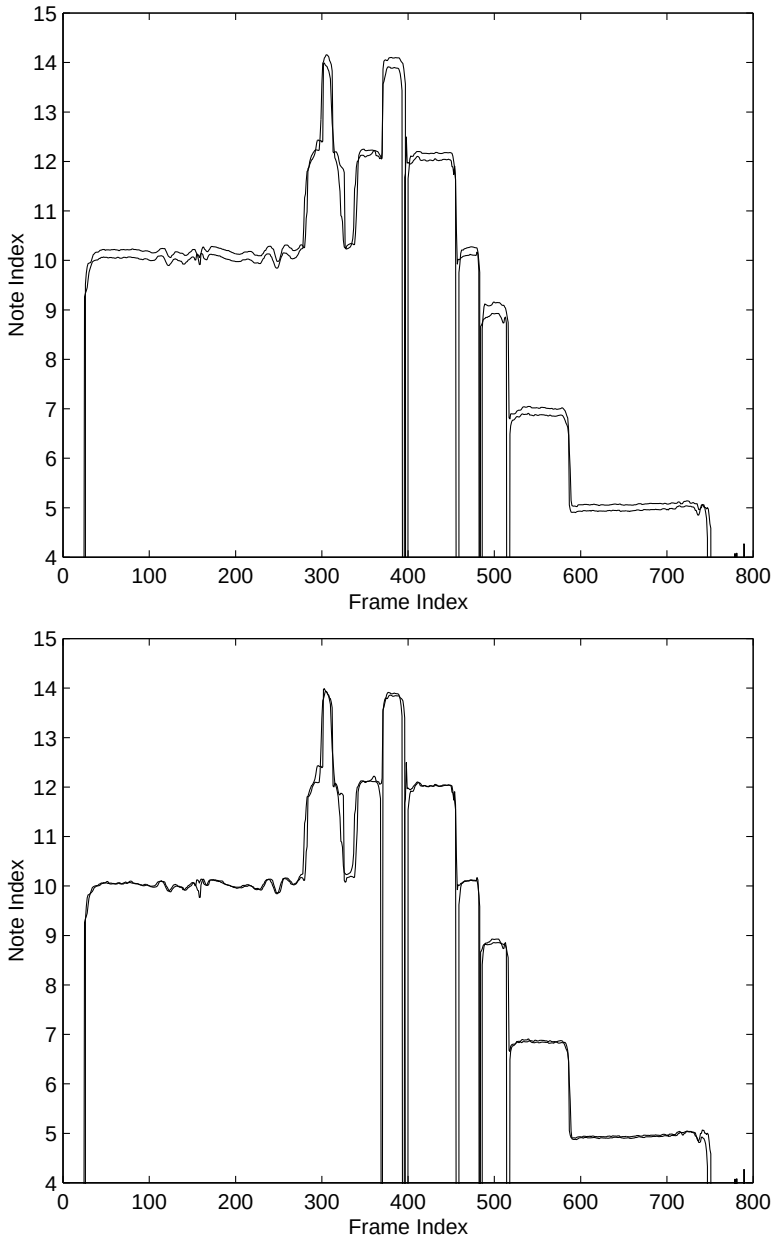


Fig. 5.5: *Comparison of the fundamental frequencies before and after posterior tuning.*

wave returns at the lips, an oscillation is initiated until finally the stable periodic state is reached. However, this procedure takes about 200 ms to complete implying that no sharp attack is obtained.

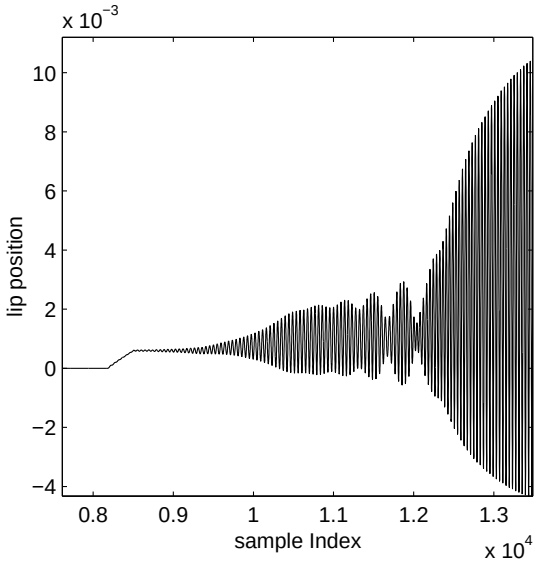


Fig. 5.6: *Lip positions at attack.*

It can be questioned whether the lips are immobile at the beginning of a note since the trumpet player uses the tongue at the attack. The effect of the tongue does not only result in the fact that the pressure augments instantaneously, but also implies an initial speed of the lips when they open. Since the goal of the attack consists in obtaining the stable sustained state as soon as possible, an initial speed was given to the lips resulting in sharper and more realistic attacks.

5.8 Conclusions and Further Research Directions

In this paper, a new automatic non parametric estimation technique is proposed for the control parameters of a physical model of a trumpet. An important aspect is that the control parameters respect the physical constraints of a real instrument and that the prior knowledge about how the instrument is played is incorporated. This means that a correct tube length and mode combination is selected in order to obtain a given note.

For each of these combinations, a data set was produced containing all possible intensities and variations in fundamental frequency in order to allow vibrato. The similarity between two short time segments was expressed by two complementary criteria being the difference in log fundamental frequency, and the difference between the log spectral envelopes. By using a conditional optimization technique and some posterior tuning of the tube length, an exact solution of the fundamental frequency was achieved while a

conditional suboptimal solution was obtained for the spectral envelope. Due to a robust implementation using local quadratic approximations, the estimated control parameters were stable and did not need any post-processing.

Since the estimation can only be applied on stable portions of the sound, an alternative was searched for the transients. These transients were realized successfully by extrapolating control parameters from its context. Also the type of transient was taken into account. Furthermore, the model failed to produce sharp attacks and needed about 200 ms to yield a stable sound. This was improved greatly by adding an additional speed to the lips at the moment of the attack. This initial speed can be related to the effect of the tongue.

The simulation of an expressive trumpet phrase showed that the fundamental frequency could be simulated with a very high accuracy. The timbre on the other hand, is clearly still very different from the original recording. This is due to the fact that the data sets did not contain more similar timbres. Interestingly, the perceived loudness of the simulation was very similar, which confirms the validity and robustness of the distance metric based on the spectral envelope. However, this distance metric has still some limitations. Although the fundamental frequency and the energy distribution over the partials is characterized, the roughness of the sound and the noise component are not taken into account. One can conclude that the estimation technique allows to realize a simulation of the original sound with the physical model that has a similar musical expression. The timbre however, can still be improved. Still, one must keep in mind that the computed signal by the physical model corresponds with the pressure wave at the bell of the instrument. This means that the effect of the room is not incorporated while this has an influence on the timbre. In addition, the estimation technique only allows to determine the gestures of the musician, while a large number of instrument parameters, like for instance the reflection function of the instrument, were assumed to be known. This implies that the resynthesis must be considered as a simulation played with a different instrument.

Finally, we remark that this work confronts the two major synthesis paradigms being the signal modelling paradigm and the physical modelling paradigm. For a wide range of signal models accurate parameter estimation techniques are available. Physical models are generally very difficult to invert. The estimation technique that is proposed is on one hand robust and has a certain generality, but on the other hand it relies on well known parameter estimation techniques from the signal modelling domain. This is at the moment the strongest limitation of the technique. Since no adequate signal parameters can be computed at the transients, it is impossible apply the parameter estimation.

Part II

SINUSOIDAL MODELLING ON SMALL ANALYSIS WINDOWS

CHAPTER 6

Amplitude Estimation: From $\mathcal{O}(N^3)$ to $\mathcal{O}(N \log N)$

6.1 Chapter Overview

In the field of sinusoidal modelling, two types of least squares amplitude estimation methods are distinguished. A first group of methods estimate the complex amplitude of each sinusoid in an iterative manner. Although their main disadvantage is that they are unable to resolve overlapping frequency responses, they are used frequently because of their computational complexity being $\mathcal{O}(N \log(N))$. By contrast, methods that compute all amplitudes simultaneously can resolve overlapping frequency responses but their computational complexity scales with a power of three in function of the number of sinusoidal components. In this chapter, a method is proposed which allows to compute all amplitudes simultaneously and still has an $\mathcal{O}(N \log(N))$ complexity. This is realized by explicitly including an analysis window with a bandlimited frequency response in the least squares derivation resulting in a band diagonal system of equations which can be solved in linear time. Since overlapping frequency responses are allowed, an iterative method must be used to optimize the frequencies resulting in a nonlinear least squares technique. A commonly used technique is Newton optimization which requires the computation of the gradient and the Hessian matrix.

After giving a short overview of the literature on sinusoidal modelling and its applications (6.2), inverse FFT synthesis and the choice of the Blackmann-Harris window is discussed (6.3). It is shown that not only the frequency response of the window w_n is bandlimited, but also the square window and their first and second derivatives. It will be shown that these properties allow to realize a considerable computational gain.

In order to achieve a very high quality, the window length is adapted to three fundamental periods of the sound signal. This window length is not necessarily a power of two

which is desired for the computation of the fourier transform using an FFT. Therefore the *scaled table look-up* method is proposed in section (6.4) which allows to compute the frequency response of a variable length window which is zero padded up to a power of two length.

The least squares derivation and its optimization is discussed in section (6.5). In addition, a preprocessing routine is proposed that determines the number of diagonal bands D that must be taken into account (6.6). Finally, the robustness of the least squares estimator is demonstrated by adding white noise (6.7).

6.2 Introduction

Sinusoidal modelling of musical signals and speech is widely recognized as a very powerful and flexible method. One of the main reasons of its popularity is that it allows to vary pitch and duration of the sound independently [71] allowing sound modifications of a very high quality [5, 34, 35, 64, 101]. In the field of audio coding, perceptual coders have become very popular over the past decade [62, 99, 63]. In the last few years, *parametric coders* have emerged [77] which encode a sound signal by parameters that describe the sinusoids, noise and transients. In addition, when the perceptual relevance of the parameters is taken into account, one obtains a *hybrid coder*, meaning both perceptual and parametric [3, 39]. A third application in which sinusoidal modelling plays a crucial role is model based separation of sound sources [85, 87, 97, 98]. Finally, a fourth application is feature extraction for classification of music and audio. Many features have been proposed that are computed from the sinusoidal parameters which are used in the context of audio annotation [65], instrument recognition [67] and non parametric estimation of control parameters for physical models [24, 25].

In this work, a highly optimized method is developed for modelling a sampled short time signal on which a window w_n is applied. This model $\tilde{x}_n$ consists of a sum of sinusoids which are parameterized by their frequency ω_k, phase ϕ_k and amplitude a_k,

$$\tilde{x}_n = w_n \sum_{k=0}^{K-1} a_k \cos(2\pi\omega_k \frac{n - n_0}{N} + \phi_k) \tag{6.1}$$

The offset value n_0 places the origin of the timescale exactly in the middle of the window so that it is symmetric, resulting in a zero phase response. For a signal with length N, n_0 equals $\frac{N-1}{2}$.

6.2.1 Amplitude Estimation

Early analysis methods estimate the parameters on individual peaks using a parabolic interpolation over the main lobe of the log frequency response [80]. A survey on extensions of this method is given in [46]. These methods however, cannot handle frequency responses that are partially overlapping and therefore require the use of large windows. Since often analysis windows are used of which the main lobe has a width of 4 FFT bins,

the spectral peaks are well defined when the window length is wider then four periods of the lowest frequency in the analyzed signal [15].

A second method is the amplitude estimation of sinusoidal components by using an iterative least squares method [45, 34, 35]. This method, called analysis-by-synthesis / overlap-add (ABS/OLA), detects in an iterative manner the most dominant sinusoidal component, estimates the amplitude and subtracts this component from the spectrum. This can be implemented efficiently by using look-up tables for the frequency responses which results in a time complexity $\mathcal{O}(N \log N)$. Their disadvantage however, is that they cannot resolve close frequencies which result in overlapping frequency responses.

For many applications it is desired to use an amplitude estimator that can handle overlapping frequency responses, for example for the separation of multiple harmonic sound sources or when very small analysis windows are used. Even for monophonic recordings with strong reverberation, the partials of consecutive notes can overlap in time. This can be handled by using a least squares technique which estimates all amplitudes simultaneously [15, 48, 97, 98, 85, 87, 99]. Their major drawback however is their significantly higher computational complexity which is $\mathcal{O}(K^2N)$ with K being the number of sinusoidal components, and N being the signal length. In addition, recent models take into account fast variations in phase and frequency by using polynomial phase and amplitude trajectories [66, 70, 58, 59] or by using exponentially damped sinusoids

All estimation techniques that are cited above, assume implicitly that the noise r_n is white. In the case of strongly colored noise, a weighted least squares technique can be applied which takes into account the correlations in the noise [82]. Also in [82], it is shown that the least squares estimation of the amplitudes using a total least squares is asymptotically consistent.

6.2.2 Frequency Estimation/Optimization

For the parabolic interpolation method [80], the frequency is estimated by determining the value at the optimum of a parabola that is fit to a spectral peak. In the case of overlapping frequency responses and especially for small analysis windows where all responses overlap, an iterative optimization is required. An efficient implementation for this optimization is developed in the next chapter.

6.2.3 Phase Continuity

Most methods assume that the amplitudes and frequencies are constant over the analysis window. For large windows however, an interpolation is desired since small frequency differences between consecutive frames can lead to a phase mismatch at the frame boundaries. In other words, when the assumption that the frequencies and amplitudes are constant does not hold, an interpolation is required to guarantee the phase continuity. A popular method to impose the phase continuity is the cubic phase interpolation method proposed by Mc Aulay and Quatieri [51]. In more recent work, methods are proposed which estimate the slope of the parameters [66, 68] and use spline based trajectories [70].

On the other hand, when the windows are small enough, the constant parameter assumption is more likely to hold and no phase interpolation is required. As a result, an inverse FFT synthesis method can be used with simple overlap-adding (OLA) [34, 35]. In addition, the use of phase interpolation in combination with large windows can even introduce undesired artifacts at the transients. For example, when the signal changes rapidly, the amplitudes and frequencies over the consecutive window will differ largely and the interpolation will introduce an unnatural transition which is perceived as a 'click'.

Therefore, overlap-add synthesis with very short analysis windows seems the best choice to obtain the highest synthesis quality. The small windows make phase interpolation obsolete and yield the additional advantage that discontinuities in amplitude and frequency are allowed at the transients. In addition, the analysis is executed in two phases. First a rough estimation of the frequencies is computed which is then refined on smaller windows adapted to the fundamental frequency.

6.3 Inverse FFT Synthesis and Window Choice

If the signal model defined in Eq. (6.1) would be synthesized by a bank of oscillators, the complexity would be $\mathcal{O}(NK)$ with N being the number of samples and K the number of sinusoidal components. Because of this considerable computational cost, many additive synthesizers construct the time domain signal model $\tilde{x}_n$ by taking the inverse fourier transform of the reciprocal spectrum model $\tilde{X}_m$ [72, 73].

$$\tilde{x}_n = \frac{1}{N} \sum_{m=0}^{N-1} \tilde{X}_m \exp(2\pi i m \frac{n - n_0}{N}) \tag{6.2}$$

By using

$$A_k = a_k \exp(i\phi_k) \tag{6.3}$$

$\tilde{X}_m$ is written as

$$\begin{aligned}
\tilde{X}_m &= \frac{1}{2} \sum_{n=0}^{N-1} w_n \left[\sum_{k=0}^{K-1} A_k \exp(2\pi i \omega_k \frac{n - n_0}{N}) + \right. \\
&\quad \left. \sum_{k=0}^{K-1} A_k^* \exp(-2\pi i \omega_k \frac{n - n_0}{N}) \right] \exp(-2\pi i m \frac{n - n_0}{N}) \\
&= \frac{1}{2} \sum_{k=0}^{K-1} A_k \sum_{n=0}^{N-1} w_n \exp(-2\pi i (m - \omega_k) \frac{n - n_0}{N}) \\
&\quad + \sum_{k=0}^{K-1} A_k^* \sum_{n=0}^{N-1} w_n \exp(-2\pi i (m + \omega_k) \frac{n - n_0}{N}) \\
&= \frac{1}{2} \sum_{k=0}^{K-1} [A_k W(m - \omega_k) + A_k^* W(m + \omega_k)]
\end{aligned} \tag{6.4}$$

with

$$W(m) = \sum_{n=0}^{N-1} w_n \exp(-2\pi i m \frac{n-n_0}{N}) \tag{6.5}$$

This shows that the spectrum model $\tilde{X}_m$ is a linear combination of frequency responses of the window, which are shifted over ω_k and weighted with a complex factor A_k.

Note that, in contrast to many other signal models, the analysis window is explicitly included. In addition, a window is chosen with a bandlimited frequency response. A possible choice is the Blackmann-Harris window given by

$$\begin{aligned} w_n &= a + b\cos(2\pi\frac{n-n_0}{N}) + \\ &\quad c\cos(4\pi\frac{n-n_0}{N}) + d\cos(6\pi\frac{n-n_0}{N}) \end{aligned} \tag{6.6}$$

with $a = 0.35875$, $b = 0.48829$, $c = 0.14128$ and $d = 0.01168$. The frequency response of the Blackmann-Harris window is shown in Fig. 6.1 from which it is observed that the width of the lobe is 2β FFT bins with $\beta = 4$.

For the computations that follow, the bandlimited property of the frequency response plays an important role. This property is not only valid for the window but also for the square window what can be understood easily when taking into account that taking the square in the time domain is equivalent with a convolution in the frequency domain. It must be noted however that this doubles the bandwidth of the main lobe. In Figure 6.1 and 6.2 it is shown that also the derivatives of these frequency responses are bandlimited. The derivative in the frequency domain is equivalent with the multiplication with a straight line in the time domain as can be observed from following equalities.

$$\begin{aligned} W(m) &= \sum_{n=0}^{N-1} w_n \exp(-2\pi i m \frac{n-n_0}{N}) \\ W'(m) &= \sum_{n=0}^{N-1} \left(-2\pi i \frac{n-n_0}{N}\right) w_n \exp(-2\pi i m \frac{n-n_0}{N}) \\ W''(m) &= \sum_{n=0}^{N-1} \left(-2\pi i \frac{n-n_0}{N}\right)^2 w_n \exp(-2\pi i m \frac{n-n_0}{N}) \\ Y(m) &= \sum_{n=0}^{N-1} w_n^2 \exp(-2\pi i m \frac{n-n_0}{N}) \\ Y'(m) &= \sum_{n=0}^{N-1} \left(-2\pi i \frac{n-n_0}{N}\right) w_n^2 \exp(-2\pi i m \frac{n-n_0}{N}) \\ Y''(m) &= \sum_{n=0}^{N-1} \left(-2\pi i \frac{n-n_0}{N}\right)^2 w_n^2 \exp(-2\pi i m \frac{n-n_0}{N}) \end{aligned} \tag{6.7}$$

These equalities will reoccur frequently in the following sections and the next chapter.

By using a lookup table for the main lobe, these frequency responses can be evaluated in constant time, resulting in the fact that the summation over n can be avoided.

6.4 Frequency Response of a Zero Padded Variable Length Window

In this section, the *scaled table look-up* method is presented which allows to compute the frequency response of a window with length M' which is zero padded up to a length N'. This allows to obtain a better adjustment of the window length to the fundamental period of the sound. The purpose of the zero padding is to obtain again a power of two length so that an FFT can be used to compute the spectrum.

6.4.1 Scaled Table Look-up

A computationally efficient method to compute this frequency response of a zero padded variable length window is described of which a schematic representation is given in Fig. 6.3. The fourier transform of a window with length M is denoted as $W^M(m)$, being

$$W^M(m - m_0) = \sum_{n=0}^{M-1} w^M(n - m_0) \exp(-2\pi i \frac{(n - m_0)(m - m_0)}{M}) \tag{6.8}$$

with $m_0 = \frac{M-1}{2}$. The window $w^M(n)$ is now padded up to a length N which is denoted $w_M^N(n)$. Its frequency response $W_M^N(m)$ can be expressed in terms of the non padded version $W^M(m)$ using

$$\begin{aligned} W_M^N(m - n_0) &= \sum_{n=0}^{N-1} w_M^N(n - n_0) \exp(-2\pi i \frac{(m - n_0)(n - n_0)}{N}) \\ &= \sum_{n=0}^{M-1} w^M(n - m_0) \exp(-2\pi i \frac{(m - n_0)M}{N} \frac{(n - m_0)}{M}) \\ &= W^M(m\frac{M}{N} - m_0) \end{aligned} \tag{6.9}$$

where m ranges from 1 to M and $n_0 = \frac{N-1}{2}$. This shows that when the frequency response of a window is zero padded, a scaled version of the original frequency response is obtained. As a result, the spectral bandwidth of the frequency response is enlarged to $-\frac{N}{M}\beta \leq m \leq \frac{N}{M}\beta$.

Now, the truncated inverse fourier transform of $W_M^N(m - n_0)$ is taken over a length N'. This is denoted $W_{M'}^{N'}(m - n_0')$ with $n_0' = \frac{N'-1}{2}$. This truncation introduces only a very error change since the frequency response of the Blackmann-Harris window is very small outside the main lobe. Taking into account that $w_M^N(n - m_0)$ is the inverse fourier transform of $W_M^N(m - m_0)$

$$w_M^N(n - n_0) = \frac{1}{N} \sum_{m=0}^{N-1} W_M^N(m - m_0) \exp(2\pi i \frac{(n - m_0)(m - n_0)}{N}) \tag{6.10}$$

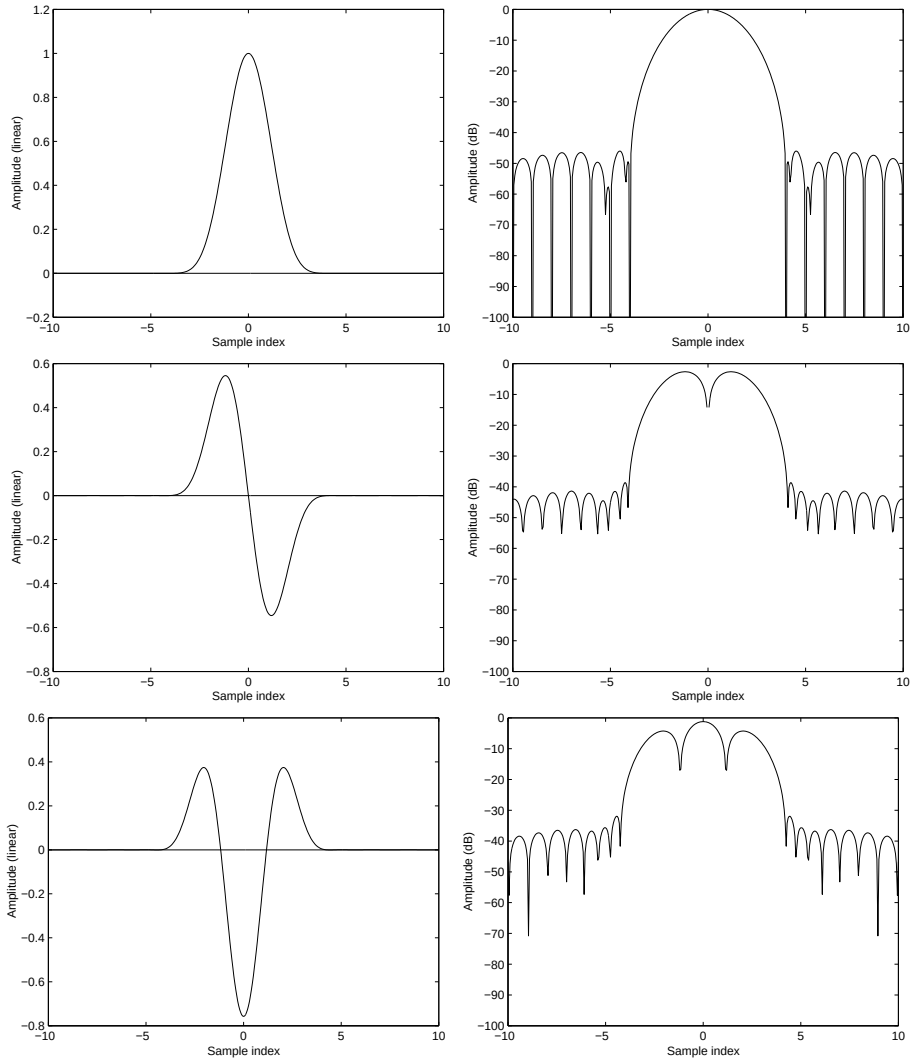


Fig. 6.1: **Top:** *Frequency response of Blackmann-Harris window* $W(m)$, **Middle:** *First derivative of the frequency response of Blackmann-Harris window* $W'(m)$, **Bottom:** *Second derivative of the frequency response of Blackmann-Harris window* $W''(m)$

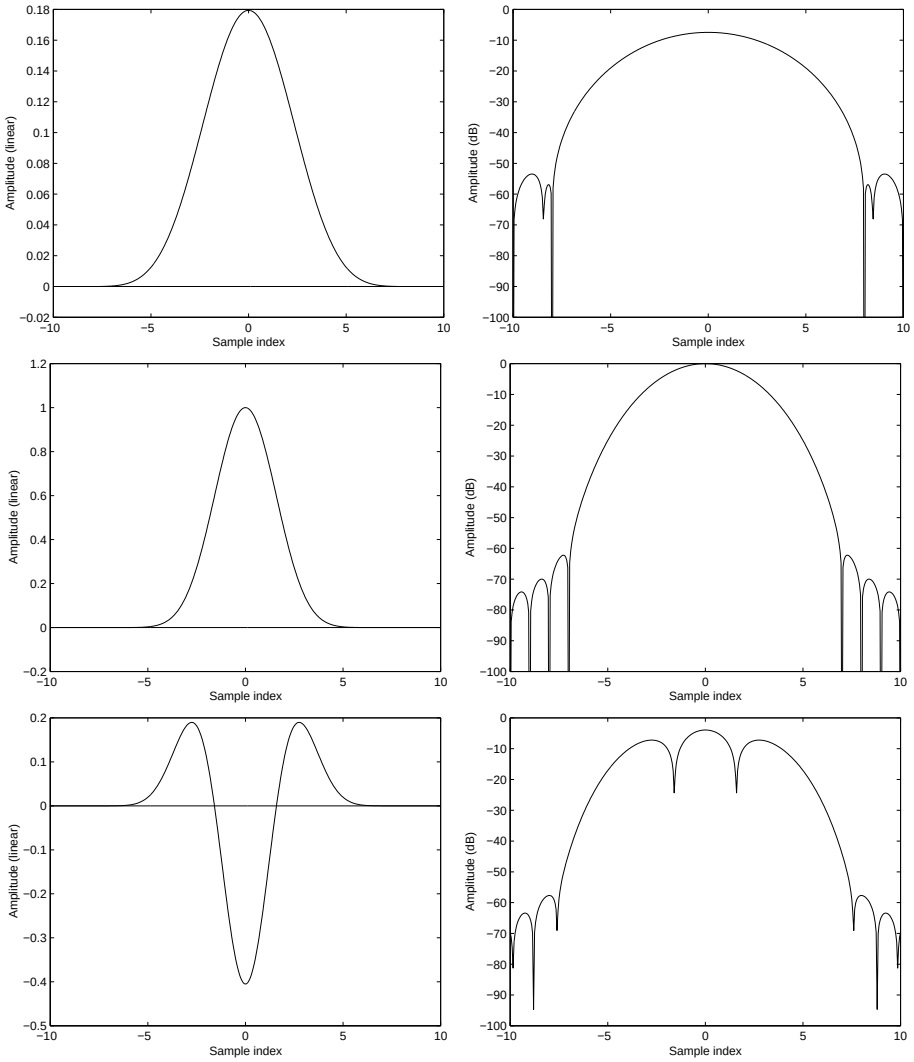


Fig. 6.2: **Top:** *Frequency response of zero padded Blackmann-Harris window* $W_M^N(m)$, **Middle:** *Frequency response of squared Blackmann-Harris Window* $Y(m)$, **Bottom:** *Second derivative of the frequency response of the Squared Blackmann-Harris Window* $Y''(m)$

the truncated inverse fourier transform $w_{M'}^{N'}(n-n_0')$ can now be written as

$$\begin{aligned} w_{M'}^{N'}(n-n_0') &= \frac{1}{N'}\sum_{m=0}^{N'-1} W_{M'}^{N'}(m-n_0')\exp(2\pi i\frac{(n-n_0')(m-n_0')}{N'}) \\ &= \frac{1}{N'}\sum_{m=0}^{N-1} W_{M}^{N}(m-n_0)\exp(2\pi in\frac{(n-n_0')N}{N'}\frac{(m-n_0)}{N}) \\ &= \frac{N}{N'}w_M^N(n\frac{N}{N'}-n_0) \\ &= \frac{N}{N'}w^M(n\frac{N}{N'}-m_0) \end{aligned} \tag{6.11}$$

from which follows that the truncation of the spectrum yields a scaling of the zero padded window. This implies that after this scaling, the window length M' is given by

$$M' = M\frac{N'}{N} \tag{6.12}$$

By applying Eq. (6.9), the zero padded and resized window $w_{M'}^{N'}(n-n_0')$ can be expressed in function of the original frequency response $W^M(m-m_0)$ yielding

$$\begin{aligned} w_{M'}^{N'}(n-n_0') &= \frac{1}{N'}\sum_{m=0}^{N'-1} W_{M'}^{N'}(m-n_0')\exp(2\pi i\frac{(n-n_0')(m-n_0')}{N'}) \\ &= \frac{1}{N'}\sum_{m=0}^{N'-1} W_{M}^{N}(m-n_0')\exp(2\pi i\frac{(n-n_0')(m-n_0')}{N'}) \\ &= \frac{1}{N'}\sum_{m=0}^{N'-1} W^{M}(\frac{M}{N}m-n_0')\exp(2\pi i\frac{(n-n_0')(m-n_0')}{N'}) \\ &= \frac{1}{N'}\sum_{m=0}^{N'-1} W^{M}(\frac{M'}{N'}m-n_0')\exp(2\pi i\frac{(n-n_0')(m-n_0')}{N'}) \end{aligned}$$

However, Eq. (6.11) shows that the spectrum truncation results not only in a scaling of the window but also in a multiplication of the window with a factor $\frac{N}{N'}$. Therefore, this scaled window must be normalized with a factor $\frac{N'}{N}$ resulting in

$$\frac{N'}{N}w_{M'}^{N'}(n-n_0') = \frac{1}{N}\sum_{m=0}^{N'-1} W^{M}(\frac{M'}{N'}m-m_0)\exp(2\pi i\frac{(n-n_0')(m-n_0')}{N'}) \tag{6.13}$$

with $N = N'\frac{M}{M'}$.

6.4.2 Variable Length Inverse FFT Synthesis

The previous method can be applied to realize a variable length additive synthesis. By using a look-up table for the main lobe of W^M, computed from a window of length M,

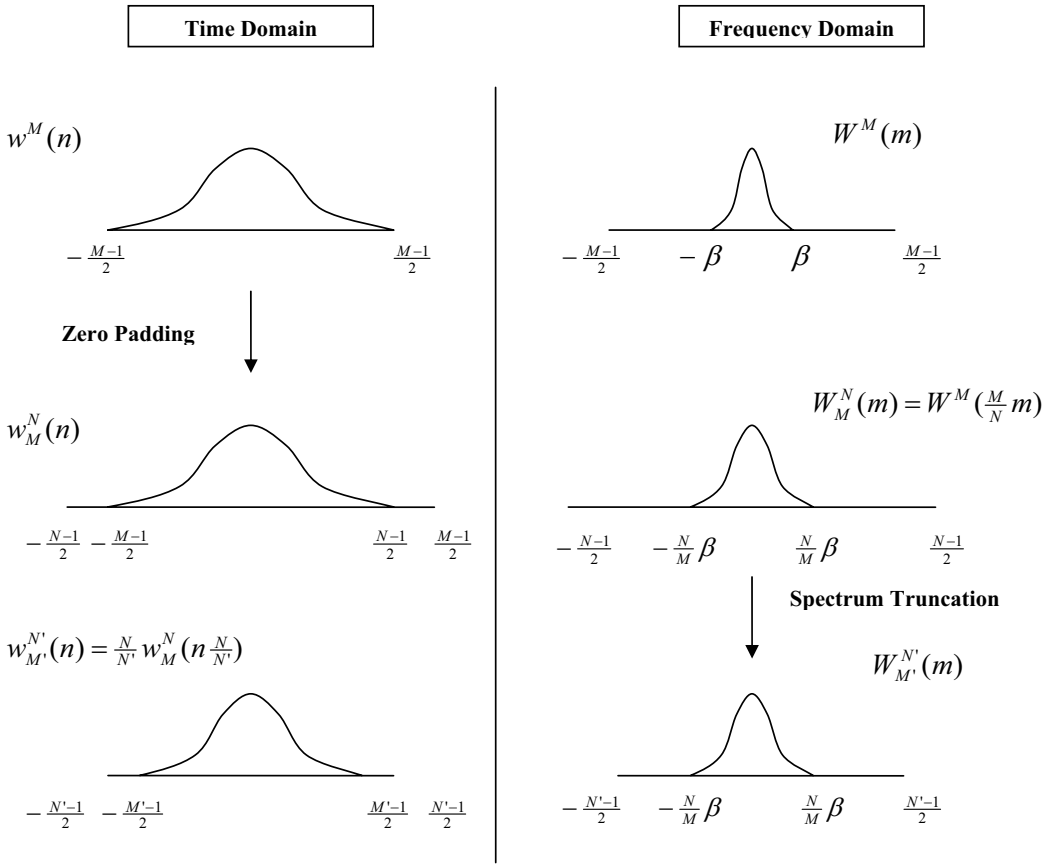


Fig. 6.3: *Theoretic Motivation for the Scaled Table Look-up.*

any scaled and zero padded window can be realized. Since the time domain signal is computed efficiently by an inverse fast fourier transform (IFFT), it is desired that the zero padded window length is a power of two. For a window with size M', the smallest FFT size N' is obtained by taking

$$N' = 2^{\lceil \log_2(M') \rceil} \tag{6.14}$$

As a result, the parameters that are required to compute the variable length frequency response given in Eq. (6.13) are

- M: window length used to compute the look-up table
- N': desired FFT size
- M': desired window size

The oversampled main lobe of $W(m)$ is stored in a table T_i. The table has a length i_L and the first index i of the table is denoted i_0. These index values correspond with the m-values over a range $[m_a, m_b]$. This leads to the following relation between the input value m and index i

$$m = m_a + (m_b - m_a)\frac{i - i_0}{i_L - 1} \tag{6.15}$$

$$i = i_0 + (i_L - 1)\frac{m - m_a}{m_b - m_a} \tag{6.16}$$

The values of $W(m)$ are obtained by a simple linear interpolation between the closest i-values yielding

$$W(m) = (i - \lfloor i \rfloor)T_{\lfloor i \rfloor} + (1 - i + \lfloor i \rfloor)T_{\lfloor i \rfloor + 1} \tag{6.17}$$

where i is computed from m using the previous formula.

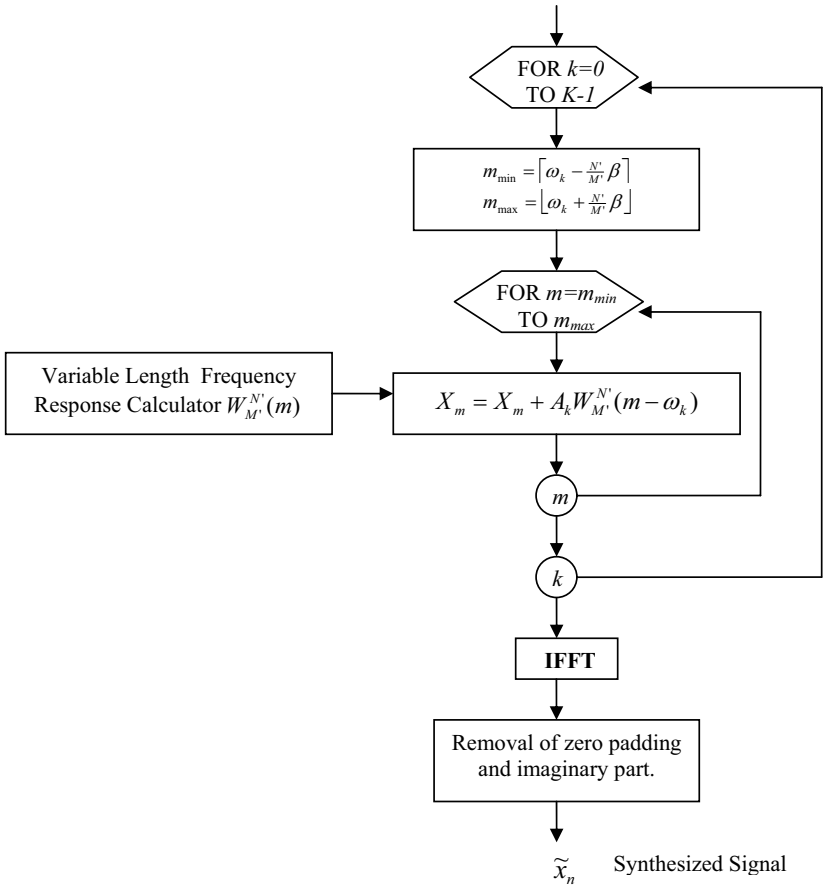
When a window with length M' is taken which is zero padded up to a length N', the main lobe is enlarged up to a size $2\frac{N'}{M'}\beta$. Therefore, the synthesis of a frequency ω_k (see Eq. 6.4) requires the computation for all frequency domain samples m for which

$$m_{min} \leq \quad m \quad \leq m_{max}$$

with

$$\begin{aligned} m_{min} &= \lceil \omega_k - \frac{N'}{M'}\beta \rceil \\ m_{max} &= \lfloor \omega_k + \frac{N'}{M'}\beta \rfloor \end{aligned} \tag{6.18}$$

The inverse FFT synthesis algorithm is illustrated in Figure 6.4.



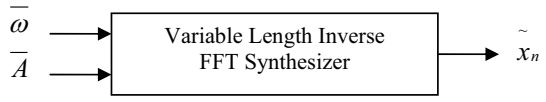


Fig. 6.4: *Variable Length Inverse FFT Synthesis*

6.5 Efficient Least Squares Amplitude Estimation

As stated in the introduction, it is desired to work on small analysis windows so that overlapping frequency responses can be handled. This requires a least squares method that computes all amplitudes simultaneously. The original computational complexity of this method is $\mathcal{O}(K^2N)$ where the K denotes the number of partials and N the window length. In this section however, a method is proposed which solves this problem in $\mathcal{O}(N\log(N))$ and reduces the space complexity, which is originally $\mathcal{O}(K^2)$, to $\mathcal{O}(K)$. The complete amplitude computation method is illustrated in Figure 6.6.

6.5.1 Complex Amplitude Computation

In order to determine the complex ampitudes, Eq. (6.1) is reformulated as a sum of cosines and sines where the real part of the complex amplitude is denoted $A_l^r = a_l\cos\phi_l$ and the imaginary part as $A_l^i = a_l\sin\phi_l$. The signal model for the short time signal $\tilde{x}_n$ can now be written as

$$\begin{aligned}\tilde{x}_n &= w_n\frac{1}{2}\sum_{k=0}^{K-1}\left(A_k\exp(2\pi i\omega_k\frac{n-n_0}{N}) + A_k^*\exp(-2\pi i\omega_k\frac{n-n_0}{N})\right)\\ &= w_n\sum_{k=0}^{K-1}\left(A_k^r\cos(2\pi\omega_k\frac{n-n_0}{N}) - A_k^i\sin(2\pi\omega_k\frac{n-n_0}{N})\right)\end{aligned} \tag{6.19}$$

The error function $\chi(\bar{A};\bar{\omega})$ expresses the square difference between the samples in the windowed signal x_n and the signal model $\tilde{x}_n$,

$$\chi(\bar{A};\bar{\omega}) = \sum_n (x_n - \tilde{x}_n)^2 \tag{6.20}$$

This notation indicates that the error is minimized with respect to a vector of variables $\bar{A}$ for a given set of frequencies $\bar{\omega}$ that are assumed to be known. The minimization is realized by putting the partial derivatives with respect to the unknowns to zero

$$\frac{\partial\chi(\bar{A};\bar{\omega})}{\partial A_l^r} = 0, \frac{\partial\chi(\bar{A};\bar{\omega})}{\partial A_l^i} = 0 \tag{6.21}$$

resulting respectively in

$$\begin{aligned}&\sum_{k=0}^{K-1}A_k^r\left(\sum_{n=0}^{N-1}w_n^2\cos(2\pi\omega_k\frac{n-n_0}{N})\cos(2\pi\omega_l\frac{n-n_0}{N})\right)\\ -&\sum_{k=0}^{K-1}A_k^i\left(\sum_{n=0}^{N-1}w_n^2\sin(2\pi i\omega_k\frac{n-n_0}{N})\cos(2\pi\omega_l\frac{n-n_0}{N})\right)\\ &= \sum_{n=0}^{N-1}x_nw_n\cos(2\pi\omega_l\frac{n-n_0}{N})\end{aligned} \tag{6.22}$$

and

$$
\begin{aligned}
-\sum_{k=0}^{K-1} A_k^r \left(\sum_{n=0}^{N-1} w_n^2 \cos(2\pi\omega_k \frac{n-n_0}{N}) \sin(2\pi\omega_l \frac{n-n_0}{N}) \right) \\
+\sum_{k=0}^{K-1} A_k^i \left(\sum_{n=0}^{N-1} w_n^2 \sin(2\pi i\omega_k \frac{n-n_0}{N}) \sin(2\pi\omega_l \frac{n-n_0}{N}) \right) \\
= -\sum_{n=0}^{N-1} x_n w_n \sin(2\pi\omega_l \frac{n-n_0}{N})
\end{aligned} \tag{6.23}
$$

These two sets of K equations have $2K$ unknown variables what can be written in the following matrix form

$$
\begin{bmatrix} \mathbf{B^{1,1}} & \mathbf{B^{1,2}} \\ \mathbf{B^{2,1}} & \mathbf{B^{2,2}} \end{bmatrix} \begin{bmatrix} \mathbf{A^r} \\ \mathbf{A^i} \end{bmatrix} = \begin{bmatrix} \mathbf{C^1} \\ \mathbf{C^2} \end{bmatrix} \tag{6.24}
$$

with

$$
\begin{aligned}
B_{l,k}^{1,1} &= \sum_{n=0}^{N-1} w_n^2 \cos(2\pi\omega_k \frac{n-n_0}{N}) \cos(2\pi\omega_l \frac{n-n_0}{N}) \\
B_{l,k}^{1,2} &= -\sum_{n=0}^{N-1} w_n^2 \sin(2\pi\omega_k \frac{n-n_0}{N}) \cos(2\pi\omega_l \frac{n-n_0}{N}) \\
B_{l,k}^{2,1} &= -\sum_{n=0}^{N-1} w_n^2 \cos(2\pi\omega_k \frac{n-n_0}{N}) \sin(2\pi\omega_l \frac{n-n_0}{N}) \\
B_{l,k}^{2,2} &= \sum_{n=0}^{N-1} w_n^2 \sin(2\pi\omega_k \frac{n-n_0}{N}) \sin(2\pi\omega_l \frac{n-n_0}{N}) \\
C_l^1 &= \sum_{n=0}^{N-1} x_n w_n \cos(2\pi\omega_l \frac{n-n_0}{N}) \\
C_l^2 &= -\sum_{n=0}^{N-1} x_n w_n \sin(2\pi\omega_l \frac{n-n_0}{N})
\end{aligned}
$$

Under the condition that every sinusoid has a different frequency, the matrix $\mathbf{B}$ cannot have two linear dependent rows implying a unique solution for $\mathbf{A}$.

The analysis of the computational complexity of this method in function of the number of samples N and number of partials K yields

- $\mathbf{B}$ is a $K \times K$ matrix of which each element is computed by a sum over N elements. This implies a complexity $\mathcal{O}(K^2 N)$.
- $\mathbf{C}$ is a vector of size K of which each element is computed by a sum over N elements. This implies a complexity $\mathcal{O}(KN)$.
- The solution of the linear set of equations requires a complexity $\mathcal{O}(K^3)$.

6.5.2 Efficient Computation of B

Several optimizations for the amplitude computation are proposed. The main computational burden comes from the construction of the matrix $\mathbf{B}$ and the solution of the system of linear equations which have a complexity $\mathcal{O}(K^2N)$ and $\mathcal{O}(K^3)$ respectively. Following derivation shows that this can be improved considerably. Using Eq. 6.7, $\mathbf{B}$ can be written in function of the frequency responses of the square window $Y(m)$ by writing

$$\begin{aligned} B_{l,k}^{1,1} &= \sum_{n=0}^{N-1} w_n^2 \cos(2\pi\omega_k \frac{n-n_0}{N}) \cos(2\pi\omega_l \frac{n-n_0}{N}) \\ &= \frac{1}{2} \sum_{n=0}^{N-1} w_n^2 \left[\cos(2\pi(\omega_k+\omega_l)\frac{n-n_0}{N}) + \cos(2\pi(\omega_k-\omega_l)\omega\frac{n-n_0}{N}) \right] \\ &= \frac{1}{2}(\Re(Y(\omega_k+\omega_l)) + \Re(Y(\omega_k-\omega_l))) \end{aligned} \tag{6.25}$$

In an analogue manner one obtains

$$\begin{aligned} B_{l,k}^{1,2} &= -\frac{1}{2}(\Im(Y(\omega_k+\omega_l)) + \Im((Y(\omega_k-\omega_l)) \\ B_{l,k}^{2,1} &= -\frac{1}{2}(\Im(Y(\omega_k+\omega_l)) - \Im(Y(\omega_k-\omega_l) \\ B_{l,k}^{2,2} &= -\frac{1}{2}(\Re(Y(\omega_k+\omega_l) - \Re(Y(\omega_k-\omega_l))) \end{aligned} \tag{6.26}$$

As was discussed previously, the frequency response of the squared Blackmann-Harris is bandlimited. Since the window is real and symmetric, its frequency response is also real and symmetric. This means that $\mathbf{B^{1,2}}$ and $\mathbf{B^{2,1}}$ only contain zeros. When defining two matrices ${\mathbf{Y}^-}_{l,k}$ and ${\mathbf{Y}^+}_{l,k}$ as

$$\begin{aligned} {\mathbf{Y}^-}_{k,l} &= Y(\omega_k-\omega_l) \\ {\mathbf{Y}^+}_{k,l} &= Y(\omega_k+\omega_l) \end{aligned} \tag{6.27}$$

one obtains

$$\begin{aligned} \mathbf{B^{1,1}} &= \frac{1}{2}(\mathbf{Y}^+ + \mathbf{Y}^-) \\ \mathbf{B^{2,2}} &= -\frac{1}{2}(\mathbf{Y}^+ - \mathbf{Y}^-) \end{aligned} \tag{6.28}$$

For example, when a non harmonic signal model with 4 partials is considered this results

in

$$\mathbf{Y}^- = \begin{bmatrix} Y(0) & Y(\omega_1 - \omega_0) & Y(\omega_2 - \omega_0) & Y(\omega_3 - \omega_0) \\ Y(\omega_0 - \omega_1) & Y(0) & Y(\omega_2 - \omega_1) & Y(\omega_2 - \omega_1) \\ Y(\omega_0 - \omega_2) & Y(\omega_1 - \omega_2) & Y(0) & Y(\omega_3 - \omega_2) \\ Y(\omega_0 - \omega_3) & Y(\omega_1 - \omega_3) & Y(\omega_2 - \omega_3) & Y(0) \end{bmatrix}$$

$$\mathbf{Y}^+ = \begin{bmatrix} Y(2\omega_0) & Y(\omega_1 + \omega_0) & Y(\omega_2 + \omega_0) & Y(\omega_3 + \omega_0) \\ Y(\omega_0 + \omega_1) & Y(2\omega_1) & Y(\omega_1 + \omega_2) & Y(\omega_1 + \omega_3) \\ Y(\omega_0 + \omega_2) & Y(\omega_1 + \omega_2) & Y(2\omega_2) & Y(\omega_3 + \omega_2) \\ Y(\omega_0 + \omega_3) & Y(\omega_1 + \omega_3) & Y(\omega_2 + \omega_3) & Y(2\omega_3) \end{bmatrix}$$

what can be simplified even further for a single harmonic sound source to

$$\mathbf{Y}^- = \begin{bmatrix} Y(0) & Y(\omega) & Y(2\omega) & Y(3\omega) \\ Y(-\omega) & Y(0) & Y(\omega) & Y(2\omega) \\ Y(-2\omega) & Y(-\omega) & Y(0) & Y(\omega) \\ Y(-3\omega) & Y(-2\omega) & Y(-\omega) & Y(0) \end{bmatrix}$$

$$\mathbf{Y}^+ = \begin{bmatrix} Y(0) & Y(\omega) & Y(2\omega) & Y(3\omega) \\ Y(\omega) & Y(2\omega) & Y(3\omega) & Y(4\omega) \\ Y(2\omega) & Y(3\omega) & Y(4\omega) & Y(5\omega) \\ Y(3\omega) & Y(4\omega) & Y(5\omega) & Y(6\omega) \end{bmatrix}$$

where ω denotes the fundamental frequency.

The crucial observation that has to be made is that the matrices $\mathbf{B^{1,1}}$ and $\mathbf{B^{2,2}}$ become band diagonal matrices when the sinusoidal components are sorted by their frequency. If the components are sorted, the frequency differences for elements close to the diagonal of $\mathbf{Y}^-$ are small and will fall in the main lobe of $Y(m)$ yielding a large value. For the elements far from the diagonal, the frequency difference is large and will fall outside the main lobe of $Y(m)$ resulting in zero values.

For $\mathbf{Y}^+$ a similar reasoning is applicable. In this case however the values $(k + l)\omega$ lie between zero and one. The main lobe of $Y(m)$ is therefore divided over the left and right hand side of the interval due to spectral replication. The smallest values result in significant matrix elements in the upper left corner. The largest values contribute to the lower right corner. As a result both $\mathbf{B^{1,1}}$ and $\mathbf{B^{2,2}}$ are band diagonal.

A typical method to solve a linear set of equations is the use of Gaussian elimination with backsubstitution. This method has a time complexity $\mathcal{O}(K^3)$. However, since the system is band diagonal, this method requires only a linear time complexity $\mathcal{O}(K)$. In addition, the space complexity can be reduced from $\mathcal{O}(K^2)$ to $\mathcal{O}(K)$ by storing only the values in the diagonal band. Therefore, a shifted matrix $\overleftarrow{B}_{l,k}$ is defined as

$$\overleftarrow{B}_{l,k} = B_{l,l+k-D} \tag{6.29}$$

where D denotes the number of diagonals that are stored around the main diagonal. Note that $l = 0, \ldots, L-1$ and $k = 0, \ldots, 2D$. For (l, k) couples resulting in an index outside $\mathbf{B}$, a zero value is returned. The amplitudes are computed directly from the

shifted versions of $\mathbf{B^{1,1}}$ and $\mathbf{B^{2,2}}$. By denoting this routine as $SOLVE$ this is written as

$$\begin{aligned}\mathbf{A^r} &= SOLVE(\overleftarrow{\mathbf{B^{1,1}}}, \mathbf{C^1}) \\ \mathbf{A^i} &= SOLVE(\overleftarrow{\mathbf{B^{2,2}}}, \mathbf{C^2})\end{aligned} \tag{6.30}$$

The impact on the space and time complexity is the following:

- Since $2D+1$ is significantly smaller than the number of partials K, the use of a shifted matrix $\overleftarrow{\mathbf{B}}$ reduces the space complexity from $\mathcal{O}(K^2)$ to $\mathcal{O}(K)$.
- By using an oversampled look-up table for the main lobe of $Y(m)$, each element of $\overleftarrow{\mathbf{B}}$ can be computed in constant time resulting in a complexity $\mathcal{O}(K)$ for the complete matrix.
- By solving the set of equations directly on $\overleftarrow{\mathbf{B}}$ and $\mathbf{C}$ the time complexity is reduced from $\mathcal{O}(K^3)$ to $\mathcal{O}(K)$.

6.5.3 Example for a Single Harmonic Sound Source

The band diagonal property of $\mathbf{B}$ for a single harmonic sound source with a fundamental frequency ω can be demonstrated in a very elegant way. In this case,the matrices ${\mathbf{Y}^-}_{l,k}$ and ${\mathbf{Y}^+}_{l,k}$ yield

$$\begin{aligned}{\mathbf{Y}^-}_{k,l} &= Y((k-l)\omega) \\ {\mathbf{Y}^+}_{k,l} &= Y((k+l)\omega)\end{aligned} \tag{6.31}$$

Since both $k\omega$ and $l\omega$ lie between zero and $\frac{1}{2}$, their difference lies between $-\frac{1}{2}$ and $\frac{1}{2}$. As was previously observed from Fig. 6.1, only values must be considered that lie within the bandwidth of the frequency response, meaning that

$$\frac{-\beta}{N} \leq (k-l)\omega \leq \frac{\beta}{N} \tag{6.32}$$

As a result only the values $k-l$ must be taken into account between $\lceil -\frac{\beta}{N\omega} \rceil$ and $\lfloor \frac{\beta}{N\omega} \rfloor$. Note that since k and l denote the row and column index of $\mathbf{Y}^-$, $k-l$ denotes the diagonal. This implies that only $2D+1$ diagonal bands must be considered with

$$D \quad = \quad \lfloor \frac{\beta}{N\omega} \rfloor \tag{6.33}$$

The number of diagonal bands is dependent on the bandwidth β of the frequency response, the number of samples N and the fundamental frequency ω. For instance, when the window length is chosen to be three periods, $N = \frac{3}{\omega}$, and knowing that $\beta = 8$ for the square Blackmann-Harris window, a value of 2 is obtained for D. This means that only the main diagonal and the first two upper and lower diagonals are relevant.

On the other hand, when considering the matrix $\mathbf{Y}^+$, the values for $(k+l)\omega$ lie between zero and one. The frequency response of the window is in this case divided over the left and right hand side of the interval. When considering the left half of the response, only significant values are obtained when $(k+l)\omega < \frac{\beta}{N}$, which yields for $N = \frac{3}{\omega}$ that $k + l \leq 2$. As a result, only significant values are obtained in the upper left corner. For the right hand side of the interval, the main lobe ranges from $1 - \beta/N$ to 1 yielding,

$$(k+l)\omega > 1 - \frac{\beta}{N} \tag{6.34}$$

$$\Rightarrow k + l > \frac{1}{\omega} - \frac{\beta}{3} \tag{6.35}$$

Note that $\frac{1}{\omega}$ corresponds with the maximal possible value of $k + l$ which corresponds with the lower right corner of the matrix. Therefore, only the skew diagonals close to these corners contain significant values. This derivation is illustrated by Figure 6.5.

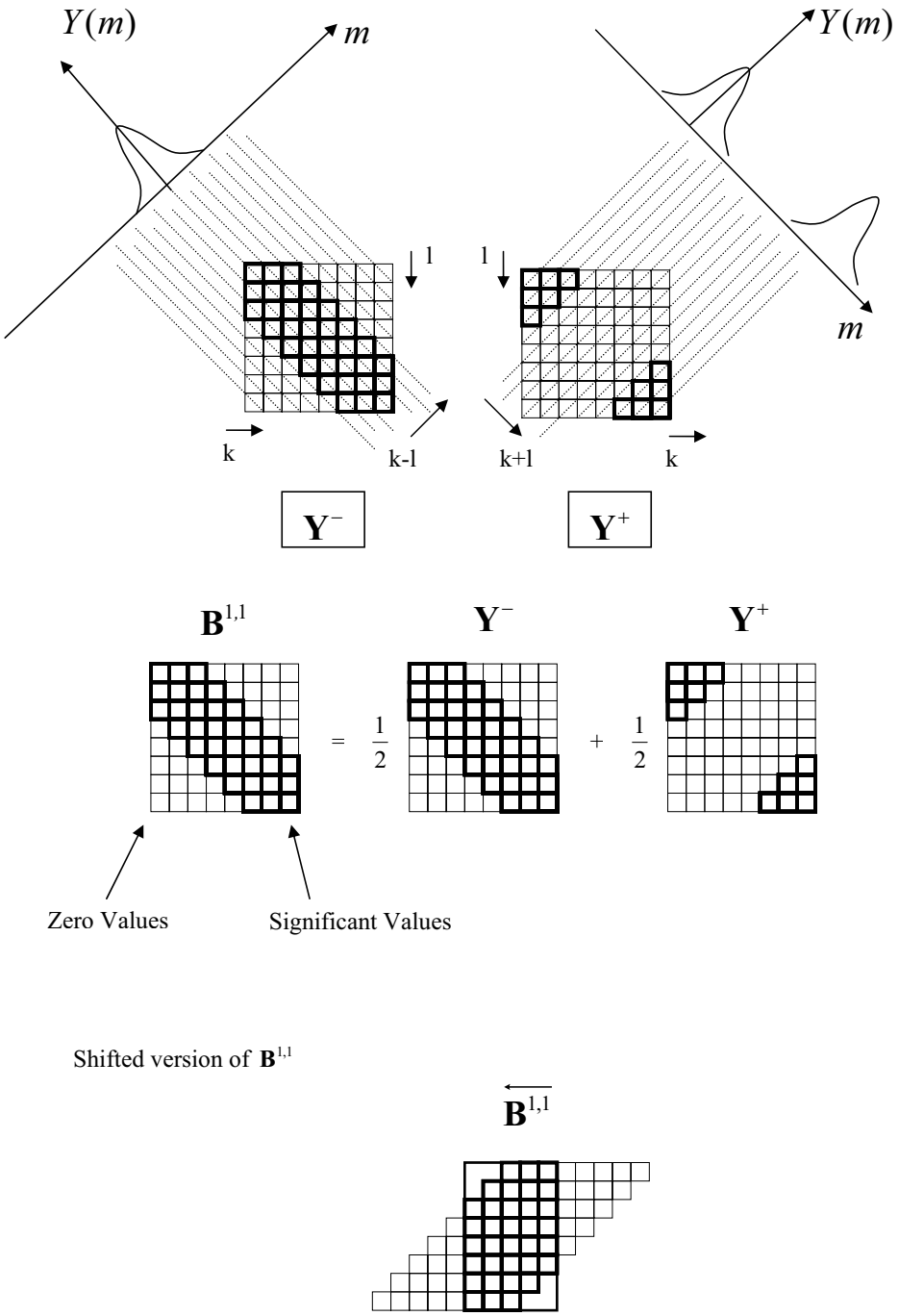


Fig. 6.5: *Example of a Band Diagonal Matrix B*

6.5.4 Efficient Computation of C

The results of the previous section move the bottleneck to the computation of $\mathbf{C}$ which has a complexity $\mathcal{O}(KN)$. When writing

$$\begin{aligned} C_l &= \sum_{n=0}^{N-1} x_n w_n \exp(2\pi i \omega_l \frac{n-n_0}{N}) \\ &= \sum_{n=0}^{N-1} \left[\frac{1}{N} \sum_{m=0}^{N-1} X_m \exp(2\pi i m \frac{n}{N}) \right] w_n \exp(2\pi i \omega_l \frac{n}{N}) \\ &= \frac{1}{N} \sum_{m=0}^{N-1} X_m W(m+\omega_l) \end{aligned} \tag{6.36}$$

the frequency response $W(m)$ can be computed as previously described by a scaled table lookup. Analogue to Eq. (6.18), only samples of the main lobe must be computed, yielding

$$C_l = \frac{1}{N} \sum_{m=m_{min}}^{m_{max}} X_m W(m+\omega_l) \tag{6.37}$$

By taking the real and imaginary part respectively, $\mathbf{C^1}$ and $\mathbf{C^2}$ are obtained.

Finally, the following computational complexity is obtained:

- The complexity for $\mathbf{C}$ is reduced from $\mathcal{O}(NK)$ to $\mathcal{O}(K)$. As a result, the complete amplitude estimation can be performed for a given spectrum X_m in linear time.
- The computation of this spectrum however has a complexity $\mathcal{O}(N \log N)$ which is the final complexity of the amplitude estimation.

This complete method is depicted in Figure 6.6.

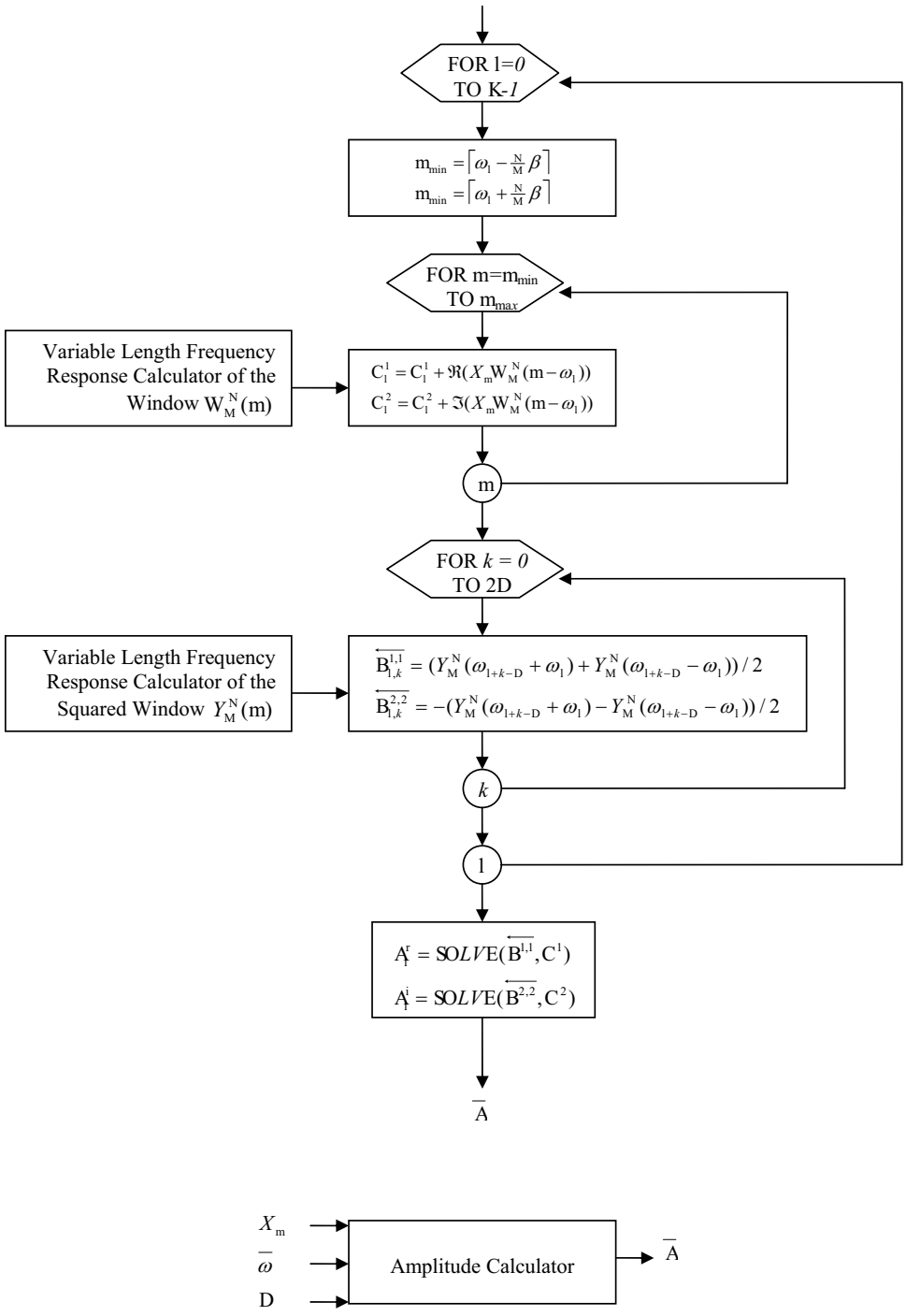


Fig. 6.6: *Efficient Amplitude Computation Algorithm*

σ	0	0.001	0.01	0.1
mean	1.56×10^{-8}	2.71×10^{-8}	1.10×10^{-6}	1.08×10^{-4}
MSE$\{A_1^r\}$	0.035×10^{-8}	2.38×10^{-8}	2.25×10^{-6}	2.20×10^{-4}
MSE$\{A_1^i\}$	0.067×10^{-8}	2.49×10^{-8}	2.33×10^{-6}	2.32×10^{-4}
MSE$\{A_2^r\}$	4.57×10^{-8}	7.00×10^{-8}	2.38×10^{-6}	2.36×10^{-4}
MSE$\{A_2^i\}$	1.11×10^{-8}	3.51×10^{-8}	2.37×10^{-6}	2.36×10^{-4}
MSE$\{A_3^r\}$	8.18×10^{-8}	10.7×10^{-8}	2.65×10^{-6}	2.47×10^{-4}
MSE$\{A_3^i\}$	2.17×10^{-8}	4.68×10^{-8}	2.49×10^{-6}	1.80×10^{-4}

Tab. 6.1: *Mean Square Error of the Estimated Amplitudes*

6.6 Analysis pre-processing

Before the amplitude computation, a pre-processing routine is executed. This routine consists of sorting the sinusoidal components by their frequencies in order to obtain a band diagonal matrix for $\mathbf{B}$. In addition, frequencies that are very close to one another are omitted since this would result in two exact rows in $\mathbf{B}$ making it a singular matrix.

Secondly, the pre-processing determines how many diagonals of the matrix $\mathbf{B}$ must be taken into account. This is done by counting the number of sinusoidal components that fall in the main lobe of each frequency response. The maximum number of components over all frequency responses yields the value for D.

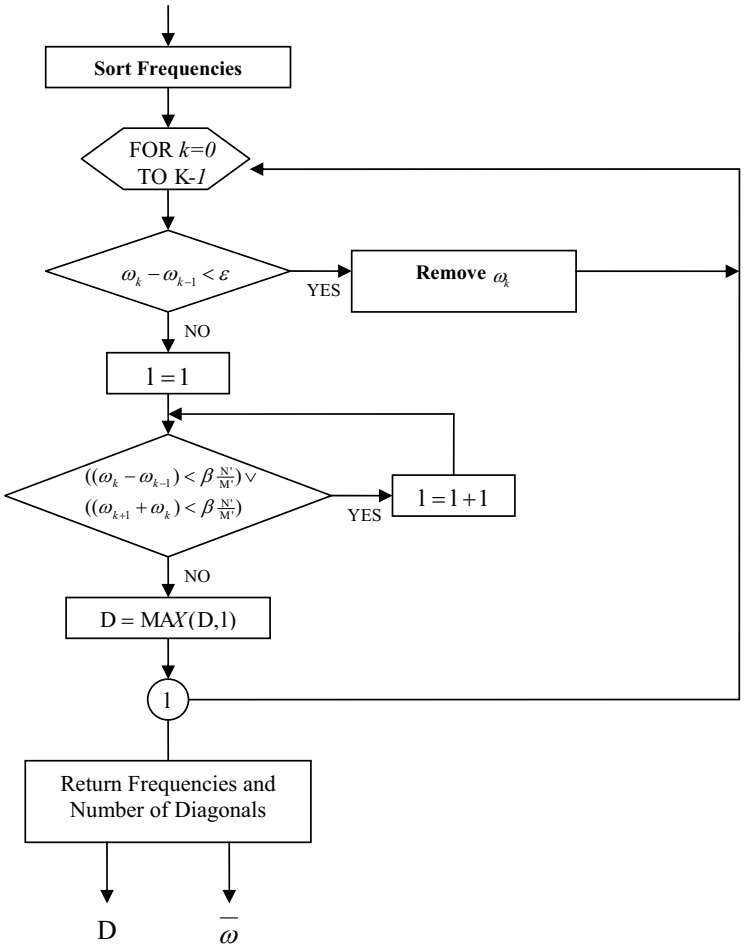
The amplitude estimation pre-processing routine is depicted in Figure 6.7.

6.7 Robustness

Least squares methods are widely used for amplitude estimation because they are simple and easy to implement. However from a statistical point of view, these estimators are suboptimal since the correlation in the noise is not taken into account. Techniques exist which partition the signal in a number of overlapping intervals in order to estimate the covariance matrix of the noise, resulting in *weighted least squares (WLS)* techniques [82]. However, because of the considerable computation cost, this is considered beyond the scope of this research.

In order to test the robustness of the least squares estimator the following experiment was performed. A windowed signal with a length of 200 samples was zero padded up to a length 256, consisting of three sinusoidal components that are relatively closely spaced $\bar{\omega} = [\frac{3}{N} \frac{7}{N} \frac{10}{N}]$ and with amplitudes $\bar{A} = [0.6 + 0.4i, 10 + 5i, 0.2 + 0.2i]$. These frequency responses are depicted in Figure 6.8 and were computed by using a look-up table containing the oversampled main lobe of a factor 10000.

By adding windowed white noise to the signal x_n with a standard deviation σ the noise sensitivity was tested. The resulta are shown in Figure 6.1. It can be seen that the estimator follows closely the *Cramer Rao Lower Bound* which is for the amplitude estimator $\frac{2\sigma^2}{N}$ [45]. This was also shown in [82].



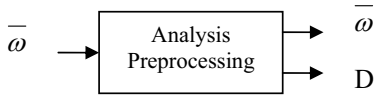


Fig. 6.7: *Preprocessing Routine Before Amplitude Computation*

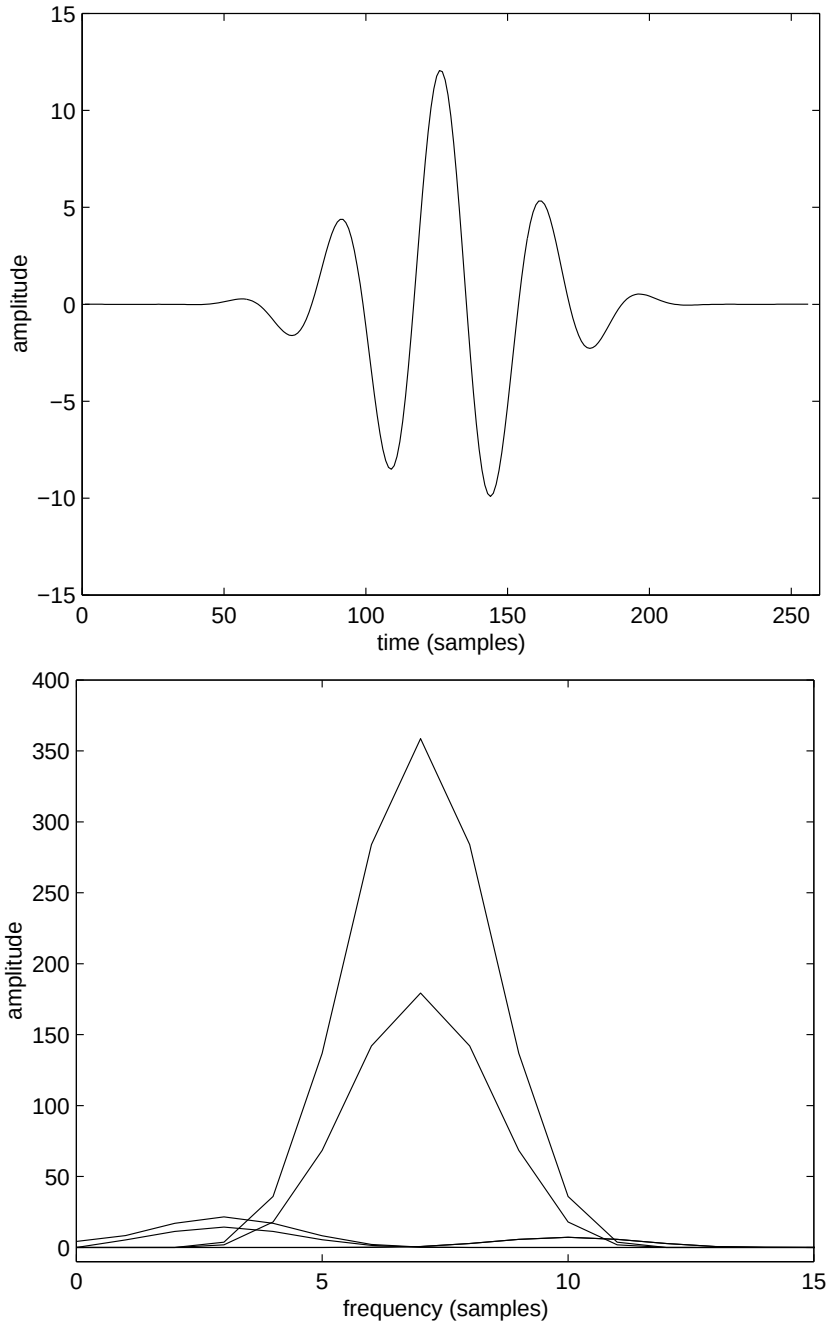


Fig. 6.8: *Signal and Frequency Responses for the Experiment.*

6.8 Conclusion

Least squares amplitude estimators are frequently used in sinusoidal analysis. Most applications currently use the iterative methods because of their computational efficiency, but since these methods are "greedy", they yield a suboptimal solution. The simultaneous estimation of the frequencies is used not very frequently because of its significantly higher computational complexity. The results in this chapter show that the simultaneous estimation can be realized in the same complexity as for the iterative estimation. This makes the use of this optimized simultaneous estimator superior over the currently known techniques. To the best of our knowledge, this is the first time that this optimization is realized.

CHAPTER 7

Frequency Optimization: From $\mathcal{O}(N^3)$ to $\mathcal{O}(N \log N)$

7.1 Chapter Overview

In the previous chapter, an efficient method was proposed to compute the amplitudes and phases for a given set of frequencies. In this chapter, methods are presented that determine the initial values of the frequencies and optimize these frequencies with respect to the error criterium in an iterative manner. To the best of our knowledge, only two different methods are currently known for frequency optimization in the case of overlapping frequency responses being

- Local quadratic optimization of the error function [99]
- Linearisation of the signal model [15, 97]

Both methods optimize the frequencies while the amplitudes are kept constant. The complete sinusoidal analysis system applies the amplitude computation and frequency optimization alternately so that they both converge to their optimal values.

By using the same methodology as in the previous chapter, the computational complexity of the frequency optimization can again be optimized from $\mathcal{O}(K^2N)$ to $\mathcal{O}(N \log(N))$. This is shown for both the local quadratic approximation strategy (7.3) and the model linearization strategy (7.4).

7.2 Initial Frequency Values

The error is very non linear in function of the frequencies and can therefore get stuck in local minima. Therefore, the initial values of the frequencies plays an important role. Different methods can be used

- **(Multi)pitch estimation:** For the harmonic signal model (Eq. 7.16), any (multi) pitch estimator can be used to compute a number of pitches from the original signal [86]. From these pitches, sets of frequencies can be produced depending on prior knowledge of the sound source.
- **Peak picking.** For the non harmonic model, individual local maxima can be detected in the sampled spectrum obtained by an FFT. Note that in that case the analysis window must be sufficiently large.
- **Prior knowledge.** In some cases, the frequencies and / or pitches can be read from an external file such as a musical score or an annotation of the signal.

7.3 Local Quadratic Approximation

The problem of minimizing continuous differential functions of many variables is widely studied and any of the conventional methods can be applied [4]. Many of these methods make a locally linear or quadratic approximation of the error function.

A second order Taylor expansion of the multidimensional error function $\chi(\omega, A)$ around a vector $\bar{v}$ is given by

$$\chi(\bar{\omega}, A) \approx C + (\bar{\omega} - \bar{v})^T \mathbf{h}|_v + \frac{1}{2}(\bar{\omega} - \bar{v})^T \mathbf{H}|_v (\bar{\omega} - \bar{v}) \tag{7.1}$$

where $\mathbf{h}|_\omega$ and $\mathbf{H}|_\omega$ denote respectively the gradient and Hessian of $\chi(\omega, \bar{A})$ defined by

$$\begin{aligned} h_k|_\omega &\equiv \frac{\partial \chi(\bar{\omega}; \bar{A})}{\partial \bar{\omega}_k} \\ H_{l,k}|_{\bar{\omega}} &\equiv \frac{\partial \chi(\bar{\omega}; \bar{A})}{\partial \bar{\omega}_l \partial \bar{\omega}_k} \end{aligned}$$

yielding

$$\mathbf{h}|_\omega = \mathbf{h}|_v + \mathbf{H}|_v (\bar{\omega} - \bar{v}) \tag{7.2}$$

$$\mathbf{H}|_\omega = \mathbf{H}|_v \tag{7.3}$$

If the series is developed at the minimum of the function, it follows that

$$\mathbf{h}|_v = 0 \tag{7.4}$$

from which is derived that

$$v = \omega - \mathbf{H}|_\omega^{-1} \mathbf{h}|_\omega \tag{7.5}$$

This method is called Newton optimization. From a rough estimation of the frequencies $\bar{\omega}^{(0)}$, these values are optimized iteratively using

$$\bar{\omega}^{(r)} = \bar{\omega}^{(r-1)} - \mathbf{H}|_{\bar{\omega}^{(r-1)}}^{-1} \mathbf{h}|_{\bar{\omega}^{(r-1)}} \tag{7.6}$$

where the superscript $^{(r)}$ denotes the index of the iteration. Because of the large computational cost of the inversion of the Hessian matrix, quasi-Newton have been developed which construct the inverse matrix iteratively [4]. In this chapter, it is shown that the computational efficiency of Newton optimization can be improved considerably.

A second method is the gradient descent method for which the frequencies are optimized using

$$\bar{\omega}^{(r)} = \bar{\omega}^{(r-1)} - \eta \mathbf{h}|_{\omega^{r-1}} \tag{7.7}$$

where parameter η is called the learning rate. Many different variations have been proposed such as using a momentum term, line search algorithms, using conjugate gradients and scaled conjugate gradients [4].

Finally, following termination criteria can be used to stop the iterative optimization of the frequencies, such as

- stop after a fixed number of iterations
- stop after fixed computation time
- stop when error function drops below a specified value
- stop when the error change drops below a specified value
- stop when error measure starts to increase.

7.3.1 Efficient Computation of the Gradient

An efficient method is developed which allows to compute each element of the gradient in constant time yielding $\mathcal{O}(K)$ for all elements. However, the FFT of the noise residual R_m is required implying an $\mathcal{O}(N \log(N))$ complexity. This noise residual r_n is the difference between the signal x_n and the model $\tilde{x}_n$. The gradient of the error function is written as

$$
\begin{aligned}
& \frac{\partial \chi(\bar{\omega}; \bar{A})}{\partial \omega_l} \\
= \; & \frac{\partial}{\partial \omega_l} \sum_{n=0}^{N-1} \left[x_n - w_n \frac{1}{2} \sum_{k=0}^{K-1} \left(A_k \exp(2\pi i \omega_k \tfrac{n-n_0}{N}) + A_k^* \exp(-2\pi i \omega_k \tfrac{n-n_0}{N}) \right) \right]^2 \\
= \; & \sum_{n=0}^{N-1} 2 \left[x_n - w_n \frac{1}{2} \sum_{k=0}^{K-1} \left(A_k \exp(2\pi i \omega_k \tfrac{n-n_0}{N}) + A_k^* \exp(-2\pi i \omega_k \tfrac{n-n_0}{N}) \right) \right] \\
& \left(-w_n \frac{1}{2} A_l \exp(2\pi i \omega_l \tfrac{n-n_0}{N}) 2\pi i \tfrac{n-n_0}{N} - w_n \frac{1}{2} A_l^* \exp(-2\pi i \omega_l \tfrac{n-n_0}{N}) \left(-2\pi i \tfrac{n-n_0}{N}\right) \right) \\
= \; & \sum_{n=0}^{N-1} \left[\sum_{m=0}^{N-1} \tfrac{n-n_0}{N} R_m \exp(2\pi i m \tfrac{n-n_0}{N}) \right] \\
& \left(-w_n A_l \exp(2\pi i \omega_l \tfrac{n-n_0}{N}) 2\pi i \tfrac{n-n_0}{N} + w_n A_l^* \exp(-2\pi i \omega_l \frac{n-n_0}{N}) 2\pi i \tfrac{n-n_0}{N} \right) \\
= \; & \frac{1}{N} \sum_{m=0}^{N-1} R_m \left[-A_l \sum_{n=0}^{N-1} w_n 2\pi i \tfrac{n-n_0}{N} \exp(2\pi i (m+\omega_l) \tfrac{n-n_0}{N}) \right. \\
& \left. + A_l^* \sum_{n=0}^{N-1} w_n 2\pi i \tfrac{n-n_0}{N} \exp(2\pi i (m-\omega_l) \tfrac{n-n_0}{N}) \right] \\
= \; & \frac{1}{N} \sum_{m=0}^{N-1} R_m (A_l^* W'(m-\omega_l) - A_l W'(m+\omega_l)) \\
= \; & \frac{2}{N} \sum_{m=0}^{N-1} \Re \left(R_m A_l^* W'(m-\omega_l) \right)
\end{aligned}
$$

Note that Eq. 6.7 was used and that the sum over m can be limited to the interval around ω_l, as can be computed from the bandwidth β of $W'(m)$ as given in Eq. (6.18). Here a similar computational gain is obtained as for the vector $\mathbf{C}$ in the previous chapter.

7.3.2 Efficient Computation of the Hessian

It is derived that the Hessian matrix for the non harmonic signal model is band diagonal and that each element of the Hessian is computed in constant time. The computation of all band diagonal elements is computed in $\mathcal{O}(K)$ time. Since the fourier transform of the noise residual R_m is required the total complexity is $\mathcal{O}(N \log(N))$.

The Hessian for the non harmonic model is computed

$$
\begin{aligned}
&\frac{\partial \chi(\bar{A};\bar{\omega})}{\partial\omega_p\partial\omega_l} \\
&= \frac{\partial}{\partial\omega_p}\frac{\partial}{\partial\omega_l}\sum_{n=0}^{N-1}\left[x_n - w_n\frac{1}{2}\sum_{k=0}^{K-1}\left(A_k\exp(2\pi i\omega_k\tfrac{n-n_0}{N}) + A_k^*\exp(-2\pi i\omega_k\tfrac{n-n_0}{N})\right)\right]^2 \\
&= \frac{\partial}{\partial\omega_p}\sum_{n=0}^{N-1} 2\left[x_n - w_n\frac{1}{2}\sum_{k=0}^{K-1}\left(A_k\exp(2\pi i\omega_k\tfrac{n-n_0}{N}) + A_k^*\exp(-2\pi i\omega_k\tfrac{n-n_0}{N})\right)\right] \\
&\quad \left(-w_n\frac{1}{2}A_l\exp(2\pi i\omega_l\tfrac{n-n_0}{N})2\pi i\tfrac{n-n_0}{N} - w_n\frac{1}{2}A_l^*\exp(-2\pi i\omega_l\tfrac{n-n_0}{N})\,(-2\pi i\tfrac{n-n_0}{N})\right)
\end{aligned}
$$

This partial derivative results in the sum of two terms. The first term yields

$$
\begin{aligned}
&\sum_{n=0}^{N-1} 2\left[x_n - w_n\frac{1}{2}\sum_{k=0}^{K-1}\left(A_k\exp(2\pi i\omega_k\tfrac{n-n_0}{N}) + A_k^*\exp(-2\pi i\omega_k\tfrac{n-n_0}{N})\right)\right] \\
&\frac{\partial}{\partial\omega_p}\left(-w_n\frac{1}{2}A_l\exp(2\pi i\omega_l\tfrac{n-n_0}{N})2\pi i\tfrac{n-n_0}{N} - w_n\frac{1}{2}A_l^*\exp(-2\pi i\omega_l\tfrac{n-n_0}{N})\,(-2\pi i\tfrac{n-n_0}{N})\right) \\
= &\sum_{n=0}^{N-1}\left[x_n - w_n\frac{1}{2}\sum_{k=0}^{K-1}\left(A_k\exp(2\pi i\omega_k\tfrac{n-n_0}{N}) + A_k^*\exp(-2\pi i\omega_k\tfrac{n-n_0}{N})\right)\right] \\
&\delta_{lp}\Big(-w_nA_p\exp(2\pi i\omega_p\tfrac{n-n_0}{N})\,(2\pi i\tfrac{n-n_0}{N})^2 \\
&-w_nA_p^*\exp(-2\pi i\omega_p\tfrac{n-n_0}{N})\,(-2\pi i\tfrac{n-n_0}{N})^2\Big) \\
= &\delta_{lp}\sum_{n=0}^{N-1}\left[\frac{1}{N}\sum_{m=0}^{N-1}R_m\exp(2\pi im\tfrac{n-n_0}{N})\right] \\
&\Big(-w_nA_p\exp(2\pi i\omega_p\tfrac{n-n_0}{N})\,(2\pi i\tfrac{n-n_0}{N})^2 \\
&-\,w_nA_p^*\exp(-2\pi i\omega_p\tfrac{n-n_0}{N})\,(2\pi i\tfrac{n-n_0}{N})^2\Big) \\
= &-\delta_{lp}\frac{1}{N}\sum_{m=0}^{N-1}R_m\left[A_p\sum_{n=0}^{N-1}w_n\,(2\pi i\tfrac{n-n_0}{N})^2\exp(2\pi i(m+\omega_p)\tfrac{n-n_0}{N}) + \right. \\
&\left. A_p^*\sum_{n=0}^{N-1}w_n\,(2\pi i\tfrac{n-n_0}{N})^2\exp(2\pi i(m-\omega_p)\tfrac{n-n_0}{N}\right] \\
= &-\delta_{lp}\frac{1}{N}\sum_{m=0}^{N-1}R_m(A_pW''(m+\omega_p) + A_p^*W''(m-\omega_p)) \\
= &-\delta_{lp}\frac{2}{N}\sum_{m=m_{min}}^{m_{max}}\Re\left(R_mA_p^*W''(m-\omega_p)\right) \qquad (7.8)
\end{aligned}
$$

Next, the second term is computed, resulting in

$$
\begin{aligned}
& \sum_{n=0}^{N-1} \left(-w_n \frac{1}{2} A_l \exp(2\pi i \omega_l \tfrac{n-n_0}{N}) 2\pi i \tfrac{n-n_0}{N} - w_n \frac{1}{2} A_l^* \exp(-2\pi i \omega_l \tfrac{n-n_0}{N}) \, (-2\pi i \tfrac{n-n_0}{N}) \right) \\
& \frac{\partial}{\partial \omega_p} 2 \left[x_n - w_n \frac{1}{2} \sum_{k=0}^{K-1} (A_k \exp(2\pi i \omega_k \tfrac{n-n_0}{N}) + A_k^* \exp(-2\pi i \omega_k \tfrac{n-n_0}{N}) \right] \\
= & \sum_{n=0}^{N-1} \left(-w_n \frac{1}{2} A_l \exp(2\pi i \omega_l \tfrac{n-n_0}{N}) 2\pi i \tfrac{n-n_0}{N} - w_n \frac{1}{2} A_l^* \exp(-2\pi i \omega_l \tfrac{n-n_0}{N}) \, (-2\pi i \tfrac{n-n_0}{N}) \right) \\
& \left(-w_n A_p \exp(2\pi i \omega_p \tfrac{n-n_0}{N}) 2\pi i \tfrac{n-n_0}{N} - w_n A_p^* \exp(-2\pi i \omega_p \tfrac{n-n_0}{N}) \, (-2\pi i \tfrac{n-n_0}{N}) \right) \\
= & \frac{1}{2} \sum_{n=0}^{N-1} A_l A_p w_n^2 \, (2\pi i \tfrac{n-n_0}{N})^2 \exp(2\pi i (\omega_l + \omega_p) \tfrac{n-n_0}{N}) - \\
& \frac{1}{2} \sum_{n=0}^{N-1} A_l A_p^* w_n^2 \, (2\pi i \tfrac{n-n_0}{N})^2 \exp(2\pi i (\omega_l - \omega_p) \tfrac{n-n_0}{N}) - \\
& \frac{1}{2} \sum_{n=0}^{N-1} A_l^* A_p w_n^2 \, (2\pi \tfrac{n-n_0}{N})^2 \exp(-2\pi i (\omega_l - \omega_p) \tfrac{n-n_0}{N}) + \\
& \frac{1}{2} \sum_{n=0}^{N-1} A_l^* A_p^* w_n^2 \, (2\pi i \tfrac{n-n_0}{N})^2 \exp(-2\pi i (\omega_l + \omega_p) \tfrac{n-n_0}{N}) \\
= & \frac{1}{2} \left(A_l A_p Y''(\omega_l + \omega_p) - A_l A_p^* Y''(\omega_l - \omega_p) \right. \\
& \left. - A_l^* A_p Y''(-\omega_l + \omega_p) + A_l^* A_p^* Y(-\omega_l - \omega_p) \right) \\
= & \Re(A_l A_p Y''(\omega_l + \omega_p) - \Re(A_l A_p^* Y''(\omega_l - \omega_p))
\end{aligned} \tag{7.9}
$$

using Eq. 6.7. Here, a same conclusion can be drawn as for the matrix $\mathbf{B}$ which was used for the computation of the amplitudes. The first term of the Hessian computation results only in nonzero values for the diagonal elements of the Hessian. The second term also produces only significant values around the main diagonal elements since the frequency differences fall in the main lobe of $Y''(m)$. As a result, the Hessian is a band diagonal matrix, implying that only the elements close to the diagonal must be computed and can be stored in a shifted form, denoted.

$$\overleftarrow{H}_{l,k} = H_{l,l+k-D} \tag{7.10}$$

Again, the gaussian elimination routine $SOLVE$, adapted to this shifted matrix, allows to compute the frequency optimization step in linear time yielding

$$\omega^{(r+1)} = \omega^{(r)} - SOLVE(\overleftarrow{\mathbf{H}}, \mathbf{h}) \tag{7.11}$$

The complete frequency optimization routine is depicted in Figure 7.1.

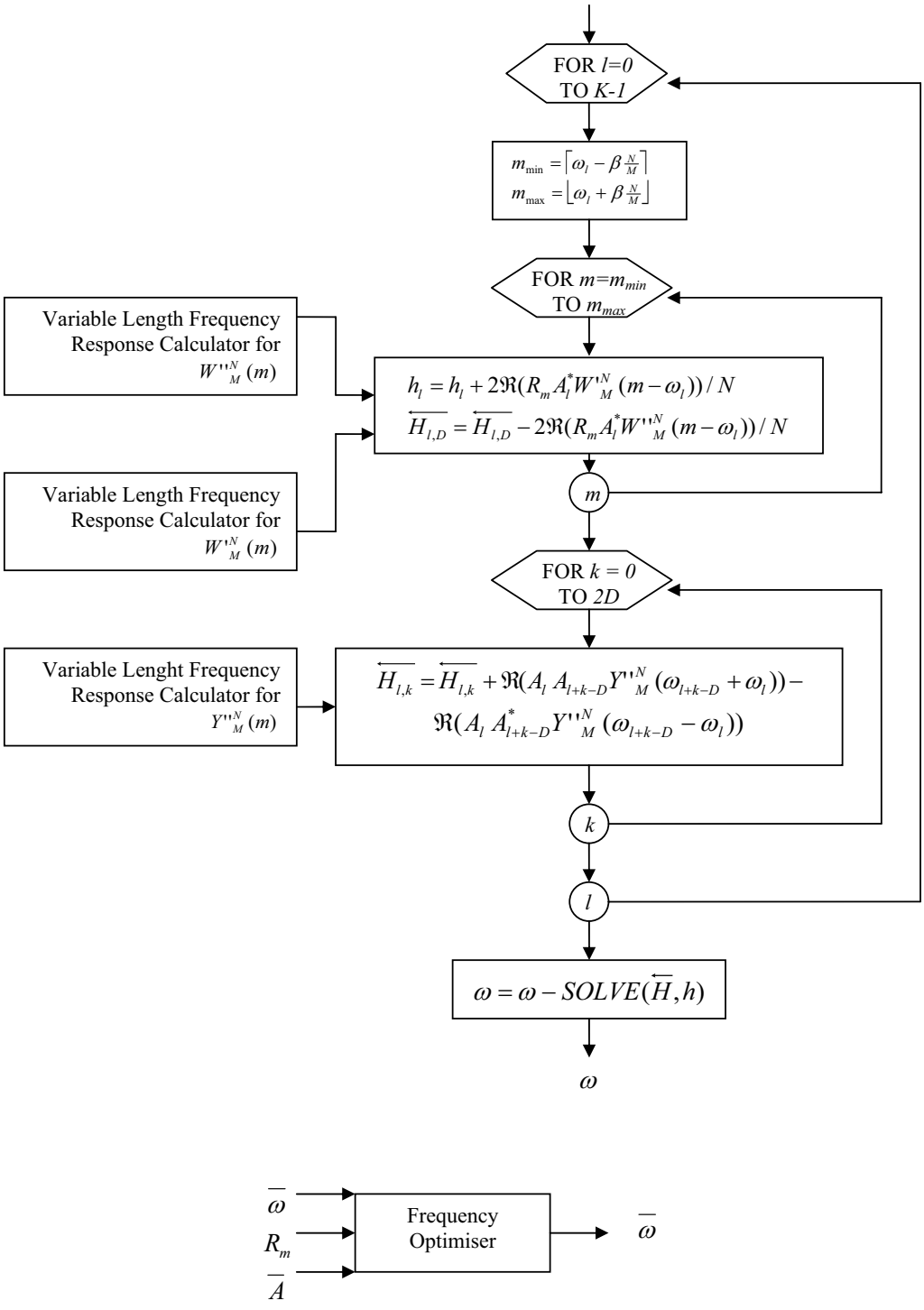


Fig. 7.1: *Frequency Optimiser for Non-Harmonic Model*

7.4 Model Linearization

In [15], a second frequency optimization method is proposed for a given set of amplitudes A_k and a rough estimation of the frequencies $\hat{\omega}_k$. A first order Taylor expansion of the frequency response around the roughly estimated frequencies $\hat{\omega}_k$ yields

$$W(m-\omega_k) \approx W(m-\hat{\omega}_k) + W'(m-\hat{\omega}_k)(\hat{\omega}_k - \omega_k) \tag{7.12}$$

which can be expressed in the time domain as

$$\begin{aligned} w_n \exp(2\pi i \omega_k \tfrac{n-n_0}{N}) &\approx w_n \exp(2\pi i \bar{\omega}_k \tfrac{n-n_0}{N}) + \\ & \quad w_n 2\pi i \tfrac{n-n_0}{N} \exp(2\pi i \bar{\omega}_k \tfrac{n-n_0}{N})(\hat{\omega}_k - \omega_k) \end{aligned}$$

yielding

$$\begin{aligned} \chi(\omega; A) = \Bigg(& x_n - \frac{1}{2} w_n \sum_{k=0}^{K-1} \left(A_k \exp(2\pi i \hat{\omega}_k \tfrac{n-n_0}{N}) + A_k^* \exp(-2\pi i \hat{\omega}_k \tfrac{n-n_0}{N}) \right. \\ & \left. + 2\pi i \tfrac{n-n_0}{N} \left[A_k \exp(2\pi i \hat{\omega}_k \tfrac{n-n_0}{N}) - A_k^* \exp(-2\pi i \hat{\omega}_k \tfrac{n-n_0}{N}) \right] (\hat{\omega}_k - \omega_k) \right) \Bigg) \end{aligned}$$

This error function depends linearly on the difference with the true frequency value ω_k. By putting the partial derivative with respect to $(\tilde{\omega}_l - \omega_l)$ to zero

$$\frac{\partial \sum_n (x_n - \tilde{x}_n)^2}{\partial(\hat{\omega}_l - \omega_l)} = 0 \tag{7.13}$$

and using $r_n = x_n - \tilde{x}_n$, one obtains

$$\begin{aligned} 2 \sum_n^{N-1} \Bigg(r_n - \frac{1}{2} 2\pi i \tfrac{n-n_0}{N} w_n \sum_{k=0}^{K-1} \left[A_k \exp(2\pi i \hat{\omega}_k \tfrac{n-n_0}{N}) - A_k^* \exp(-2\pi i \hat{\omega}_k \tfrac{n-n_0}{N}) \right] (\hat{\omega}_k - \omega_k) \Bigg) \\ \left(-\frac{1}{2} 2\pi i \tfrac{n-n_0}{N} w_n \left[A_l \exp(2\pi i \hat{\omega}_l \tfrac{n-n_0}{N}) - A_l^* \exp(-2\pi i \hat{\omega}_l \tfrac{n-n_0}{N}) \right] \right) = 0 \end{aligned}$$

This expression is written in a matrix form resulting in

$$\mathbf{GA} = \mathbf{g} \tag{7.14}$$

with

$$
\begin{aligned}
A_l &= \hat{\omega}_l - \omega_l \\
g_l &= \sum_n^{N-1} r_n w_n 2\pi i \tfrac{n-n_0}{N} \left(A_l \exp(2\pi i \hat{\omega}_l \tfrac{n-n_0}{N}) - A_l^* \exp(-2\pi i \omega_l \tfrac{n-n_0}{N})\right) \\
&= \sum_n^{N-1} \left[\tfrac{n-n_0}{N} \sum_m^{N-1} R_m \exp(2\pi i m \tfrac{n-n_0}{N}) \right] \\
&\quad w_n 2\pi i \tfrac{n-n_0}{N} \left(A_l \exp(2\pi i \omega_l \tfrac{n-n_0}{N}) - A_l^* \exp(-2\pi i \omega_l \tfrac{n-n_0}{N})\right) \\
&= \frac{1}{N} \sum_m^{N-1} R_m (A_l W'(m+\omega_l) + A_l^* W'(m-\omega_l)) \\
&= \frac{2}{N} \sum_m^{N-1} \Re(R_m A_l^* W'(m-\omega_l)) \\
G_{l,k} &= 2 \sum_{n=0}^{N-1} \left[-w_n \frac{1}{2} 2\pi i \tfrac{n-n_0}{N} \left(A_k \exp(2\pi i \omega_k \tfrac{n-n_0}{N}) - A_k^* \exp(-2\pi i \omega_k \tfrac{n-n_0}{N})\right) \right] \\
&\quad \left[-w_n \frac{1}{2} 2\pi i \tfrac{n-n_0}{N} \left(A_l \exp(2\pi i \omega_l \tfrac{n-n_0}{N}) - A_l^* \exp(-2\pi i \omega_l \tfrac{n-n_0}{N})\right) \right] \\
&= \frac{1}{2} \big[A_k A_l Y''(\omega_k + \omega_l) - A_k A_l^* Y''(\omega_k + \omega_l) \\
&\quad - A_k^* A_l Y''(\omega_k - \omega_l) + A_k^* A_l^* Y''(\omega_k + \omega_l) \big] \\
&= \Re(A_k A_l Y''(\omega_k + \omega_l) - A_k A_l^* Y''(\omega_k - \omega_l))
\end{aligned} \tag{7.15}
$$

When comparing the vector $\mathbf{g}$ with $\mathbf{h}$ one observes that these vectors are identical. Also $\mathbf{G}$ and $\mathbf{H}$ are very similar and differ only by their diagonal values. The speed of convergence of both methods is compared in the next section.

7.5 Comparison of the Speed of Convergence

The speed of convergence is compared for the local quadratic approximation method and the model linearisation method. A short time signal with length $M' = 200$ and zero padded up to length $N' = 256$ was synthesized containing three partials with complex amplitudes $\bar{A} = [1+i, 1+i, 1+i]$ and frequencies $\bar{\omega} = [\frac{4}{N}, \frac{7}{N}, \frac{10}{N}]$. The initial frequency values were chosen relatively far away from the optimal values at $\bar{\omega} = [\frac{3}{N}, \frac{8}{N}, \frac{12}{N}]$. However these initial values are chosen so that that the frequency responses of the initial frequency and the optimal frequency still overlap. Figure 7.2 shows the evolution of the frequency values for each iteration. In a first experiment, the exact amplitudes were assumed to be known. This resulted in a very fast convergence for all partials which was obtained in about 4 or 5 iterations. For the second experiment where also the amplitudes are estimated in each iteration, convergence is considerably slower, resulting in minimum 30 iterations. It must be noted however that in practical cases the initial frequency values is much nearer to the optimal values requiring less iterations.

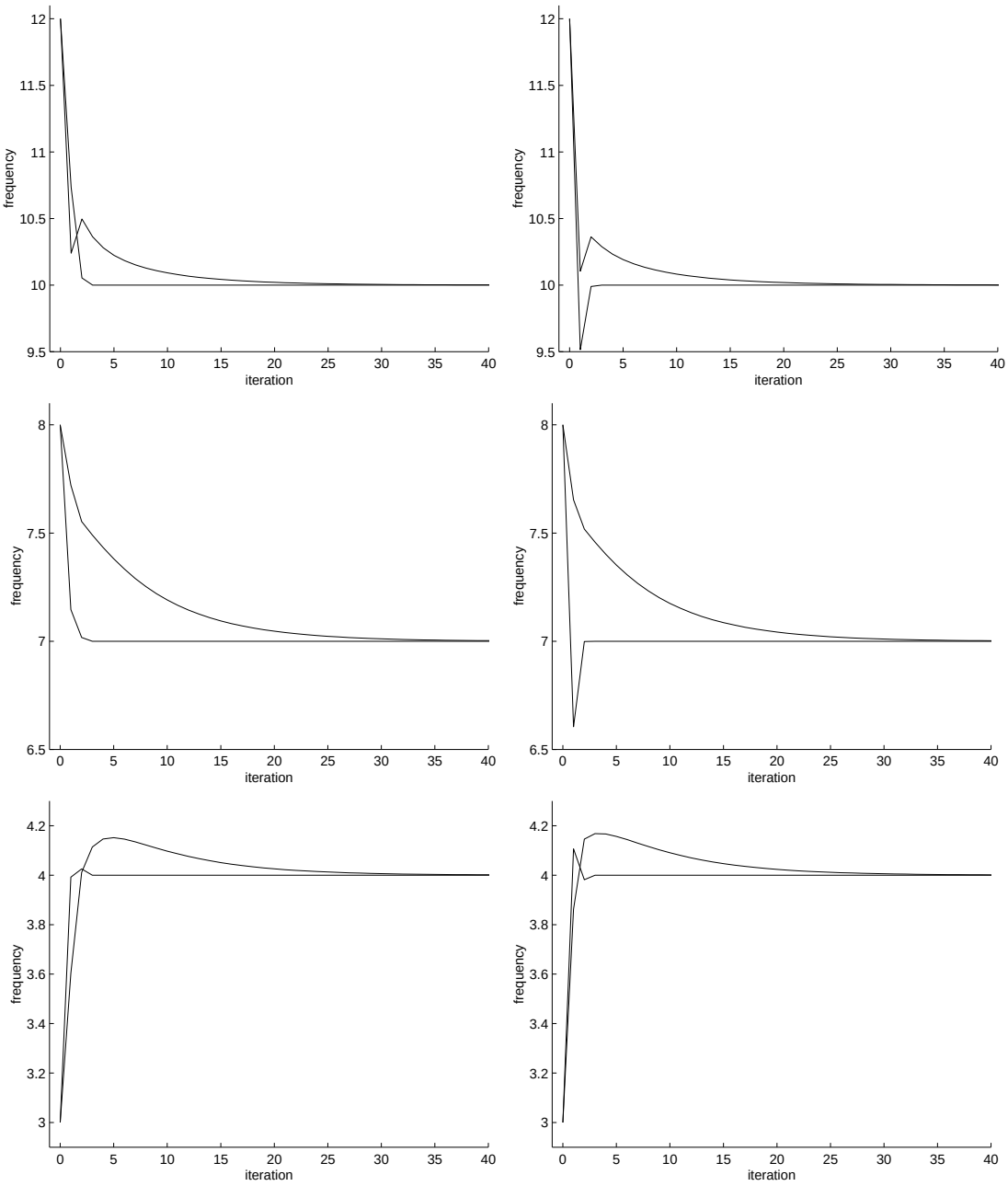


Fig. 7.2: *Convergence for frequency optimization.* **Left:** *Local quadratic approximations,* **Right:** *Model linearization*

Note that for this experiment, no significant difference is observed between both frequency optimization methods.

7.6 Harmonic Signal Model

In previous derivation, no prior relationship between the frequencies was imposed. On the other hand, when the sound sources are known to be pitched, a model containing a harmonic series can be used which is written as

$$\tilde{x}_n = w_n \frac{1}{2} \sum_{k=0}^{K-1} \sum_{q=1}^{Q_k-1} \left(A_{k,q} \exp(2\pi i q \omega_k \frac{n-n_0}{N}) + A_{k,q}^* \exp(-2\pi i q \omega_k \frac{n-n_0}{N}) \right) + r_n \quad (7.16)$$

In this case, the model consists of K sources each modelled by Q_k harmonic components. By contrast to the previous method, only the fundamental frequencies are optimized.

7.6.1 Gradient for Harmonic Model

By following a very similar derivation as for Eq. (7.8) the gradient of the error function of the harmonic model yields

$$\frac{\partial \chi(\bar{\omega}; \bar{A})}{\partial \omega_l} = \frac{2}{N} \sum_{q=1}^{Q_l-1} \sum_{m=m_{min}}^{m_{max}} \Re \left(R_m q A_{l,q}^* W'(m - q\omega_l) \right) \quad (7.17)$$

7.6.2 Hessian for Harmonic Model

Analogue to Eq. (7.8) and Eq. (7.9) the Hessian for the harmonic model results in two terms. The first term results in

$$-\delta_{lp} \frac{2}{N} \sum_{q=1}^{Q_l-1} \sum_{m=m_{min}}^{m_{max}} \Re \left(R_m q^2 A_{p,q}^* W''(m - q\omega_p) \right) \quad (7.18)$$

which again produces only non zero values at the diagonal of the Hessian matrix.

For the second term one obtains

$$\begin{aligned}
& \frac{1}{2}\sum_{n=0}^{N-1}\left(w_n 2\pi i\frac{n-n_0}{N}\sum_{q=1}^{Q_p-1} q\left[A_{p,q}\exp(2\pi iq\omega_p\frac{n-n_0}{N}) + A^*_{p,q}\exp(-2\pi iq\omega_p\frac{n-n_0}{N})\right]\right) \\
& \left(w_n 2\pi i\frac{n-n_0}{N}\sum_{r=1}^{Q_l-1} r\left[A_{l,r}\exp(2\pi ir\omega_l\frac{n-n_0}{N}) + A^*_{l,r}\exp(-2\pi ir\omega_l\frac{n-n_0}{N})\right]\right) \\
= & \frac{1}{2}\sum_{n=0}^{N-1}\left(w_n^2(2\pi i\frac{n-n_0}{N})^2\sum_{r=1}^{Q_l-1}\sum_{q=1}^{Q_p-1} qr\left[A_{p,q}\exp(2\pi iq\omega_p\frac{n-n_0}{N}) + A^*_{p,q}\exp(-2\pi iq\omega_p\frac{n-n_0}{N})\right]\right. \\
& \left.\left[A_{l,r}\exp(2\pi ir\omega_l\frac{n-n_0}{N}) + A^*_{l,r}\exp(-2\pi ir\omega_l\frac{n-n_0}{N})\right]\right) \\
= & \frac{1}{2}\sum_{n=0}^{N-1}\left(w_n^2(2\pi i\frac{n-n_0}{N})^2\sum_{r=1}^{Q_l-1}\sum_{q=1}^{Q_p-1} qr\left[A_{p,q}A_{l,r}\exp(2\pi i(q\omega_p + r\omega_l)\frac{n-n_0}{N})\right.\right. \\
& A_{p,q}A^*_{l,r}\exp(2\pi i(q\omega_p - r\omega_l)\frac{n-n_0}{N}) + A^*_{p,q}A_{l,r}\exp(2\pi i(-q\omega_p + r\omega_l)\frac{n-n_0}{N}) + \\
& \left.\left.A^*_{p,q}A^*_{l,r}\exp(-2\pi i(q\omega_p + r\omega_l)\frac{n-n_0}{N})\right]\right) \\
= & \sum_{q=1}^{Q_p-1}\sum_{r=1}^{Q_l-1} qr\left[\Re(A_{p,q}A_{l,r}Y''(q\omega_p + r\omega_l) - \Re(A_{p,q}A^*_{l,r}Y''(q\omega_p - r\omega_l)\right] \qquad (7.19)
\end{aligned}$$

The cross product of all terms in the sum can be optimized by taking into account the bandlimited property of $Y''(m)$, implying that only components which have close frequencies must be evaluated. For instance for a given value q, and a given frequency response bandwidth β, only the r values must be considered for which $r\omega_l$ falls in the main lobe. Since

$$\begin{aligned}
0 \leq q\omega_p \leq< \frac{N}{2} \\
0 \leq r\omega_l \leq< \frac{N}{2}
\end{aligned}$$

the input values of Y'' are bounded by

$$\begin{aligned}
-\frac{N}{2} \leq q\omega_p - r\omega_l \leq< \frac{N}{2} \\
0 \leq q\omega_p + r\omega_l \leq< N
\end{aligned}$$

since the main lobe of $Y(q\omega_p - r\omega_l)$ ranges from $-\frac{N}{M}\beta$ to $\frac{N}{M}\beta$. This implies that for $Y(q\omega_p - r\omega_l)$ only the r values must be considered for which

$$\begin{aligned}
& -N\beta/M \leq q\omega_p - r\omega_l \leq N\beta/M \\
\Rightarrow \quad & \frac{q\omega_p - N\beta/M}{\omega_l} \leq r \leq \frac{q\omega_p + N\beta/M}{\omega_l}
\end{aligned}$$

For $Y(q\omega_p + r\omega_l)$ the main lobe is divided over the left and right side of the spectrum due to spectral replication yielding the intervals $[0, \frac{N}{M}\beta]$ and $[N - \frac{N}{M}\beta, N]$. The two intervals for $Y(q\omega_p + r\omega_l)$ yield

$$\begin{aligned} & 0 \leq q\omega_p + r\omega_l \leq N\beta/M \\ \Rightarrow \quad & \frac{-q\omega_p}{\omega_l} \leq r \leq \frac{N\beta/M - q\omega_p}{\omega_l} \end{aligned}$$

and

$$\begin{aligned} & N - N\beta/M \leq q\omega_p + r\omega_l \leq N \\ \Rightarrow \quad & \frac{N(1-\beta/M) - q\omega_p}{\omega_l} \leq r \leq \frac{N - q\omega_p}{\omega_l} \end{aligned}$$

This results finally in

$$\begin{aligned} \sum_{q=1}^{Q_p-1} \Bigg[& \sum_{r=1}^{r_{max,1}} qr\Re(A_{p,q}A_{l,r}Y''(q\omega_p + r\omega_l) \\ & + \sum_{r=r_{min,2}}^{r_{max,2}} qr\Re(A_{p,q}A_{l,r}Y''(q\omega_p + r\omega_l) \\ & - \sum_{r=r_{min,3}}^{r_{max,3}} qr\Re(A_{p,q}A^*_{l,r}Y''(q\omega_p - r\omega_l) \Bigg] \end{aligned}$$

with

$$\begin{aligned} r_{max,1} &= \lfloor \frac{N\beta/M - q\omega_p}{\omega_l} \rfloor \\ r_{min,2} &= \lceil \frac{N(1-\beta/M) - q\omega_p}{\omega_l} \rceil \\ r_{max,2} &= \lfloor \frac{N - q\omega_p}{\omega_l} \rfloor \\ r_{min,3} &= \lceil \frac{q\omega_p - N\beta/M}{\omega_l} \rceil \\ r_{max,3} &= \lfloor \frac{q\omega_p + N\beta/M}{\omega_l} \rfloor \end{aligned}$$

As a result the double sum can be computed in linear time. For the harmonic model, the hessian is not band diagonal but evidently, fewer frequencies are required.

The computation of the gradient and Hessian for the harmonic model is depicted in figure 7.3 and 7.4

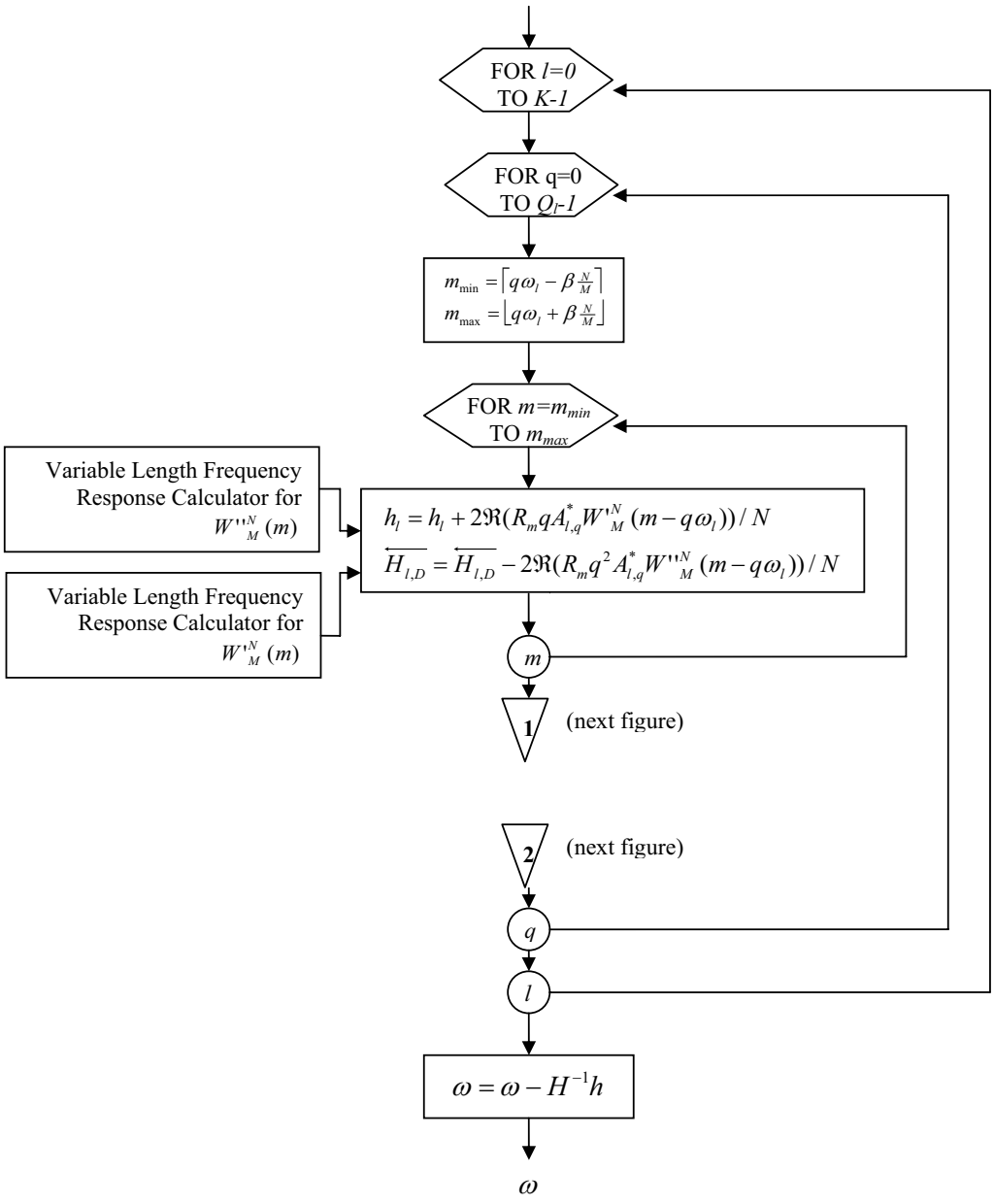


Fig. 7.3: *Frequency Optimiser for Harmonic Model*

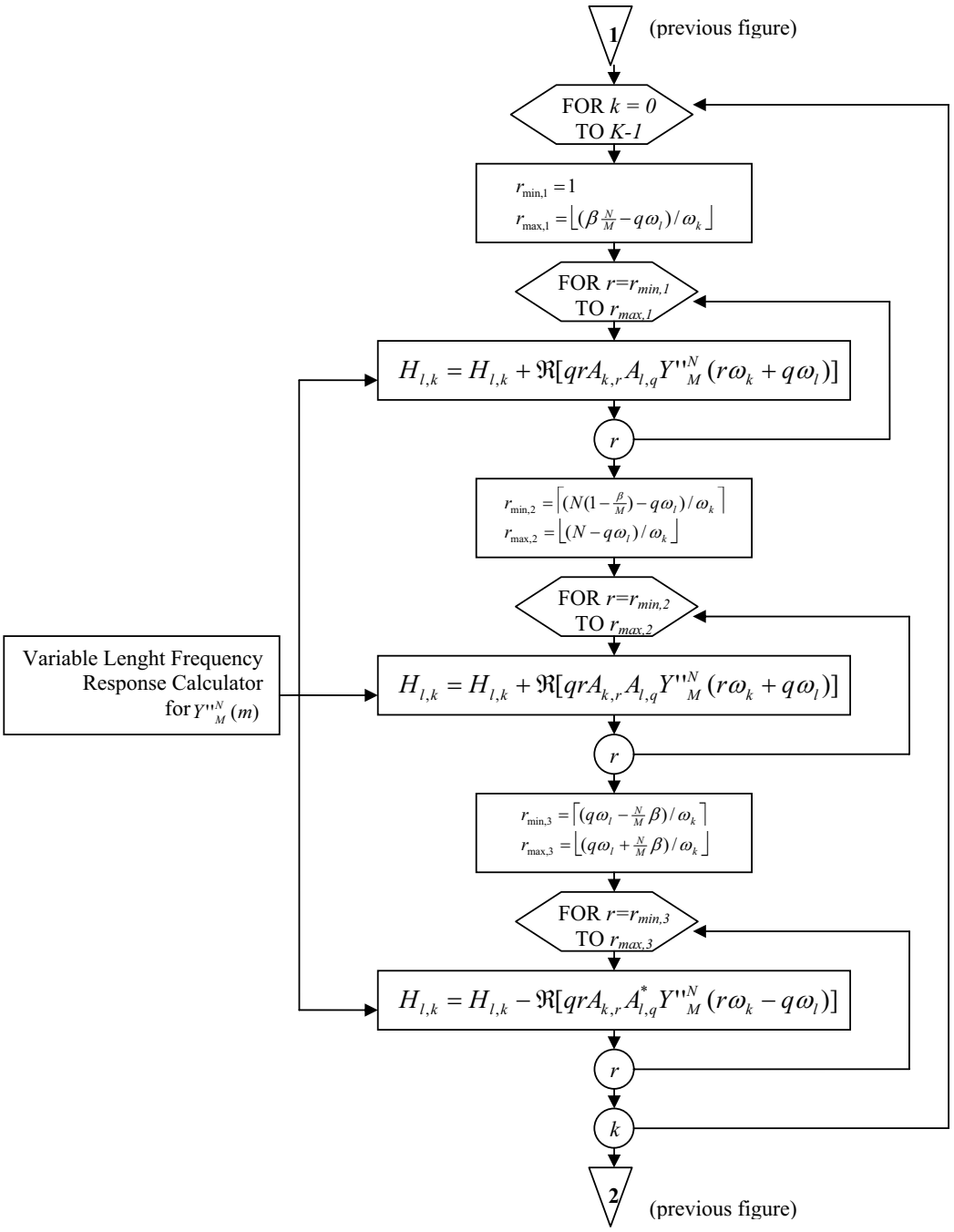


Fig. 7.4: *Frequency Optimiser for Harmonic Model (subroutine)*

7.7 The complete method

In figure 7.7, the complete analysis / synthesis method is depicted. This method can be applied for a harmonic or non-harmonic signal model.

First, the initial frequencies are computed. In the case of the harmonic model, a (multi)pitch estimator is used which determines an initial set of pitches from which a series of frequencies is computed for each source. For the non-harmonic model, peak picking can be used on the spectrum of the signal.

The preprocessing routine sorts the frequencies, removes frequencies that are too close to one another and determines how many diagonal bands D must be considered for the amplitude computation. Then, the amplitudes are computed. This is realized by computing the band diagonal elements of matrix $\mathbf{B}$ and storing them in a shifted form $\overleftarrow{\mathbf{B}}$. The matrix $\mathbf{C}$ is computed next and by solving the band limited system, the amplitudes are obtained.

The IFFT syntesizer computes the spectrum $\tilde{X}_m$ from the amplitudes $\bar{A}$ and frequencies $\bar{\omega}$. The difference with the original spectrum X_m results in the residual spectrum R_m which is used for the frequency optimization in the next step.

In the case of the non-harmonic model, the Hessian matrix $\mathbf{H}$ is band limited and is stored in a shifted form $\overleftarrow{\mathbf{H}}$. A second result of this property is that its inverse can be computed in linear time. The hessian and gradient are used to optimize the frequency values (Figure 7.1).

In the harmonic case, the fundamental frequencies of the different sources is optimized. Here, the Hessian is not band diagonal but it is very small since typically just a few sources are considered (Figure 7.3 and 7.4).

The iterative loop is continued until a stopping criterium is met. The results of the analysis are; the synthesized signal $\bar{x}_n$, the noise residual r_n, the amplitudes $\bar{A}$ and frequencies $\bar{\omega}$. Note that the amplitudes are complex and therefore contain the phases.

7.8 Results

Results are presented for the complete analysis/synthesis method for a single monophonic sound source. The window length was adapted to the pitch and was chosen chosen to be three fundamental periods. Results are presented for a recording of a trumpet playing slurred notes. This sound signal is shown in Fig. 7.6 and is particularly difficult because of the many transients. However, the obtained resynthesis is free of artifacts and undistinguishable from the original signal by our listening experience. Figure 7.7 shows the estimated amplitudes and frequencies over time. Finally, Figure 7.8 shows the detail of a transient and its resynthesis. In addition, a lower pitched version is presented of which the fundamental period is enlarged. One can observe clearly how the lower pitched sound follows the amplitude envelope of the original sound.

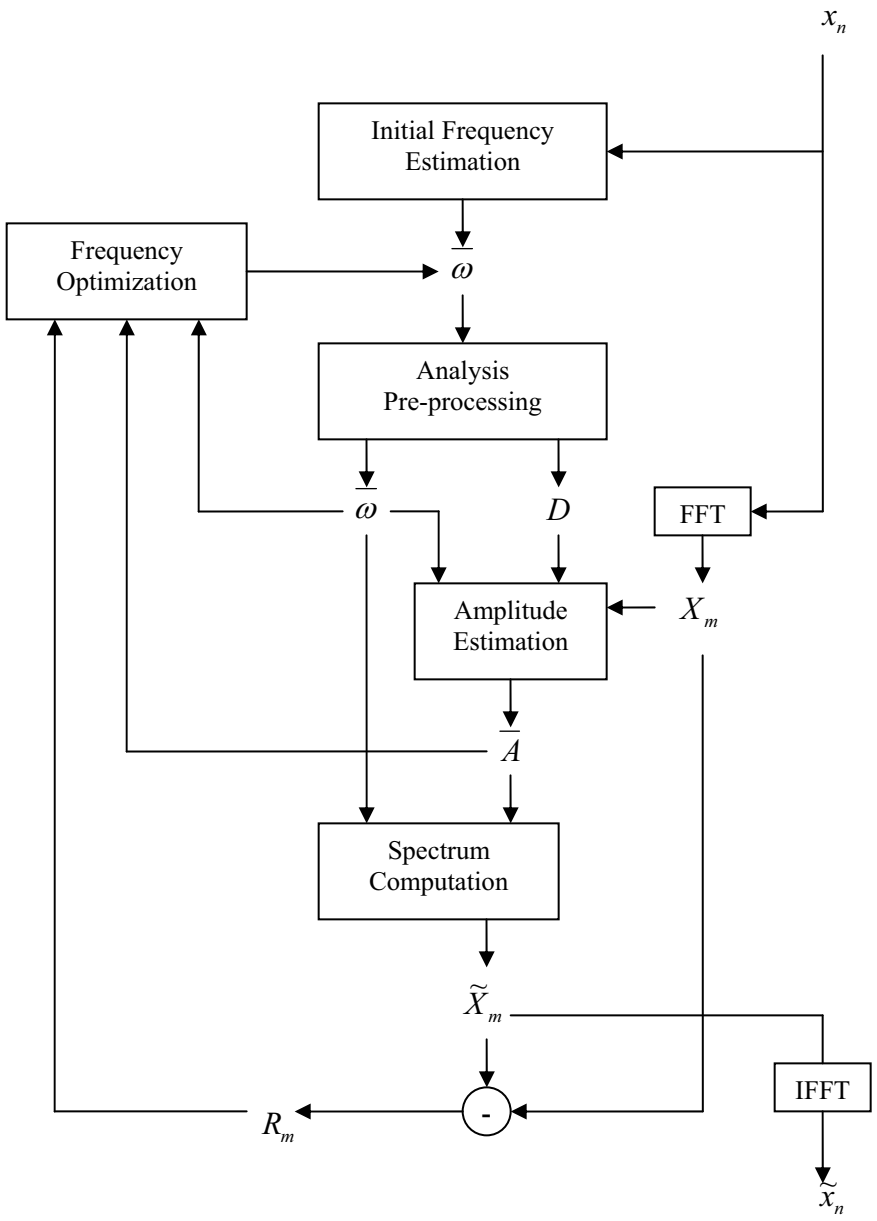


Fig. 7.5: *Highly Optimized Nonlinear Least Squares Technique for Sinusoidal Analysis*

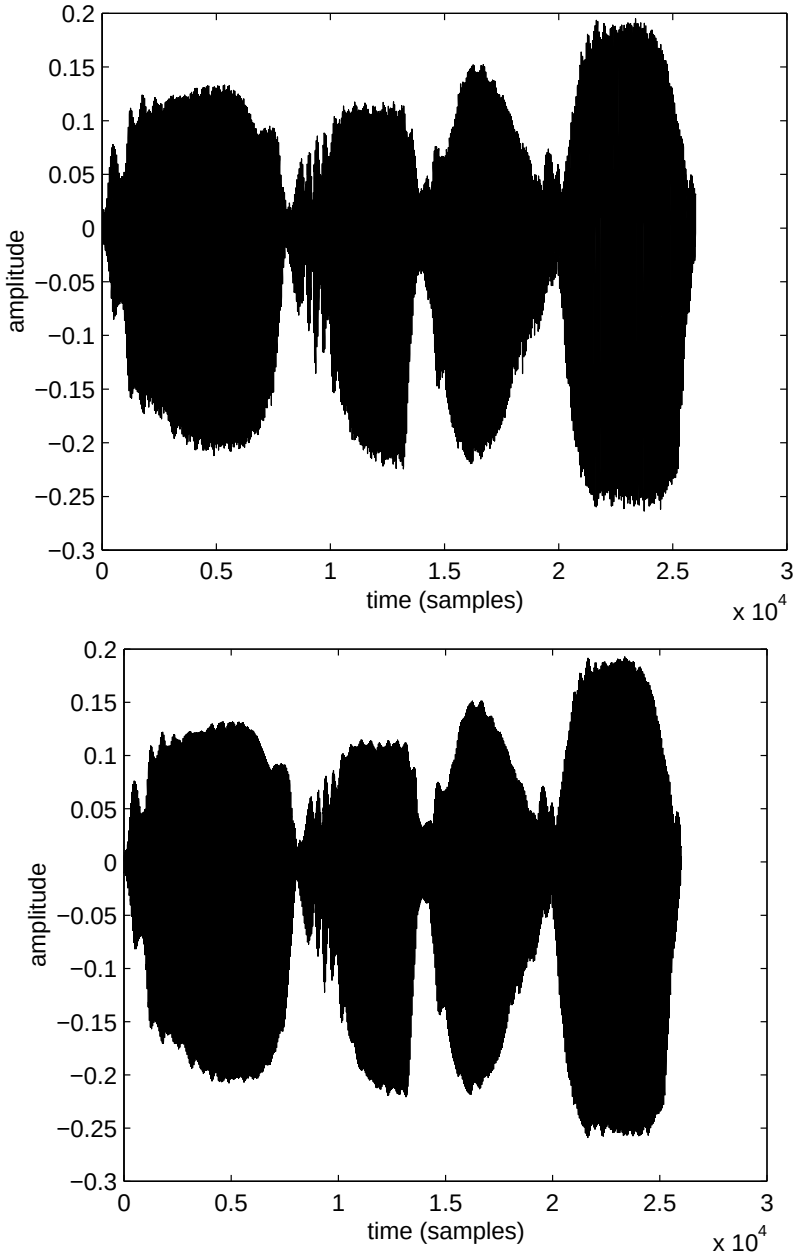


Fig. 7.6: **Top:** *Original Signal,* **Bottom:** *Resynthesis*

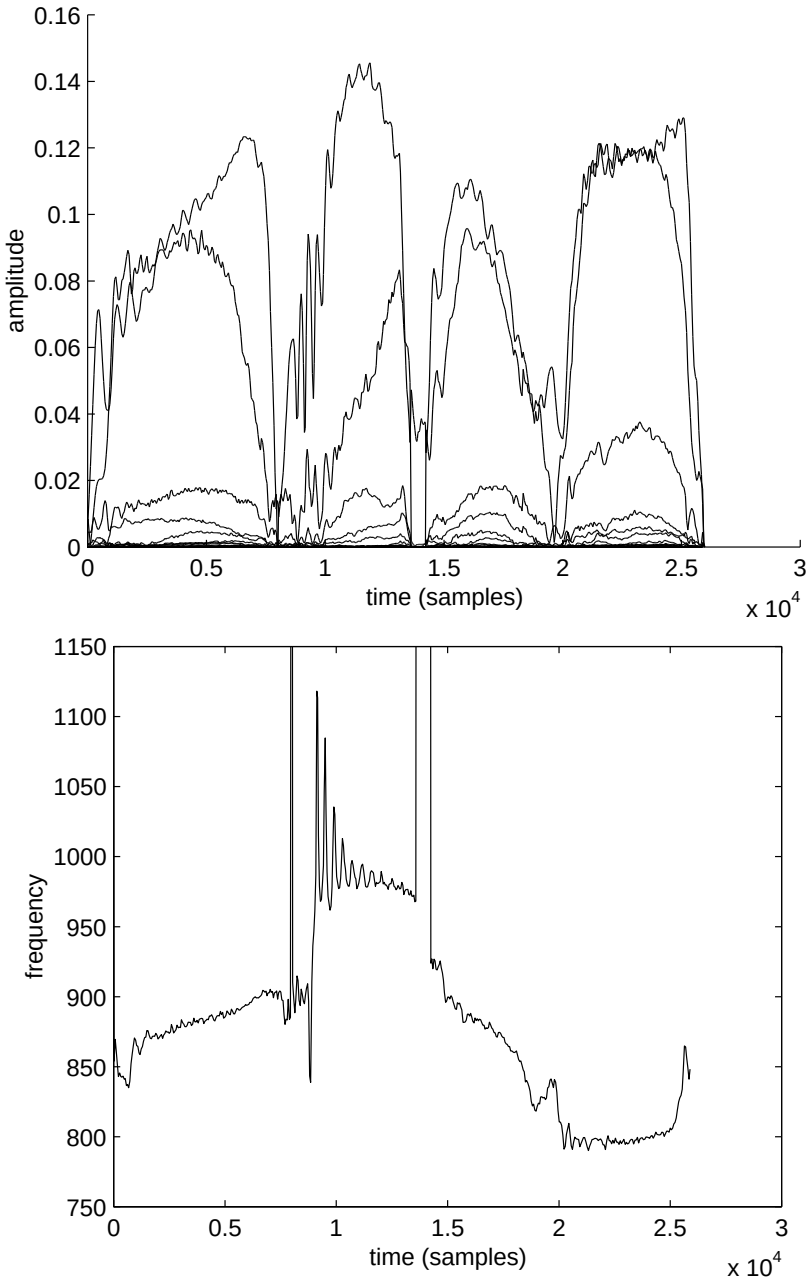


Fig. 7.7: **Left:** *Estimated Amplitudes* **Right:** *Estimated Fundamental Frequency*

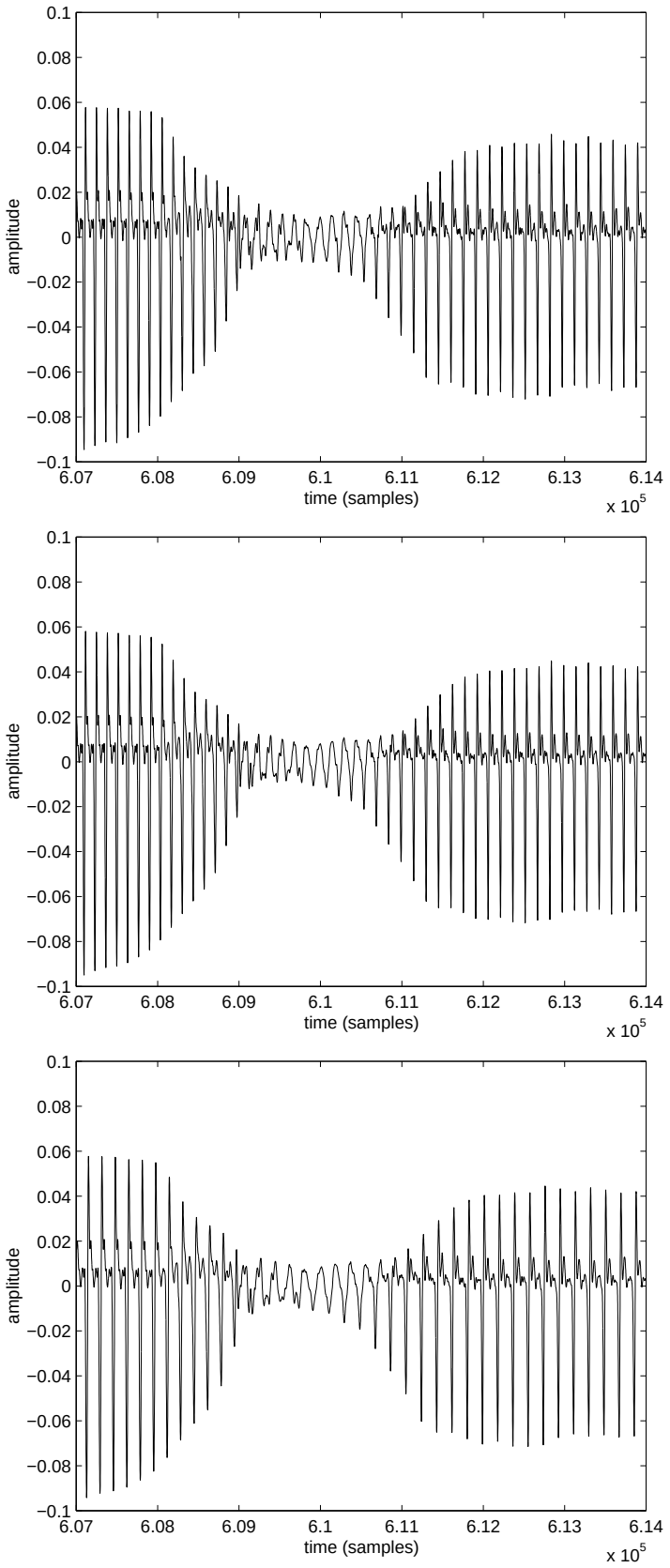


Fig. 7.8: *Zoom at transient* **Top:** *Original Signal* **Middle:** *Synthesis* **Bottom:** *Transposed Synthesis*

7.9 Conclusions and Future Work

The results of last two chapters provide a complete analysis/synthesis system which is able to handle overlapping frequency responses by computing all amplitudes simultaneously using a least squares method. Although the principles on which these methods are based are commonly known, they are not frequently applied because of the large computational effort they require. Our contribution consists of optimizing the amplitude estimation which had originally a third power complexity, to $\mathcal{O}(N\log(N))$. Also for the two known frequency optimization methods this computational gain was realized.

The frequency optimization techniques that were discussed previously are based implicitly on the assumption that the amplitudes are independent of the frequencies meaning that

$$\frac{\partial A(\bar{\omega})}{\partial \omega_k} = 0 \tag{7.20}$$

for each k. One remarkable observation when looking at Figure 7.2 however, is that the frequency optimization is very fast in case the correct amplitudes are known. This suggests that the slow convergence is due to this incorrect assumption.

In order to illustrate the frequency dependence of the amplitudes, a simple experiment was conducted. A spectrum was computed consisting of three sinusoidal components with amplitudes $\bar{A} = [111]$ and frequencies $\bar{\omega} = [\frac{4}{N}\frac{7}{N}\frac{10}{N}]$. The frequency dependence was tested by taking different values for ω_2 ranging from $\frac{4.5}{N}$ to $\frac{9.5}{N}$ and computing the amplitudes with these modified values. The results are depicted in Figure. 7.9, from which a strong dependence can be concluded. Note that the amplitudes all obtain their desired values when ω_2 reaches its true value being $\frac{7}{N}$.

A future challenge consists of including the frequency dependence of the amplitudes in the gradient and Hessian.

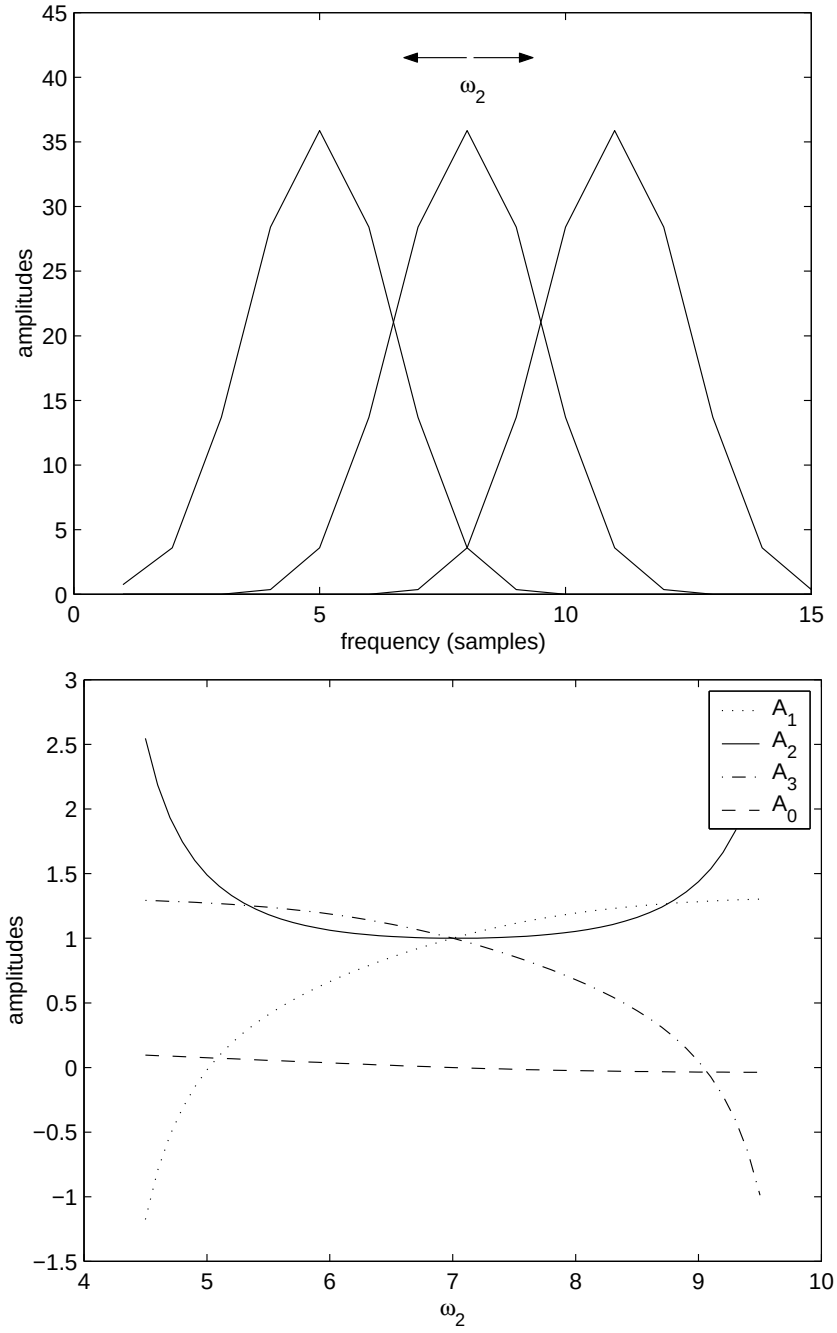


Fig. 7.9: *Frequency Dependence of the Amplitudes.*

Conclusions

In the first part of the thesis, methods are developed which determine in an automatic manner the control parameters for a physical model of a trumpet.

- **Chapter 1: Non Parametric Control Parameter Estimation**

  A non parametric control parameter estimation method is developed which is based on K-nearest neighbors classification. This approach does not use any prior information on the relationship between the control parameters and the signal features and is based entirely on a deta set. First, a large set of sounds is synthesized. For this set of sounds, perceptually relevant features are computed which are concatenated with their corresponding control parameters and stored in a data set. In order to simulate a given recording, the same features are computed. Then, the nearest neighbor of this feature vector is computed from the data set, and the corresponding control parameters are returned. These parameters are finally used to synthesize a sound signal that simulates the original signal.

  This approach is implemented and tested for a physical model of trumpet developed by the Analysis/Synthesis team of IRCAM. The results show that the method is successful for sounds that are well represented by the data set. However, the method has very weak generalization properties which means that the method has difficulties simulating other sounds than the ones that are in the data set. The main reason for this problem is that the feature space has a larger dimensionality than the control parameter space. This makes the feature space very sparse. In addition, the following problems that were encountered.

  - The physical constraints are not respected.
  - The distance criterium does not have a physical meaning and contains a user specified parameter.
  - The feature extraction fails during transients. In this case, the control parameters are extrapolated from their context.

- **Chapter 2: Fast K-Nearest Neighbors Computation**

  The size of the data set can prohibit the practical use of the nearest neighbor classifier. Therefore, branch and bound search algorithms were developed of which the time complexity is sublinear in function of the number of feature vectors.

  In a pre-processing step, the data is decomposed hierarchically which can be represented by a tree. Each node of the tree represents a subset of the data. The search algorithm itself is a depth first tree traversal algorithm which avoids nodes which cannot contain nearest neighbors. The rule which determines whether a node can be eliminated is called the *elimination rule* and is based on a lower bound distance between a vector and a node.

  In this work, a statistical model of the total computation cost is proposed from which two efficiency criteria can be derived. The first criterium states that each child node should contain the same number of feature vectors while the second criterium expresses that the number of vectors that lie close to the hyperplane should be minimized. It is shown that this is the case when the separating hyperplane is orthogonal to the maximal variance.

  Another aspect that influences the efficiency of the search algorithm is the traversal order. A local optimization of the traversal order is realized by pushing the child that most likely contains the nearest neighbors last so that it is evaluated first. When the traversal order is optimized globally, the node is selected which has the smallest lower bound distance.

  The decomposition level is a user defined parameter that has a strong influence on the efficiency. A method is proposed which determines the optimal decomposition using the statistical cost model.

  By combining the decomposition methods, elimination ruled and traversal orders, ten different search algorithms are obtained. These algorithms are compared for artificial data sets containing gaussian distributed vectors after determining the optimal decomposition level. For a low number of dimensions, the differences between the number of traversed nodes is quite small and the evaluation cost of a single node is the predominant factor. For a high number of dimensions, the efficiency of the elimination rule was more important than the node evaluation cost. The globally optimized traversal order introduced too much overhead and failed to realize a lower computation cost.

  One can conclude that the elimination rules become less effective when the number of dimensions increases. Fortunately, real high dimensional data sets are often highly dependent which makes that they can be represented in a lower dimensional vector space. The strength of our decomposition method is that it adapts itself automatically to the distribution of the vectors and takes into account local correlations of the data. Interestingly, lower computation costs were obtained for correlated and clustered data.

- **Chapter 3: Physical Model of a Trumpet and its Constraints**

  Although the initial objective was to develop a method which uses as few prior knowledge as possible, the non parametric estimation technique returned control parameter values that did not respect the physical constraints of a real instrument. When a sound with vibrato was simulated, a varying tube length was returned. This problem can only be tackled by designing data sets that take into account these constraints. When a player controls a trumpet he can only obtain seven different tube lengths by pressing the valves. Therefore, a data set must be designed which only uses these fixed tube lengths.

  By studying the physical model and its implementation, it is shown that the relationship between the tube length and lip frequency is particularly important. By increasing the lip frequency and keeping the tube length constant, strong resonances were obtained for multiples of the resonance frequency of the tube. Such a strong resonance is called a "mode", and by exciting the different modes of the tube, different notes are obtained.

  Some very simple and approximative relationships are derived between the tube length and lip frequency which results in the optimal resonance. From this result it was possible to determine a set of tube lengths with respect to a given tuning frequency. These tube lengths were then used for the synthesis of the data set.

- **Chapter 4: Discrete Cepstrum Coefficients as Perceptual Features**

  Since a trumpet sound is quasi-periodic, it can be described appropriately by its fundamental frequency and the relative amplitude of all partials. Many methods are available to model the envelope of a short time spectrum such as linear prediction coefficients, the cepstrum and the discrete cepstrum. For a harmonic signal, where the spectrum consists of regularly spaced peaks, the discrete cepstrum is the most appropriate method. In order to compute the discrete cepstrum coefficients, the spectrum is first modelled by a set of sinusoids for which the amplitudes and frequencies are computed. The discrete cepstrum is then computed by fitting a harmonic cosine series to the amplitude values. In addition, the perceived frequency scale of a human listener is taken into account by expressing the envelope on the Mel scale.

  However, when these coefficients are plot over consecutive time frames, one observes that they are very noisy even when the perceived sound is very stable. The reasons for this instability are twofold. A first reason is that overfitting occurs. Since the envelope values are only defined at the frequencies of the sinusoids, one cannot predict its behavior between two spectral peaks. This can result in envelopes which are very accurate at the peaks but oscillate very wildly in between. This problem is even more pronounced when the spectral peaks are transformed to the Mel sale since this results in very large gaps in the low frequency band. Known methods such as cloud smoothing and regularization all rely on user specified parameters. It is shown to be quite easy to obtain a smooth envelope on the

linear scale by adapting the order. This led to the idea to compute the Mel scale coefficients from the linear scale coefficients directly which was name posterior warping.

A second reason is that in the high frequency band many of the spectral peaks have a small amplitude and therefore a very low signal to noise ratio. Although the perceptual relevance of this low amplitude partials is very small, their noise-like behavior is amplified enormously by the log function. The effect of these partials is reduced by applying a simple threshold.

- **Chapter 5: A Conditional Estimation Technique for Determining the Control Parameters**

  The results obtained in the two previous chapters are now taken into account to develop a new non parametric estimation technique for the control parameters. This means that the physical constraints are taken into account and that the features have been stabilized.

  The distance criterium that was used in the previous method contains a parameter λ which allows to control the relative importance of the spectral and tonal similarity. In the new method, the two similarity criteria were kept separately. Evidently, it is not guaranteed that a minimum can be found for both criteria.

  Therefore, one criterium is given a higher priority resulting in a method which was tentatively named conditional estimation. The method realizes the optimization with respect to two parameters P_1 and P_2, and takes into account two criteria $D_1(P_1, P_2)$ and $D_2(P_1, P_2)$. For each value P_1, the value P_2 is determined for which the similarity criterium D_2 is optimized. This is expressed by f yielding $P_2 = f(P_1)$. By substituting P_2 in the argument of D_1 by this function, one obtains $D_1(P_1, f(P_1))$. The function D_1 is now one-dimensional and is the optimized for P_1. This optimal value is then used to compute the second parameter using $P_2 = f(P1)$. The name *conditional optimization* was chosen because the function f yields the optimal value P_2 under the condition the other parameter has a given value P_1.

  The method was applied successfully on the control parameter estimation problem. It was shown to be very robust, and yields a unique solution for each parameter.

The estimation of the control parameters could not be performed at the transients since the feature extraction which is based on sinusoidal modelling fails. The main reason is that the analysis window is quite large which makes it not suitable to analyze fast variations in frequency and amplitude. Therefore, the second part of this dissertation studies methods which allow to use very small analysis windows.

- **Chapter 6: Amplitude Estimation, from $\mathcal{O}(K^2N)$ to $\mathcal{O}(N \log(N))$**

  When modelling a short time signal of length N by a sum of K sinusoids, a least squares method is commonly used to compute the amplitudes for a given set of

frequencies. When the frequency responses of the individual sinusoidal components do not overlap, the amplitude can be computed for each sinusoid iteratively which requires an $\mathcal{O}(N \log(N))$ time complexity. However, when small windows are used, all frequency responses overlap which implies that all amplitudes must be computed simultaneously. It is known that this requires a complexity $\mathcal{O}(K^2 N)$.

Our contribution consists of reducing this time complexity to $\mathcal{O}(N \log(N))$. This is realized by explicitly including an analysis window with a bandlimited frequency response to the least squares derivation. Therefrom, it is shown that the set of equations which is solved to compute the amplitudes is band diagonal. This results in linear time complexity. However, the estimation is realized on the spectrum which is computed by a fast fourier transform requiring an $\mathcal{O}(N \log(N))$ complexity.

- **Chapter 7: Frequency Optimization, from $\mathcal{O}(K^2 N)$ to $\mathcal{O}(N \log(N))$**

  In the case that frequency responses are not overlapping, the frequency can be estimated by determining the value at the maximum of each peak. For small analysis windows however, the individual peaks cannot be distinguished which implies that an iterative optimization method must be used.

  Any of the conventional optimization techniques can be applied for the optimization of the frequencies. In this work, Newton's method is considered and a model linearization method. Again, the computational complexity can prohibit its practical use. For both methods, we show that by applying the same methodology as for the amplitude estimation, the computational complexity can again be reduced to $\mathcal{O}(N \log(N))$. No significant difference was observed for the speed of convergence of both methods.

Nederlandse Samenvatting

Dit proefschrift behandelt de automatische schatting van controleparameters voor muzikale synthesealgoritmes ter simulatie van een gegeven signaal. In het domein van de muzieksynthese worden twee paradigmas onderscheiden namelijk; *signaalmodellerings-synthese* die gebaseerd is op een wiskundig model en synthese door *fysische modellering* die gebaseerd is op de simulatie van de akoestische en mechanische eigenschappen van een muziekinstrument.

Singaalmodelleringssynthese beschrijft een geluidssignaal in termen van een wiskundig model waarvoor de parameters kunnen berekend worden door de fout tussen een gegeven signaal en het model te minimaliseren. Een frequent gebruikte methode is sinusoïdale modellering waarbij het signaal wordt beschreven als een som van sinusoïdes met tijdsvariërende amplitudes en frequencties. Het residu wordt dikwijls gemodelleerd door gefilterde witte ruis. Vele technieken zijn reeds bekend die toelaten om deze parameters op een accurate manier te bepalen waardoor een synthese van zeer hoge kwaliteit wordt bekomen.

Voor fysische modellen, is de schatting van de controleparameters veel moeilijker omdat deze parameters niet op een triviale manier gerelateerd zijn met het voortgebrachte signaal. De meeste muziekinstrumenten gedragen zich op een niet-lineaire manier. Dit is bijvoorbeeld het geval wanneer de lippen van een trompetspeler tegen elkaar botsen, bij het aanstrijken van een vioolsnaar en bij het openen en sluiten van de stembanden. Daarenboven doet zich ook vaak een terugkoppelling voor zoals bij de weerkaatsing van een drukgolf aan het einde van een buis, of een transversale golf die wordt gereflecteerd aan het einde van een snaar. Dit werk, heeft als doel om zo weinig mogelijk voorafgaande informatie over het fysisch model te gebruiken zodat het mogelijk is de schattingsmethodes toe te passen op andere modellen dan het model dat in dit proefschrijft wordt bestudeerd. De voorgestelde methodes zijn afkomstig uit het patroonherkenningsdomein en zijn zuiver gebaseerd op een data set die invoer- en uitvoerwaarden bevat.

In het eerste deel van de thesis, wordt de schatting van controleparameters behandeld voor een fysisch model van een trompet.

- **Hoofdstuk 1: Niet-Parametrische Schatting van Controleparameteres**

  Een niet-parametrische schattingsmethode wordt voorgesteld die gebaseerd is op classificatie door K-dichtste naburen (K-nearest neighbors). Deze aanpak gebruikt geen voorkennis over de relatie tussen de controleparameters en signaalkenmerken en is volledig gebaseerd op de data set. Eerst wordt een grote set geluiden gesynthetiseerd. Uit deze geluiden worden relevante signaalkenmerken berekend die vervolgens geconcateneerd worden met de overeenkomstige controleparameters en uiteindelijk worden opgeslagen in de data set. Om een gegeven opname te simuleren worden ook uit dit signaal deze kenmerken berekend. Vervolgens wordt de dichtst bijzijnde nabuur van de data set bepaald en worden de overeenkomstige controleparameters als resultaat teruggegeven. Deze parameters worden uiteindelijk gebruikt om een geluidssignaal te synthetiseren dat het oorspronkelijk signaal simuleert.

  Deze benadering werd geïmplementeerd en getest voor een fysisch model van een trompet ontwikkeld in de Analyses/Synthese groep van het IRCAM. De resultaten tonen aan dat de methode succesvol geluiden simuleert die adequaat gerepresenteerd worden in de data set. De methode heeft echter zeer zwakke generalisatieëigenschappen wat betekent dat ze niet behoorlijk werkt voor andere geluiden dan degene die in de data set aanwezig zijn. De hoofdreden voor dit probleem is dat het aantal dimensies van de kenmerkruimte groter is dan de ruimte van de controleparameters. Dit maakt de kenmerkruimte zeer ijl. Daarenboven werden volgende problemen waargenomen.

  - De fysische beperkingen worden niet gerespecteerd.
  - De afstandsmaat heeft geen fysische betekenis en omvat een parameter die door de gebruiker moet worden ingesteld.
  - De kermerkextractie faalt tijdens transiënten. In dit geval worden de controleparameters afgeleid uit de context.

- **Hoofdstuk 2: Snelle Berekening van de K-Dichtste Naburen**

  De grootte van de data set kan het praktisch gebruik van dichtste naburen classificatie aanzienlijk bemoeilijken. Daarom werden "verdeel-en-begrens" zoekalgoritmes (branch and bound search algorithms) ontwikkeld waarvoor de tijdscomplexiteit sublineair is in functie van het aantal kenmerkvectoren.

  Vooraf wordt de data hiërarchisch gestuctureerd wat kan worden voorgesteld door een boom. Elke node van deze boom stelt een subset van de data voor. Het zoekalgoritme zelf doorloopt de boom in een "diepte eerst" volgorde waarbij nodes die geen dichtste naburen bevatten worden vermeden. De regel die bepaalt of een node effectief naburen bevat wordt de *eliminatieregel* genoemd en is gebaseerd op een benedengrens van de afstand tussen een vector en een node.

In dit werk, wordt een statistisch model van de totale berekeningskost voorgesteld waaruit twee efficiëntiecriteria kunnen worden afgeleid. Het eerste criterium stelt dat elke subnode een gelijk aantal vectoren moet bevatten terwijl het tweede criterium uitdrukt dat het aantal vectoren die dicht bij het hypervlak liggen moet worden geminimaliseerd. Er wordt aangetoond dat dit het geval is wanneer de data wordt opgesplitst volgens het hypervlak dat orthogonaal is met de maximale variantie van de dataset.

Een ander aspect dat de efficiëntie beïnvloedt is de doorloopvolgorde. Een lokaal geoptimaliseerde doorloop wordt gerealiseerd door de node met de kleinste ondergrens laatst op de stack te zetten zodat deze eerst wordt behandeld. De doorloopvolgorde kan ook globaal geoptimaliseerd worden door uit de hele stack de node te selecteren met de kleinste benedengrens.

Het decompositieniveau is een parameter die door de gebruiker wordt ingesteld en eveneens een grote invloed heeft op de rekentijd. Er wordt een methode voorgesteld die toelaat om de optimale decompositie te bepalen door gebruik te maken van het statistische model van de rekentijd.

Door de decompositiemethodes, eliminatieregels en doorloopvolgordes te combineren werden tien verschillende zoekalgoritmes bekomen. Deze zoekalgoritmes werden vergeleken voor artificiële data sets die gaussisch verdeelde vectoren bevatten nadat het optimale decompositieniveau werd bepaald. Voor een laagdimensionale kenmerkruimte is het verschil in aantal doorlopen nodes voor de verschillende eliminatieregels vrij klein, waardoor de doorloopkost van een node de belangrijkste factor is. Voor een hoogdimnesionale kenmerkruimte echter. is de efficiëntie van de eliminatieregel belangrijker dan de evaluatiekost. De globale optimalisatie van de doorloopvolgorde introduceert te veel overhead en slaagt er niet in een lagere rekentijd te bekomen.

Men kan concluderen dat de eliminatieregels minder efficiënt worden wanneer het aantal dimensies toeneemt. Bij echte data sets echter, zijn de verschillende kenmerken vaak sterk gerelateerd wat maakt dat ze kunnen worden voorgesteld in een ruimte met minder dimensies. De kracht van onze methode is dat ze zich automatisch aanpast aan de distributie van de vectoren en lokale correlaties in de data in acht neemt. Interessant om op te merken is dat voor gecorreleerde en geclusterde data lagere rekentijden werden bekomen.

- **Hoofdstuk 3: Fysisch Model van een Trompet en haar Fysische Beperkingen**

  Hoewel het oorspronkelijke doel erin bestond om een methode te ontwikkelen die zo weining mogelijk voorkennis vergt, werd er waargenomen dat de niet parametrisch schattingsmethode controleparameters opleverde die de fysische beperking van het instrument niet respecteerden. Wanneer een opname met vibrato werd gesimuleerd, resulteerde dit namelijk in een variatie van de buislengte. Dit probleem kan alleen worden verholpen door bij het ontwerp van de data set deze beperkingen

in acht te nemen. Een trompetspeler kan slechts zeven verschillende buislengtes bekomen door het indrukken van de ventiles. Daarom moet een data set gebruikt worden die eveneens werd bekomen door deze vaste buislengtes.

Door het fysisch model en zijn implementatie te bestuderen wordt er aangetoond dat het verband tussen de buislengte en lip frequenctie van zeer groot belang is. Door de buislengte constant te houden en de lipfrequenctie continu te verhogen werden werden sterke resonanties waargenomen bij veelvouden van de resonantie frequenctie van de buis. Zulk een sterke resonantie wordt een "mode" genoemd en door de verschillende modes te exciteren worden de verschillende noten bekomen.

Enkele zeer eenvoudige en benaderende relaties tussen de buislengte en lipfrequentie werden afgeleid waarvoor de optimale resonantie werd bekomen. Met dit resultaat was het mogelijk om een set van buislengtes te bepalen voor een gegeven stemmingsfrequentie. Deze buislengtes werden vervolgens gebruikt voor het ontwerp van de dat set.

- **Hoofdstuk 4: Discrete Cepstrum Coefficienten als Perceptuele Kenmerken**

  Daar een trompetgeluid quasi periodisch is, kan het beschreven worden door een harmonische reeks van sinusoïdes. De relatieve amplitude van deze sinusoïdes en de grondfrequenctie zijn daarom geschikte kenmerken om deze signalen te karakteriseren. Verschillende methodes zijn bekend om de spectrale enveloppe (functie die energieverdeling over het spectrum beschrijft) te modelleren. Zo kan deze enveloppe functie gekarakteriseerd worden door middel van lineaire predictie coëfficiënten, cepstrum coëfficiënten en het discreet cepstrum. Het is bekend dat voor een harmonisch spectrum waar de pieken zich op regelmatige afstanden in het spectrum bevinden, het discreet cepstrum de meest geschikte manier is. Na een analyse die de frequenties en amplitudes van de sinusoïdale componenten berekent uit het spectrum, worden de discrete cepstrum coefficienten berekend door een hamonische cosinus reeks te fitten aan de amplitudes. Bovendien wordt de enveloppe op de Mel frequentie schaal uitgedrukt die de toonhoogteschaal voorstelt zoals ze door het menselijk oor wordt waargenomen.

  Wanneer echter, de discrete cepstrum coëfficiënent werden gevisualiseerd over verschillende tijdsframes, kon men waarnemen dat deze zeer veel ruis bevatten hoewel het originele signal als zeer stabiel wordt waargenomen. De reden hiervoor is tweeledig. Ten eerste kan er zich overfitting voordoen. Sinds de enveloppe functie alleen is gedefinieerd voor de waargenomen frequenties is het gedrag van de functie tussen deze frequenties niet gedefiniëerd. Dit kan resulteren in enveloppes die zeer exact zijn op de waargenomen frequenties, maar sterk oscilleren ertussen. Dit is meer uitgesproken op de Mel schaal daar dit resulteert in grote intervallen in de lage frequentieband. Gekende methodes zoal cloud smoothing en regularisatie zijn allen afhankelijk van manueel ingestelde parameters. Er werd aangetoond dat op de lineaire schaal het vrij eenvoudig is om overfitting te vermijden door de orde van het discreet cepstrum aan te passen. Dit leidde tot het idee om de coefficiënten op

de Mel schaal rechtstreeks te berekenen uit de coëfficienten op de lineaire schaal. Deze methode werd posterior warping genoemd.

Een tweerde reden is dat in de hoge frequentieband de amplitudes zeer klein zijn en bijgevolg een zeer kleine signaal tot ruis verhouding hebben. Hoewel de perceptuele relevantie van deze amplitudes te verwaarlozen is wordt hun random gedrag enorm uitvergroot door de log functie. Het effect hiervan werd gereduceerd met behulp van een eenvoudige drempelwaarde.

- **Hoofdstuk 5: Een Voorwaardelijke Parameterschattingstechniek ter Bepaling van de Controleparameters**

  De resultaten die werden bekomen in de vorige twee hoofdstukken worden nu in acht genomen om een nieuwe niet-parametrische schattingsmethode voor de controleparameters te ontwikkelen. Dit betekent dat de fysische beperkingen worden gerespecteerd en dat het discreet cepstrum werd gestabiliseerd.

  De afstandsmaat die gebruikt werd in de vorige hoofstukken omvat een parameter λ die toelaat om het relatief belang van het spectraal en tonaal verschil te controleren. In de nieuwe methode worden deze criteria apart geoptimaliseerd. Uiteraard is het niet gegarandeerd dat voor beide criteria een minimum kan worden gevonden.

  Daarom wordt het ene criterium met een hogere prioriteit behandeld dan het andere. De methode realiseert de optimalisatie met betrekking tot twee parameters P_1 en P_2 en neemt de criteria $D_1(P_1, P_2)$ en $D_2(P_1, P_2)$ in acht. Voor elke waarde P_1, wordt de waarde P_2 geoptimaliseerd met betrekking tot criterium D_2. Dit wordt uitgedrukt door een functie f wat resulteert in $P_2 = f(P_1)$. Door P_2 te substitueren in het argument van D_1 bekomt men $D_1(P_1, f(P_1))$. De functie is nu 1-dimensionaal en wordt vervolgens geoptimaliseerd in functie van P_1. De naam *conditionele optimalisatie* werd gekozen omdat de functie f de optimale waarde P_2 teruggeeft voor een welbepaalde waarde P_1.

  Deze methode werd succesvol toegepast op het schattingsprobleem van de controleparameters van het fysisch model. Er werd aangetoond dat ze zeer robust is en een unieke waarde teruggeeft voor elke parameter.

De schatting vam de controleparameters kon niet worden uitgevoerd tijdens transiënten omdat in dat geval de kenmerkexctractie faalt. De hoofdreden hiervoor is dat het analysevenster vrij groot is wat ze niet geschikt maakt om snelle variaties in frequentie en amplitude te analyseren. Daarom worden in het tweede deel van de thesis methodes bestudeerd die toelaten om zeer kleine analysevensters te gebruiken.

- **Hoofdstuk 6: Amplitude Schatting, van $\mathcal{O}(K^2N)$ naar $\mathcal{O}(N\log(N))$**

  Wanneer een signaal bestaande uit N samples wordt gemodelleerd door K sinusoides wordt vaak een kleinste kwadraten methode gebruikt om de amplitudes te berekenen. Wanneer de frequentieresponses van de individuele sinusoidale componenten niet overlappen kan de amplitude van elke sinusoide iteratief worden

bepaald wat een complexiteit $\mathcal{O}(N \log(N))$ vergt. Hoewel, wanneer kleine analysevensters worden gebruikt worden alle amplitudes simultaan berekend. In dit geval vergt de berekening een complexiteit $\mathcal{O}(K^2 N)$.

Onze bijdrage bestaat erin om deze tijdscomplexiteit terug te dringen naar $\mathcal{O}(N \log(N))$. Dit wordt gerealiseerd door expliciet een analysevenster met een bandgelimiteerde frequentierespons in rekening te brengen bij de afleiding van de kleinste kwadraten. Daaruit kan worden afgeleid dat het stelsel vergelijkingen dat gebruikt wordt om de amplitudes te berekenen banddiagonaal is kan worden opgelost in $\mathcal{O}(K)$. Men moet echter wel de fourier transformatie in rekening brengen die een complexiteit $\mathcal{O}(N \log(N))$ vergt.

- **Hoofdstuk 7: Frequentie Optimalisatie, van $\mathcal{O}(K^2 N)$ naar $\mathcal{O}(N \log(N))$**

  In het geval dat de frequentieresponses niet overlappen kan de frequentie geschat worden door de waarde aan het maximum van de piek te bepalen. Voor kleine analysevensters echter, kunnen de pieken niet apart worden onderscheiden wat impliceert dat een iteratieve optimalisatie moet worden toegepast.

  Elk van de conventionele optimalizatietechnieken kan worden toegepast voor de bepaling van de frequenties. In dit werk, wordt de Newton methode bestudeerd en een linearisatiemethode van het model. Opnieuw kan de berekeningcomplexiteit gereduceerd worden naar $\mathcal{O}(N \log(N))$ door dezelfde methodologie toe passen als voor de amplitudeberekening. Wat betreft de convergentiesnelheid werd er geen significant verschil waargenomen tussen beide optimalisatiemethodes.

Bibliography

[1] Jean-Julien Aucouturier and Francois Pachet. Finding songs that sound the same. *1st IEEE Workshop on Model based Processing and Coding of Audio (MPCA)*, pages 91–98, 2002.

[2] Steve De Backer. *Unsupervised Pattern Recognition: Dimensionality Reduction and Classification*. PhD thesis, University of Antwerp, 2002.

[3] Heiko Purnhagen Bernd Edler. Concepts for hybrid audio coding schemes based on parametric techniques. *105th AES convention, San Francisco*, September 1998.

[4] Christopher Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[5] Gianpaolo Borin, Giovani De Poli, and Augusto Sarti. *Musical Signal Processing*. Swets & Zeitlinger, 1997.

[6] Judith C. Brown. Computer identification of musical instruments using pattern recognition with cepstral coefficients as features. *Journal of the Acoustic Society of America*, pages 1933–1941, 1999.

[7] Marine Campedel-Oudot, Olivier Cappé, and Eric Moulines. Estimation of the spectral envelope of voiced sounds using a penalized likelihood approach. *IEEE Transactions on Speech and Audio Processing*, 9(5):469–481, july 2001.

[8] Olivier Cappé, Jean Laroche, and Eric Moulines. Regularized estimation of cepstrum coefficients from discrete frequency points. *IEEE WASPAA*, October 1995.

[9] Olivier Cappé, Marine Oudot, and Eric Moulines. Spectral envelope estimation using a penalized likelihood criterion. *IEEE WASPAA*, October 1997.

[10] R. Caussé, R. Kergomard, and X. Lurton. Input impedance of brass musical instruments. *J. Acoust. Soc. Amer.*, 75:241–254, 1984.

[11] K.L. Clarkson. Nearest Neighbor Queries in Metrical Spaces. *Discrete and Computational Geometry*, 22:63–94, 1999.

[12] T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 1967.

[13] Roger Dannenberg, Jonathan Foote, George Tzanetakis, and Christoper Weare. Panel: New directions in music information retrieval. *Proceedings of the ICMC*, September 2001.

[14] Roger B. Dannenberg and Istvan Derenyi. Combining instrument and performance models for high-quality music synthesis. *Journal of New Music Research*, pages 211–238, September 1998.

[15] Philippe Depalle and Thomas Helie. Extraction of spectral parameters using a short time fourier transform modeling and no sidelobe windows. *Proceedings of the IEEE Workshop of Applications of Signal Processing to Audio and Acoustics (WASPAA), New Paltz, New York*, 1997.

[16] Wim D'haes. Estimation des paramètres d'un modèle physique de trompette par apprentissage. Technical report, IRCAM, july 2000.

[17] Wim D'haes. A highly optimized least squares technique for sinusoidal analysis: from $\mathcal{O}(k^2n)$ to $\mathcal{O}(n\log(n))$. *116th Audio Engineering Society (AES) Convention*, may 2004.

[18] Wim D'haes. A highly optimized method for computing amplitudes over a windowed short time signal. *IEEE Benelux Signal Processing Symposium*, april 2004.

[19] Wim D'haes, Dirk Van Dyck, and Xavier Rodet. An efficient branch and bound search algorithm for computing k nearest neighbors in a multidimensional vector space. *IEEE Advanced Concepts for Intelligent Vision Systems (ACIVS)*, September 2002.

[20] Wim D'haes, Dirk Van Dyck, and Xavier Rodet. Control parameter estimation for a physical model of a trumpet using pattern recognition. *IEEE Workshop on Model Based Processing and Coding of Audio (MPCA), Leuven, Belgium*, November 2002.

[21] Wim D'haes, Dirk Van Dyck, and Xavier Rodet. An efficient branch and bound search algorithm for computing k nearest neighbors in a multidimensional vector space. *Signal Proessing, Pattern Recognition and Applications (SPPRA)*, 2002.

[22] Wim D'haes, Dirk Van Dyck, and Xavier Rodet. Physical constraints for the control of a physical model of a trumpet. *International Conference on Digital Audio Effects (DAFx-02), Hamburg, Germany*, September 2002.

[23] Wim D'haes and Xavier Roder. Automatic estimation of control parameters: An instance-based learning approach. *Proceedings of the International Computer Music Conference (ICMC)*, september 2001.

[24] Wim D'haes and Xavier Rodet. Discrete cepstrum coefficients as perceptual features. *Proceedings of the International Computer Music Conference (ICMC), Singapore*, september 2003.

[25] Wim D'haes and Xavier Rodet. A new estimation technique for determining the control parameters of a physical model of a trumpet. *International Conference on Digital Audio Effects (DAFx-03)*, september 2003.

[26] Wim D'haes, Dirk van Dyck, and Xavier Rodet. PCA-based branch and bound search algorithms for computing K nearest neighbors. *Pattern Recognition Letters*, 24:1437–1451, june 2003.

[27] András Faragó, Thomás Linder, and Gábor Lugosi. Fast nearest neighbor search in dissimilarity spaces. *IEEE transactions on pattern analysis and machine intelligence*, 15:957–962, September 1993.

[28] Jonathan Foote. Content-based retrieval of music and audio. *Multimedia Storage and Archiving Systems II, Proc. of SPIE*, 3229:138–147, 1997.

[29] Keinosuke Fukunaga. *Statistical Pattern Recognition*. Academic Press, 1990.

[30] Keinosuke Fukunaga and Patrenahalli M. Nerada. A branch and bound algorithm for computing k nearest neighbors. *IEEE transactions on computers*, 24:750–753, July 1975.

[31] Thierry Galas and Xavier Rodet. An improved cepstral method for deconvolution of source-filter systems with discrete spectra: Application to musical sounds. *ICMC*, pages 82–84, 1990.

[32] Thierry Galas and Xavier Rodet. Generalized discrete cepstral method analysis for deconvolution of source-filter systems with discrete spectra. *IEEE WASPAA*, september 1991.

[33] Thierry Galas and Xavier Rodet. Generalized functional approximation for source filter system modeling. *Proceedings of Eurospeech*, pages 1985–1088, 1991.

[34] E. B. George and M. Smith. Analysis-by-synthesis overlap-add sinusoidal modeling applied to the synthesis of musical tones. *Journal of the Audio Engineering Society*, 40(6):497–516, 1992.

[35] E. B. George and M. Smith. Speech analysis/synthesis and modification using an analysis-synthesis/overlap-add sinusoidal model. *IEEE Transactions on Speech and Audio Processing*, 5(5):389–406, 1997.

[36] R. Gribonval, E. Bacry, S. Mallat, Ph. Depalle, and Xavier Rodet. Analysis of sound signals with high frequency matching pursuit. *Proc. IEEE Conf. Time-Frequency and Time-Scale Analysis*, june 1996.

[37] Liang Gu and Kenneth Rose. Perceptual harmonic cepstral coefficients as the front-end for speech recognition. *ICASSP*, May 2001.

[38] Thomas Helie, Christophe Vergez, Jean Levine, and Xavier Rodet. Inversion of a physical model of a trumpet. *Proceedings of the ICMC*, pages 149–152, 1999.

[39] Kris Hermus, Werner Verhelst, and Patrick Wambacq. A hybrid scheme for perceptual audio coding. *1st IEEE Benelux Workshop on Model based Processing and Coiding of Audio (MPCA-2002), Leuven, Belgium*, november 2002.

[40] Richard Heusdens. Rate-distortion optimal sinusoidal modeling of audio and speech using psychoacoustical matchinf pursuits. *1st IEEE Benelux Workshop on Model based Processing and Coiding of Audio (MPCA-2002), Leuven, Belgium*, november 2002.

[41] J.D. Reiss and J.-J. Aucouturier and M. B. Sandler. Efficient Multidimensional Searching Routines for Music Information Retrieval. *2nd Annual International Symposium on Music Information Retrieval (ISMIR), Bloomington, Indiana, USA*, 2001.

[42] Joselíto J. Chua. Fast Full-Search Equivalent Nearest Neighbour Search Algorithms. november 1999.

[43] A. Juan, E. Vidal, and P. Aibar. Fast k nearest neighbor searching through extended versions of the approximating and eliminating search algorithm (AESA). *Proc. of the 14th ICPR*, Vol. I:828–830, 1998.

[44] Behrooz Kamgar-Parsi. An improved branch and bound algorithm for computing k nearest neighbors. *Pattern Recognition Letters*, 3:7–12, January 1985.

[45] Steven M. Kay. *Fundamentals of Statistical Signal Processing, Estimation theory.* Prentice Hall, 1993.

[46] Florian Keiler and Sylvain Marchand. Survey on extraction of sinusoids in stationary sounds. *International Conference on Digital Audio Effects (DAFx-03)*, pages 51–58, 2002.

[47] Baek S. Kim and Song B. Park. A fast k nearest neighbor finding algorithm based on the ordered partition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):761–766, november 1986.

[48] Qin Li and Les Atlas. Time-variant least squares harmonic modeling. *International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2003.

[49] Carl G. Looney. *Pattern Recognition using Neural Networks*. Oxford University Press, 1997.

[50] S.G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, december 1993.

[51] Robert J. McAulay and Thomas F. Quatieri. Speech analysis/synthesis based on a sinusoidal representation. *IEEE transactions on acoustics, speech and signal processing*, 34(4):744–754, august 1986.

[52] James McNames. A fast nearest neighbor algorithm based on a principal axis search tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(9):964–976, september 2001.

[53] María Luisa Micó, José Oncina, and Raphael C.Carrasco. A fast branch & bound nearest neighbor classifier in metric spaces. *Pattern Recognition Letters*, 17:731–739, June 1996.

[54] María Luisa Micó, José Oncina, and Enrique Vidal. An algorithm for finding nearest neighbors in constant average time with a linear space complexity. *Proc. of the 11th ICPR*, Vol. II:557–560, 1992.

[55] María Luisa Micó, José Oncina, and Enrique Vidal. A new version of the nearest-neighbour approximation and elimination search alorithm (AESA) with linear preprocessing time and memory requirements. *Pattern Recognition Letters*, 15:9–18, January 1994.

[56] Tom M. Mitchel. *Machine Learning*. McGraw-Hill International Editions, 1997.

[57] Sirko Molau, Michael Pitz, Ralf Schlüter, and Hermann Ney. Computing Mel-frequency Cepstral Coefficients on the Power Spectrum. *ICASSP*, pages 73–76, May 2001.

[58] F.P. Myburg, A.C. den Brinker, and S.J.L. van Eijndhoven. Multi-component chirp analysis in parametric audio coding. *IEEE Benelux Signal Processing Symposium (SPS)*, april 2004.

[59] F.P. Myburg, A.C. den Brinker, and S.J.L. van Eijndhoven. Sinusoidal analysis of audio with polynomial phase and amplitude. *IEEE Benelux Signal Processing Symposium (SPS)*, april 2004.

[60] Axel Nackaerts, Bart De Moor, and Rudy Lawereins. Parameter estimation for dual-polarized plucked string models. *Proceedings of the ICMC*, September 2001.

[61] H. Niemann and R. Goppert. An efficient branch and bound nearest neighbour classifier. *Pattern Recognition Letters*, 7:67–72, February 1988.

[62] Peter Noll. Mpeg digital audio coding. *IEEE Signal Processing Magazine*, September 1997.

[63] Ted Painter and Andreas Spanias. Perceptual coding of digital audio. *Proceedings of the IEEE*, 88(4), April 2000.

[64] Geoffroy Peeters. *Modèles et modification du signal sonore adaptés à ses caractéristiques locales.* PhD thesis, IRCAM - Paris VI, july 2001.

[65] Geoffroy Peeters, Stephen McAdamns, and Perfecto Herrera. Instrument Sound Description in the Context of MPEG-7. *Proc. of the International Computer Music Conference (ICMC), Beijing, Chine*, September 2000.

[66] Geoffroy Peeters and Xavier Rodet. Sinola: A new analysis/synthesis method using spectrum shape distortion, phase and reassigned spectrum. *Proc. of the International Computer Music Conference (ICMC), Beijing, Chine*, October 1999.

[67] Geoffroy Peeters and Xavier Rodet. Hierarchical gaussian tree with inertiao ratio maximization for the classification of large musical instrument databases. *6th International Conference on Digital Audio Effects (DAFX03), London, UK*, September 2003.

[68] Heiko Purnhagen. Parameter estimation and tracking for time-varying sinusoids. *IEEE Workshop on Model Based Processing and Coding of Audio (MPCA), Leuven, Belgium*, november 2002.

[69] V. Ramasubramanian and Kuldip K. Paliwal. Fast nearest neighbor search algorithms based on approximation-elimination search. *Pattern Recognition*, 33:1497–1510, 2000.

[70] Axel R obel. Adaptive additive synthesis using spline based parameter trajectory models. *Proc. of the International Computer Music Conference (ICMC), Havana, Cuba*, September 2001.

[71] Xavier Rodet. Musical sound signals analysis/synthesis: Sinusoidal+residual and elementary waveform models. *Proceedings of the IEEE Time-Frequency and Time-Scale Workshop (TFTS'97), University of Warwick, Coventry, UK*, august 1997.

[72] Xavier Rodet and Philippe Depalle. A new additive synthesis method using inverse fourier transform and spectral envelopes. *Proceeding of the International Computer Music Conference (ICMC), San Jose, California*, pages 410–411, 1992.

[73] Xavier Rodet and Philippe Depalle. A new additive synthesis method using inverse fourier transform and spectral envelopes. *Convention of the Audio Engineering Society (AES), San Francisco*, 1992.

[74] Xavier Rodet and Christophe Vergez. Nonlinear dynamics in physical models: From basic models to true musical instruments. *Computer Music Journal*, 23(3):35–49, 1999.

[75] Xavier Rodet and Christophe Vergez. Nonlinear dynamics in physical models: Simple feedback-loop systems and properties. *Computer Music Journal*, 23(3):18–34, 1999.

[76] Manfred R. Schroeder and Bishnu S. Atal. Code-excited linear prediction (CELP): High quality speech at very low bit rates. *ICASSP*, pages 937–940, 1985.

[77] E.G.P Schuijers, A.W.J. Oomen, A.C. den Brinker, and A.J. Gerrits. Advances in parametric coding for high-quality audio. *1st IEEE Benelux Workshop on Model based Processing and Coiding of Audio (MPCA-2002), Leuven, Belgium*, november 2002.

[78] Diemo Schwarz and Xavier Rodet. Spectral envelope estimation and representation for sound analysis-synthesis. *ICMC*, pages 351–354, 1999.

[79] Stefania Serafin, Julius O. Smith III, Harvey Thornburg, Frederic Mazzella, Arnaud Tellier, and Guillaume Thonier. Data driven identification and computer animation of a bowed string model. *Proceedings of the ICMC*, September 2001.

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

[81] Christian Spevak. Soundspotter - a prototype system for content based audio retrieval. *International Conference on Digital Audio Effects (DAFx02)*, pages 27–32, 2002.

[82] Petre Stoica, Hongbi Li, and Jian Li. Amplitude estimation of sinusoidal signals: Survey, new results and an application. *IEEE Transactions on Signal Processing*, 48(2):338–352, February 2000.

[83] Fumitada Itakura Takafumi Hikichi, Naotoshi Osaka. Sho-so-in: New synthesis method for addition of articulations based on a sho-type physical model. *Internation Computer Music Conference (ICMC)*, september 2003.

[84] Patrice Tisseran. Portage du modèle physique de trompette sous jMAX. Technical report, IRCAM, 1999.

[85] Tero Tolonen. Methods for separation of harmonic sound sources using sinusoidal modeling. *106th Audio Engineering Society Convention (AES) convention, Munich, Germany*, september 1999.

[86] Tero Tolonen. A computationally efficient multiplitch analysis model. *IEEE transactions on speech and audio processing*, 8(6):708–716, november 2000.

[87] Tero Tolonen. Oject-based sound source modeling for musical signals. *109 th Audio Engineering Society Convention (AES)*, september 2000.

[88] Caroline Traube. Estimating the plucked point on guitar string. *Conference on Digital Audio Effects (DAFX-00)*, December 2000.

[89] Caroline Traube and Philippe Depalle. Extraction of the excitation point location on a string using wighted least-square estimation of comb filter delay. *Conference on Digital Audio Effects (DAFX-03)*, December 2003.

[90] Christophe Vergez. *Trompette et trompettiste: un système dynamique non linéaire à analyser, modéliser et simuler dans un contexte musical*. PhD thesis, Université Paris 6, IRCAM, 1999.

[91] Christophe Vergez and Xavier Rodet. Comparison of real trumpet playing, latex model of lips and computer model. *Proceedings of the ICMC*, September 1997.

[92] Christophe Vergez and Xavier Rodet. Dynamical systems and physical models of trumpet-like instruments. analytical study and asymptotical properties. *ACUSTICA - acta acustica*, 86:147–162, 2000.

[93] Christophe Vergez and Xavier Rodet. Trumpet and trumpet player: Modeling and simulation in a musical context. *Proceedings of the ICMC*, September 2001.

[94] E. Vidal. An algorithm for finding nearest neighbors in (approximately) constant average time complexity. *Pattern Recognition Letters*, 4:145–157, July 1986.

[95] E. Vidal. New formulation and improvements of the nearest neighbour approximation and elimination search alorithm (AESA). *Pattern Recognition Letters*, 15:1–7, January 1994.

[96] J.M. Vilar. Reducing the overhead of the AESA metric-space nearest neighbor searching algorithm. *Information Processing Letters*, 56:265–271, 1996.

[97] Tuomas Virtanen. Separation of harmonic sounds using multiplitch analysis and iterative parameter estimation. *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, December 2001.

[98] Tuomas Virtanen. Algorithm for the separation of harmonic sounds with time-frequency smoothness constraint. *International Conference on Digital Audio Effects (DAFx)*, September 2003.

[99] Koen Vos, Renat Vafin, Richard Heusdens, and Bastiaan Kleijn. High-quality consistent analysis-synthesis in sinusoidal coding. *106th AES convention, Munich, Germany*, may 1999.

[100] David Wessel, Cyril Drame, and Matthew Wright. Removing the time axis from spectral model analysis-based additive synthesis: Neural networks versus memory-based machine learning. *ICMC*, pages 62–65, 1998.

[101] Udo Zölzer. *DAFX: Digital Audio Effects*. Wiley, 2002.