An Introduction to Audio Content Analysis

IEEE Press
445 Hoes Lane
Piscataway, NJ 08854

Technical Reviewers
Roger B. Dannenberg, Carnegie Mellon University

An Introduction to Audio Content Analysis

Applications in Signal Processing and Music Informatics

Alexander Lerch
zplane.development, Berlin

IEEE PRESS

A JOHN WILEY & SONS, INC., PUBLICATION

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.
Published simultaneously in Canada.

Library of Congress Cataloging-in-Publication Data:

Lerch, Alexander.
Audio content analysis : an introduction / Alexander Lerch.
p. cm.
Includes bibliographical references and index.
ISBN 978-1-118-26682-3 (hardback)
1. Computer sound processing. 2. Computational auditory scene analysis. 3. Content analysis (Communication)—Data processing. I. Title.
TK7881.4.L485 2012
006.4'5—dc23 2012008107

Printed in the United States of America.

10 9 8 7 6 5 4 3 2

CONTENTS IN BRIEF

CONTENTS

PREFACE

The growing amount of audio and music data on the Internet and in user databases leads to an increasing need for intelligent browsing, retrieving, and processing of this data with automated methods. *Audio content analysis*, a subfield of the research field *music information retrieval*, aims at extracting (musical and perceptual) properties directly from the audio signal to support these tasks. Knowledge of these properties allows us to improve the interaction of humans or machines with digital audio signals. It enables new ways of assessing, processing, and visualizing music.

Although analysis of audio signals covers other research areas such as automatic speech recognition, we will restrict ourselves to the analysis of music signals in the context of this book.

When preparing classes on audio content analysis with a focus on music recordings it became quickly clear that — although there is a vast and growing amount of research literature available — there exists no introductory literature. This observation led to writing this book in the hope it might assist students, engineers, and developers who have basic knowledge of digital signal processing. The focus lies on the signal processing part of audio content analysis, but wherever it may improve the understanding of either algorithmic design choices or implementation details some basic characteristics of human perception, music theory, and notation as well as machine learning will be summarized.

Chapter 2 starts by introducing some definitions and offers a short reiteration of the most important tools of digital signal processing for the analysis of audio signals. The following chapters encompass the basic four technical content categories timbre, level, pitch, and rhythm. A fifth category is reserved for purely technical and statistical signal descriptions. Chapter 3 introduces low-level or short-term features that are widely used in systems for signal analysis. A large part of the chapter deals with timbre represen-

tations of a signal, accompanied by the introduction of statistical features. The chapter concludes with a summary of approaches to feature selection and post-processing. Chapter 4 focuses on intensity-related features. It covers envelope features and simple models of human loudness perception. The extraction of pitch-related information such as the detection of fundamental frequency, harmony, key, etc. is described in Chap. 5. Chapter 6 focuses on the temporal and rhythmic aspects of the audio signal. It explains the segmentation of audio signals into musical events and covers higher level information such as the detection of tempo and meter. The remaining chapters deal with analysis systems using combinations of timbre, loudness, onset, and pitch features to derive higher level information. Chapter 7 describes the automatic synchronization of two similar audio sequences or an audio and a score sequence. Musical genre classification, one of the most prominent research fields of audio content analysis, is explained in Chap. 8. Chapter 9 is about audio fingerprinting which is probably the commercially most successful application in audio content analysis. The concluding chapter, targeting classical music, covers the analysis of music performance. It is not a core field in audio content analysis but emphasizes the differentiation between performance aspects and musical aspects of recordings and elaborates on the manual and automated analysis methods used for musicological music performance analysis. The appendices provide details and derivations of some of the most important signal processing tools as well as a short survey on available software solutions for audio content analysis.

Downloadable MATLAB files are available at: http://www.audiocontentanalysis.org.

A. LERCH

Berlin
January, 2012

ACRONYMS

ACA	Audio Content Analysis
ACF	Autocorrelation Function
ADPCM	Adaptive Differential Pulse Code Modulation
AMDF	Average Magnitude Difference Function
ANN	Artificial Neural Network
AOT	Acoustic Onset Time
API	Application Programmer's Interface
BPM	Beats per Minute
CAMEL	Content-based Audio and Music Extraction Library
CASA	Computational Auditory Scene Analysis
CCF	Cross Correlation Function
CCIR	Comité Consultatif International des Radiocommunications
CD	Compact Disc
CiCF	Circular Correlation Function
CLAM	C++ Framework for Audio and Music
COG	Center of Gravity
CQT	Constant Q Transform

DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DP	Dynamic Programming
DTW	Dynamic Time Warping
EBU	European Broadcasting Union
ERB	Equivalent Rectangular Bandwidth
FEAPI	Feature Extraction Application Programmer's Interface
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FN	False Negative
FP	False Positive
FT	Fourier Transform
FWR	Full-Wave Rectification
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
HPS	Harmonic Product Spectrum
HSS	Harmonic Sum Spectrum
HTK	HMM Toolkit
HWR	Half-Wave Rectification
IBI	Inter-Beat Interval
ICA	Independent Component Analysis
IDFT	Inverse Discrete Fourier Transform
IFT	Inverse Fourier Transform
IIR	Infinite Impulse Response
IO	Input/Output
IOI	Inter-Onset Interval
ISMIR	International Society for Music Information Retrieval
ITU	International Telecommunication Union
JNDL	Just Noticeable Difference in Level
KNN	K-Nearest Neighbor

LDA	Linear Discriminant Analysis
MA	Moving Average
MFCC	Mel Frequency Cepstral Coefficient
MIDI	Musical Instrument Digital Interface
MIR	Music Information Retrieval
MIREX	Music Information Retrieval Evaluation eXchange
MPA	Music Performance Analysis
MPEG	Motion Picture Experts Group
NOT	Note Onset Time
PAT	Perceptual Attack Time
PCA	Principal Component Analysis
PDF	Probability Density Function
POT	Perceptual Onset Time
PPM	Peak Program Meter
PSD	Peak Structure Distance
RBF	Radial Basis Function
RFD	Relative Frequency Distribution
RLB	Revised Low Frequency B Curve
RMS	Root Mean Square
ROC	Receiver Operating Curve
SIMD	Single Instruction Multiple Data
SNR	Signal-to-Noise Ratio
SOM	Self-Organizing Map
STFT	Short Time Fourier Transform
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
WEKA	Waikato Environment for Knowledge Analysis
YAAFE	Yet Another Audio Feature Extractor

LIST OF SYMBOLS

A	Amplitude
a	Filter Coefficient (recursive)
$\mathcal{B}$	Number of Beats
b	Filter Coefficient (transversal)
β	Exponent
$\mathcal{C}$	Number of (Audio) Channels
$\chi(.)$	Center Clipping Function
$\boldsymbol{C}_{\mathrm{AB}}$	Cost Matrix for the Distance Matrix between Sequences A and B
$\mathfrak{C}_{\mathrm{AB}}$	Overall Cost of a Path through the Cost Matrix
$c_x(\cdot)$	Cepstrum of the Signal x
$\boldsymbol{D}_{\mathrm{AB}}$	Distance Matrix between Sequences A and B
d	Distance Measure
Δ_{Q}	Quantization Step Size
e_{P}	Prediction Error
e_{Q}	Quantization Error
$\mathfrak{e}(f)$	Equivalent Rectangular Bandwidth

η	(Correlation) Lag
e_{Tfp}	(Spectral) Prediction Error
F	F-Measure
f	Frequency in Hz
f_0	Fundamental Frequency in Hz
f_{S}	Sample Rate
f_{A4}	Tuning Frequency in Hz
$\mathcal{F}$	Number of Features
f_I	Instantaneous Frequency in Hz
$\mathfrak{F}(\cdot)$	(Discrete) Fourier Transform
G	Threshold
$\mathbf{\Gamma}$	Chord Transformation Matrix
$\gamma_{x,\mathcal{O}}$	Central Moment of Order $\mathcal{O}$ of Signal x
$H(\cdot)$	Transfer Function
$h(\cdot)$	Impulse Response
$\mathcal{H}$	Hop Size
i	Sample Index
$\mathcal{J}$	Impulse Response Length
j	Integer (Loop) Variable
$\mathcal{K}$	Block Size
k	Frequency Bin Index
κ	Percentage
$\Lambda(k,n)$	Tonalness Spectrum
λ	Weighting Factor
$\mathcal{M}$	Number of (Quantization) Steps
m	Key Index
$\mathfrak{m}(f)$	Pitch (Mel)
μ_x	Arithmetic Mean of Signal x
$\mathcal{N}$	Number of Observations or Blocks
n	Block Index

$\mathcal{O}$	Order (e.g., Filter Order)
o_r	Block Overlap Ratio
ω	Angular Velocity ($\omega = 2\pi f$) in radians per second
O	Number of Onsets
P	Precision
$\boldsymbol{p}$	Alignment Path
Φ_X	Phase Spectrum of the Signal x
$\varphi(\cdot)$	Gaussian Function
$\mathfrak{p}$	(MIDI) Pitch
$\boldsymbol{\nu}$	Pitch Chroma Vector/Key Profile
P_x	Power of the Signal x
$p_x(x)$	Probability Density Function of the Signal x
$\boldsymbol{\psi}(\cdot)$	Chord Probability Vector
$\mathcal{Q}$	Quality Factor (Mid-Frequency divided by Bandwidth)
q	Evaluation Metric
$Q_x(\cdot)$	Quantile Boundary
R	Recall
$r_{xy}(\cdot)$	Correlation Function between the Signals x and y
r	Radius
$\boldsymbol{R}$	Covariance Matrix
σ_x	Standard Deviation of Signal x
σ_x^2	Variance of Signal x
SNR	Signal-to-Noise Ratio
T	Time Period in s
t	Time in s
T_0	Time Period of the Fundamental Frequency in s
$\mathcal{T}$	Number of (Chord) Templates
$\mathfrak{T}$	Tempo in BPM
$\boldsymbol{T}$	(PCA) Transformation Matrix
T_S	Sample Period in s
$\boldsymbol{V}$	Feature Matrix with dimensions $\mathcal{F} \times \mathcal{N}$
v_ACF^η	ηth Autocorrelation Coefficient

v_{C}	Centroid
$\mathcal{V}$	Feature Set
v_{K}	Kurtosis
v^{j}_{MFCC}	jth MFCC
v_{Peak}	Peak Envelope
v_{PPM}	Peak Program Meter
v_{RMS}	RMS
v_{SC}	Spectral Centroid
v_{SD}	Spectral Decrease
v_{SF}	Spectral Flux
v_{SK}	Spectral Kurtosis
v_{Sk}	Skewness
v_{SR}	Spectral Rolloff
v_{SS}	Spectral Spread
v_{SSk}	Spectral Skewness
v_{SSl}	Spectral Slope
v_{Ta}	ACF Maximum
v_{Tf}	Spectral Flatness
v_{Tfp}	Spectral Predictivity
v_{Tp}	Predictivity Ratio
v_{Tpr}	Tonal Power Ratio
v_{Tsc}	Spectral Crest Factor
v_{ZC}	Zero Crossing Rate
w	Word Length in Bit
w_{AB}	Window Function with Alternative Blackman Shape
w_{B}	Window Function with Blackman Shape
w_{BH}	Window Function with Blackman-Harris Shape
w_{C}	Window Function with Cosine Shape
w_{H}	Window Function with von-Hann Shape
w_{Hm}	Window Function with Hamming Shape
w_{R}	Window Function with Rectangular Shape
w_{T}	Window Function with Bartlett Shape
$X(\cdot)$	Fourier Representation of the Signal x
$\mathfrak{x}(f)$	Normed Frequency Position on the Cochlea
$X^{*}(\cdot)$	Conjugate-Complex Spectrum of the Signal x
$\mathfrak{z}(f)$	Critical Band Rate (Bark)

CHAPTER 1

INTRODUCTION

The objective of *Audio Content Analysis (ACA)* is the extraction of information from audio signals such as music recordings stored on digital media. The information to be extracted is usually referred to as *meta data*: it is data about (audio) data and can essentially cover any information allowing a meaningful description or explanation of the raw audio data. The meta data represents (among other things) the musical content of the recording. Nowadays, attempts have been made to automatically extract practically everything from the music recording including formal, perceptual, musical, and technical meta data. Examples range from tempo and key analysis — ultimately leading to the complete transcription of recordings into a score-like format — over the analysis of artists' performances of specific pieces of music to approaches to modeling the human emotional affection when listening to music.

In addition to the meta data extractable from the signal itself there is also meta data which is neither implicitly nor explicitly included in the music signal itself but represents additional information on the signal, such as the year of the composition or recording, the record label, the song title, information on the artists, etc.

The examples given above already imply that ACA is a multi-disciplinary research field. Since it deals with audio signals, the main emphasis lies on (digital) signal processing. But depending on the task at hand, the researcher may be required to use knowledge from different research fields such as musicology and music theory, (music) psychology, psychoacoustics, audio engineering, library science, and last but not least computer science for pattern recognition and machine learning. If the research is driven by commercial interests, even legal and economical issues may be of importance.

An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics, First Edition. Alexander Lerch.

The term *audio content analysis* is not the only one used for systems analyzing audio signals. Frequently, the research field is also called *Music Information Retrieval (MIR)*. MIR should be understood as a more general, broader field of which ACA is a part. Downie and Orio have both published valuable introductory articles in the field of MIR [1, 2]. In contrast to ACA, MIR also includes the analysis of symbolic non-audio music formats such as musical scores and files or signals compliant to the so-called *Musical Instrument Digital Interface (MIDI)* protocol [3]. Furthermore, MIR may include the analysis and retrieval of information that is music-related but cannot be (easily) extracted from the audio signal such as the song lyrics, user ratings, performance instructions in the score, or bibliographical information such as publisher, publishing date, the work's title, etc. Therefore the term audio content analysis seems to be the most accurate for the description of the approaches to be covered in the following. In the past, other terms have been used more or less synonymously to the term audio content analysis. Examples of such synonyms are *machine listening* and *computer audition*. *Computational Auditory Scene Analysis (CASA)* is closely related to ACA but usually has a strong focus on modeling the human perception of audio.

Historically, the first systems analyzing the content of audio signals appear shortly after technology provided the means of storing and reproducing recordings on media in the 20th century. One early example is Seashore's Tonoscope, which allowed one to analyze the pitch of an audio signal by visualizing the fundamental frequency of the incoming audio signal on a rotating drum [4]. However, the development of digital storage media and digital signal processing during the last decades, along with the growing amount of digital audio data available through broadband connections, has significantly increased both the need and the possibilities of automatic systems for analyzing audio content, resulting in a lively and growing research field. A short introduction to extracting information from audio on different levels has been published by Ellis [5].

Audio content analysis systems can be used on a relatively wide variety of tasks. Obviously, the automatic generation of meta data is of great use for the retrieval of music signals with specific characteristics from large databases or the Internet. Here, the manual annotation of meta data by humans is simply not feasible due to the sheer amount of (audio) data. Therefore, only computerized tags can be used to find files or excerpts of files with, e.g., a specific tempo, instrumentation, chord progression, etc. The same information can be used in end consumer applications such as for the automatic generation of play lists in music players or in automatic music recommendation systems based on the user's music database or listening habits. Another typical area of application is music production software. Here, the aim of ACA is on the one hand to allow the user to interact with a more "musical" software interface — e.g., by displaying score-like information along with the audio data — and thus enabling a more intuitive approach to visualization and editing the audio data. On the other hand, the software can support the user by giving suggestions of how to combine and process different audio signals. For instance, software applications for DJs nowadays include technology allowing the (semi-) automatic alignment of audio loops and complete mixes based on previously extracted information such as the tempo and key of the signals. In summary, ACA can help with

- automatic organization of audio content in large databases as well as search and retrieve audio files with specific characteristics in such databases (including the tasks of song identification and recommendation),
- new approaches and interfaces to search and retrieval of audio data such as query-by-humming systems,

- new ways of sound visualization, user interaction, and musical processing in music software such as an audio editor displaying the current score position or an automatically generated accompaniment,
- intelligent, content-dependent control of audio processing (effect parameters, intelligent cross fades, time stretching, etc.) and audio coding algorithms, and
- automatic play list generation in media players.

1.1 Audio Content

The content or information conveyed by recordings of music is obviously multi-faceted. It originates from three different sources:

- *Score*: The term score will be used broadly as a definition of musical ideas. It can refer to any form of notating music from the *basso continuo* (a historic way of defining the harmonic structure) and the classic western score notation to the lead sheet and other forms of notation used for contemporary and popular music.

  Examples of information originating in the score are the melody or hook line, the key and the harmony progression, rhythmic aspects and specific temporal patterns, the instrumentation, as well as structural information such as repetitions and phrase boundaries.

- *Performance*: Music as a performing art requires a performer or group of performers to generate a unique acoustical rendition of the underlying musical ideas. The performers will use the information provided by the score but may interpret and modify it as well as they may dismiss parts of the contained information or add new information.

  Typical performance aspects include the tempo and its variation as well as the micro-timing, the realization of musical dynamics, accents and instantaneous dynamic modulations such as tremolo (see Sect. 4.2), the usage of specific temperaments (see Sect. 5.2.5.2) and expressive intonation and vibrato (see Sect. 5.2.5.3), and specific playing (e.g., bowing) techniques influencing the sound quality.

- *Production*: The process of recording the performance and the (post-) production process will impact certain characteristics of the recording.

  These are mainly the sound quality of the recording (by microphone positioning, equalization, and by applying effects to the signal) and the dynamics (by applying manual or automatic gain adjustments). Changes in timing and pitch may occur as well by editing the recording and applying software for pitch correction.

There are certain characteristics which cannot easily be assigned to a single category; the timbre of a recording can be determined by the instrumentation indicated by the score, by the specific choice of instruments (e.g., historical instruments, specific guitar amps, etc.), by specific playing techniques, and by sound processing choices made by the sound engineer or producer.

ACA systems may in principle cover the extraction of information from all three categories. In many cases, however, no distinction is being made between those categories by researchers and their systems, respectively. The reason is that popular music in the tradition of western music is one of the main targets of the research for several — last but not

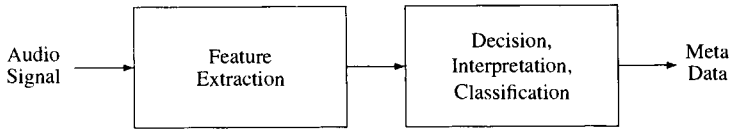

Figure 1.1 General processing stages of a system for audio content analysis

least commercial — reasons and that with popular music a score-like raw representation of musical ideas cannot be distinguished as easily from the performance and production as in "classical" or traditional western music.

From a technical point of view, five general classes can be identified to describe the content of a music recording on a low level:

- *statistical or technical signal characteristics* derived from the audio data such as the amplitude distribution etc. (see Sects. 3.2 and 3.4),
- *timbre or sound quality characteristics* (see Sect. 3.3),
- *intensity-related characteristics* such as envelope-, level-, and loudness-related properties (see Chap. 4),
- *tonal characteristics* which include the pitches and pitch relations in the signal (see Chap. 5), and
- *temporal characteristics* such as rhythmic and timing properties of the signal (see Sect. 6).

The basic information clustered in each individual class can be used and combined to gather a deeper knowledge of the music such as on musical structure, style, performance characteristics, or even transported mood or emotional affection. Especially the last example, however, shows that while many parameters to be extracted from an audio file are objective in a way that they describe music properties independent of the perceptual context (e.g., key, tempo), other properties depend on the individual listener's music experience or way of perceiving music. Not only might this experience vary between the multitude of different listeners, but it might also vary with the listener's individual mood and situation.

1.2 A Generalized Audio Content Analysis System

Most existing systems for the analysis of audio content can be structured into two major processing stages as depicted in Fig. 1.1.

In the first processing stage so-called features are extracted from the audio signal. This extraction process serves two purposes:

- *Dimensionality reduction*: When processing a whole audio file, the raw amount of data in a whole audio file is too large to handle it in a meaningful way. One channel of a digital audio file in Compact Disc (CD) quality (44,100 samples per second, 16 bits per sample) with a length of 5 minutes contains

$$5\,\text{min} \cdot 60\,^{\text{s}}/_{\text{min}} \cdot 44100\,^{\text{samples}}/_{\text{s}} \cdot 16\,^{\text{bits}}/_{\text{sample}} = 211,680,000\,\text{bits}. \quad (1.1)$$

A feature (or a series of featuress) is used to represent this data with fewer values by suppressing (hopefully) irrelevant information. A typical instantaneous feature will produce one single feature value for each block of audio samples or even for the whole signal from beginning to end.

- More *meaningful representation*: Although all the information that can possibly be extracted is implicitly contained in the raw audio data, it is necessary to focus on the relevant aspects and to transform the audio data into a representation easily interpretable by humans or machines. If, for instance, the variation of brightness over time is of interest, one would have difficulties to extract such information by simply observing the series of audio samples. Instead, a model of the human perception of brightness is required, however simple or sophisticated this model may be. In the case of brightness, we would probably be interested in a measure of spectral distribution (see Sect. 3.3).

A feature is not necessarily required to be meaningful in a perceptual or musical way and does not have to be interpretable by humans. It may also just be designed to provide condensed information to the second processing stage of an ACA system to support the generation of a reliable overall result. Usually a distinction is made between low-level features and high-level features. Low-level features are generally considered to have no direct (humanly interpretable) meaning as opposed to high-level features which represent terms in which humans refer to music such as tempo, structure, etc. Those high-level features are usually extracted in the second processing stage shown in Fig. 1.1.

Obviously, the term feature is not very clearly defined but is used for any lower dimensional representation of the audio signal to be interpreted. Features can be used to compute a result but can also be used to calculate derived, more meaningful "features."

The second processing stage of an ACA system takes the extracted feature data and attempts to map it into a domain both usable and comprehensible. Thus, it turns the low-level feature data into a high-level feature and meaningful meta data, respectively. This process can be accomplished by a classification system (sorting the input into pre-defined and trained categories) or by applying (empirical or musical) knowledge to the task.

Since there is no clear objective distinction between low-level features and high-level features, it is sometimes a context-dependent decision whether the system output is referred to as low-level or high-level description. In fact, we face probably an unlimited number of abstraction levels between the raw audio data and the different (human) ways of referring to music. While one system might be referring to the tempo of a recording as high-level information, another system might use this information just as one feature amongst many others to, for example, automatically recognize the musical style. Ultimately, there can only be the conclusion that an ACA system may either consist only of one instance of the two processing stages or of any arbitrary number of nested instances of such processing stages with the output of one instance forming one of the inputs of the following instance.

CHAPTER 2

FUNDAMENTALS

This chapter re-introduces and summarizes some of the important characteristics of digital audio signals and tools used in signal processing and ACA. Furthermore, it will introduce some definitions used in the following chapters. In order to keep this chapter short, some of the definitions may lack derivation since it is not possible to give an extensive introduction to digital audio signal processing in this context. The interested reader may, for instance, refer to the books by Oppenheim and Schafer [6], Ohm and Lüke [7], and Smith [8] for more complete introductions.

2.1 Audio Signals

An *audio signal* as humans perceive it can be described as a function of time-variant sound pressure level. A microphone can be used to convert the sound pressure level into voltage. The signal is defined for all times t and is therefore a (time-) continuous signal $x(t)$.

2.1.1 Periodic Signals

A *periodic signal* repeats itself in constant time intervals. Its periodicity can be formulated by

$$x(t) = x(t + T_0) \tag{2.1}$$

with T_0 being the periodicity interval, or, in other words, the *period length* of the first of

An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics, First Edition. Alexander Lerch.

the harmonics. The *fundamental frequency* of this periodic signal is then

$$f_0 = \frac{1}{T_0}. \tag{2.2}$$

The frequency of other tonal components, the remaining harmonics, is always an integer multiple of the frequency of the first harmonic.

Every periodic signal $x(t)$ can be represented by a superposition of weighted sinusoidal signals, the *Fourier series*

$$x(t) = \sum_{k=-\infty}^{\infty} a(k) e^{\mathrm{j}\omega_0 kt}. \tag{2.3}$$

The periodicity, or the fundamental frequency, of the signal is represented as angular frequency $\omega_0 = 2\pi f_0$. The real part of $e^{\mathrm{j}\omega_0 kt}$ represents the cosine components (even) and its imaginary part the sine components (odd) of the signal: $e^{\mathrm{j}\omega t} = \cos(\omega t) + \mathrm{j}\sin(\omega t)$. The coefficient $a(k)$ is the weight of the kth harmonic and can be calculated by

$$a(k) = \frac{1}{T_0} \int_{-T_0/2}^{T_0/2} x(t) e^{-\mathrm{j}\omega_0 kt}\, dt. \tag{2.4}$$

If the number of frequencies used to represent the signal is limited

$$\hat{x}(t) = \sum_{k=-\mathcal{O}}^{\mathcal{O}} a(k) e^{\mathrm{j}\omega_0 kt}, \tag{2.5}$$

then some periodic signals may be constructed only imperfectly. The larger the order $\mathcal{O}$, the higher is the highest frequency included and the better is the representation of the signal.

Periodic signals featuring "sudden changes" such as discontinuities of the waveform require higher order Fourier coefficients to model the waveform sufficiently well. The modeling of discontinuities and so-called transients requires high frequencies. More systematically we can say that the higher the order of derivations of the signal that are monotone is the lower the required order will be to represent the signal sufficiently well. The most extreme example is a sinusoidal signal which has an infinite number of monotone derivatives and for which only one coefficient $a(1)$ is required to represent the signal perfectly.

Figure 2.1 shows one period of two periodic signals, a *sawtooth* (top) and a *square wave* (bottom), and their representation with the model orders 3, 25, and unlimited. The coefficients $a(k)$ for these two signals are

$$a_{\mathrm{saw}}(k) = \frac{2}{\pi \cdot k}, \tag{2.6}$$

$$a_{\mathrm{rect}}(k) = \frac{4}{\pi \cdot (2k-1)}. \tag{2.7}$$

Due to the discontinuities of these two signals there are model errors in the form of overshoots around the point of discontinuity. It can be observed that the frequency of the overshoots increases and the duration of the overshoots decreases with increasing model order. However, the amplitude of the overshoots stays constant unless the order is infinite. This is called *Gibbs' phenomenon* [9, 10].

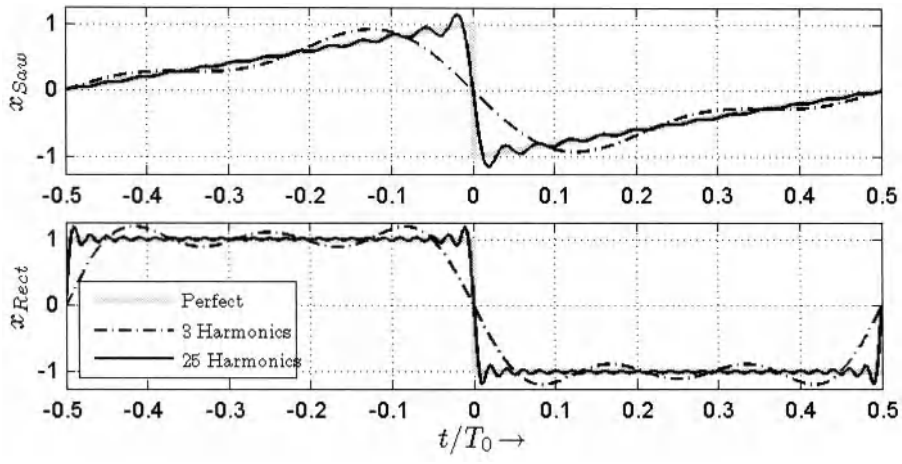

Figure 2.1 Approximation of periodic signals: sawtooth (top) and square reconstructed with 3 and 25 sinusoidal harmonics

2.1.2 Random Signals

In contrast to periodic signals, future values of *random signals* cannot be predicted no matter how long the signal has been observed. That means that there exists no fundamental frequency for this type of signal. Every observation of such a signal is therefore only an example of an unlimited number of possible incarnations of the signal. A complete signal description is theoretically only possible with an unlimited number of observations.

An important subcategory of random signals is built of *stationary* signals for which basic properties (such as arithmetic mean and higher central moments, see below) do not change over time. *White noise* is a typical example of a stationary random signal.

2.1.3 Sampling and Quantization

In the digital domain, we cannot deal with continuous signals. Therefore the analogue, continuous signal $x(t)$ has to be discretized in both amplitude and time, where *quantization* refers to the discretization of amplitudes and *sampling* refers to the discretization in time. The resulting signal is a series of quantized amplitude values $x(i)$.

2.1.3.1 Sampling

The discretization in time is done by *sampling* the signal at specific times. The distance T_S in seconds between sampling points is equidistant in the context of audio signals and can be derived directly from the *sample rate* f_S by

$$T_S = \frac{1}{f_S}. \tag{2.8}$$

Figure 2.2 visualizes the process of sampling by plotting a continuous signal (top) and a sampled representation of it (bottom) sampled at a sample rate of $f_S = 700$ Hz.

An important property of sampled signals is that the sampled representation is ambiguous as exemplified in Fig. 2.3. This example shows that different input signals may lead to the same series of samples.

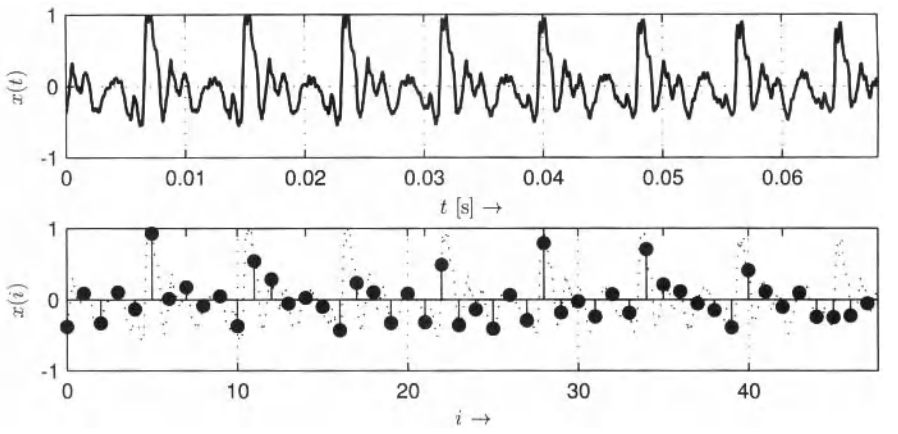

Figure 2.2 Continuous audio signal (top) and corresponding sample values at a sample rate of $f_S = 700$ Hz)

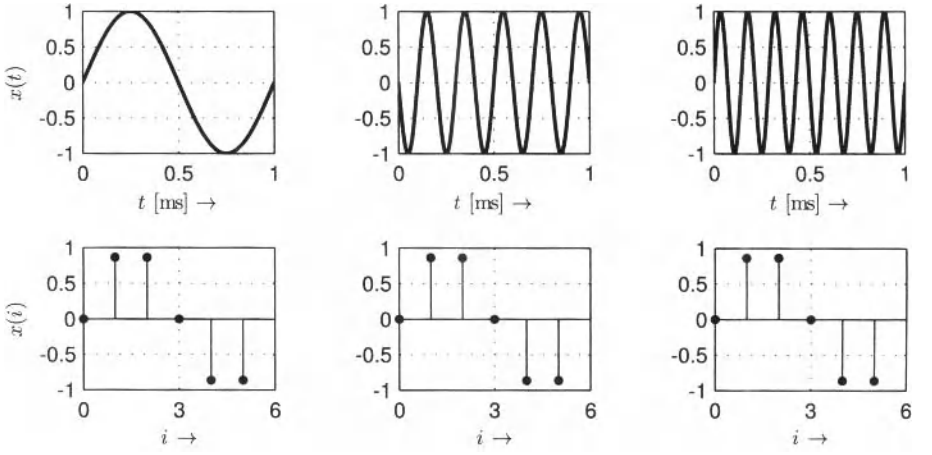

Figure 2.3 Continuous (top) and sampled (below) sinusoidal signals with the frequencies 1 kHz (left), 5 kHz (mid), 7 kHz (right). The sample rate is $f_S = 6$ kHz

The signal can only be reconstructed unambiguously when certain requirements for audio signal and sample rates are met as defined by the *sampling theorem*:

Sampling Theorem

A sampled signal can only be reconstructed without loss of information if the sample rate f_S is higher than twice the highest frequency f_{max} in the sampled audio signal.

$$f_S > 2 \cdot f_{max} \tag{2.9}$$

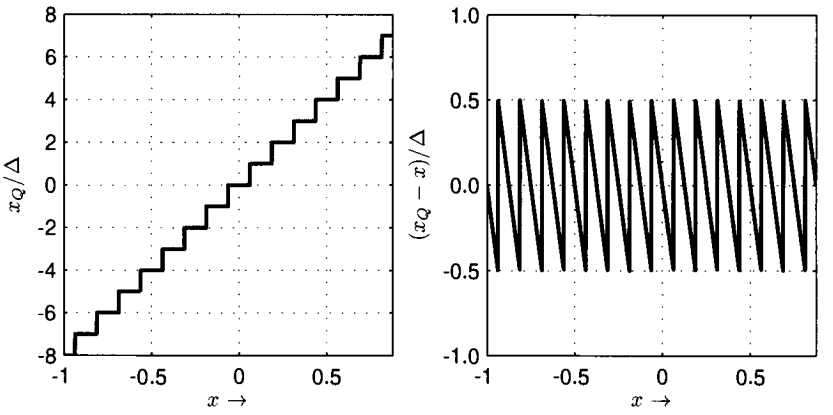

Figure 2.4 Characteristic line of a quantizer with word length $w = 4$ bits showing the quantized output amplitude for a normalized input amplitude (left) and the quantization error with respect to the input amplitude (right)

Historically the sampling theorem is attributed to Kotelnikov [11] and Shannon [12] and is based on work by Whittaker [13] and Nyquist [14].

If the signal contains frequency components higher than half the sample rate, these components will be mirrored back and *aliasing* occurs. See also Appendix B.3 for further explanation of this artifact.

2.1.3.2 Quantization

In order to discretize the signal amplitudes the signal is quantized. *Quantization* means that each amplitude of the signal is rounded to a pre-defined set of allowed amplitude values. Usually, the input amplitude range between the maximum and the minimum amplitude is assumed to be symmetric around 0 and is divided into $\mathcal{M}$ steps. If $\mathcal{M}$ is a power of 2, the amplitude of each quantized sample can easily be represented with a binary code of *word length*

$$w = \log_2(\mathcal{M}). \tag{2.10}$$

Since $\mathcal{M}$ is an even number, it is impossible to both represent zero amplitude as a quantization step and maintain a symmetric number of quantization steps above and below zero. The usual approach is to have one step less for positive values. Typical word lengths for audio signals are 16, 24, and 32 bits. Figure 2.4 shows the characteristic line of a quantizer and the corresponding *quantization error* in dependence of the input amplitude.

The distance Δ_Q between two neighboring quantization steps is usually constant for all steps. That means that the quantization error e_Q is always in the range $[-\Delta_Q/2; \Delta_Q/2]$ if the maximum input signal amplitude is within the range spanned by the maximum allowed quantized amplitudes (no *clipping*). The quantization error is the difference between original signal x and quantized signal x_Q:

$$e_Q(i) = x(i) - x_Q(i). \tag{2.11}$$

If the input signal amplitudes are much larger than the step size Δ_Q but in the range of allowed amplitudes, the distribution of quantization error amplitudes can be assumed to

be rectangular. The power of the quantization error P_Q can then be estimated with its *Probability Density Function (PDF)* $p_q(e_Q)$ (compare Sect. 2.1.4.1):

$$P_Q = \int_{-\infty}^{\infty} e_Q^2 \cdot p_q(e_Q)\, de_Q = \frac{1}{\Delta_Q} \int_{-\Delta_Q/2}^{\Delta_Q/2} e_Q^2\, de_Q = \frac{\Delta_Q^2}{12}. \tag{2.12}$$

It is obvious that the power of the quantization error will be lower the smaller the step size Δ_Q and the higher the word length w is, respectively. The *Signal-to-Noise Ratio (SNR)* in decibels is a good criterion of the quality of a quantization system. The SNR can be calculated from the ratio of the signal power P_S and the quantization error power P_Q by

$$\mathrm{SNR} = 10 \log_{10} \left(\frac{P_S}{P_Q} \right) \text{ [dB]}. \tag{2.13}$$

A high SNR thus indicates good quantization quality. Using Eqs. (2.12) and (2.13) it follows that increasing the word length by one bit gains approximately 6 dB:

Signal-to-Noise Ratio (SNR)

$$\mathrm{SNR} = 6.02 \cdot w + c_S \text{ [dB]} \tag{2.14}$$

The constant c_S depends on the signal's amplification and PDF:

- square wave (full scale): $c_S = 10.80\,\mathrm{dB}$
- sinusoidal wave (full scale): $c_S = 1.76\,\mathrm{dB}$
- rectangular PDF (full scale): $c_S = 0\,\mathrm{dB}$
- Gaussian PDF (full scale = $4\sigma_g$)[1]: $c_S = -7.27\,\mathrm{dB}$

Assuming that a real-world audio signal is closer to the PDF of a noise with Gaussian PDF than to that of a sine wave, the typical SNR is approximately 8 dB lower than commonly stated. Furthermore, the above reference values are given for signals that are leveled more or less optimally; if the input signal is scaled, the SNR will decrease accordingly (by 6.02 dB per scaling factor $1/2$).

The term *full scale* refers to the highest quantization step. A full-scale sine wave will thus have an amplitude equaling the highest quantization step (compare also page 72).

These considerations are often only of limited use to the signal processing algorithm designer, as the quantized signal is usually converted to a signal in floating point format, effectively resulting in a non-linear characteristic line of the quantizer. The floating point stores the number's mantissa and exponent separately so that the quantization step size Δ_Q basically increases with the input amplitude. In practice the signals in floating point format are just used as if they had continuous amplitude values, although the quantization error still persists. The choice of a maximum amplitude is arbitrary in floating point format; the most common way is to map full scale to the range $[-1; 1]$ to make the processing independent of the quantizer's word length.

[1] Using the approximation that no values are clipped.

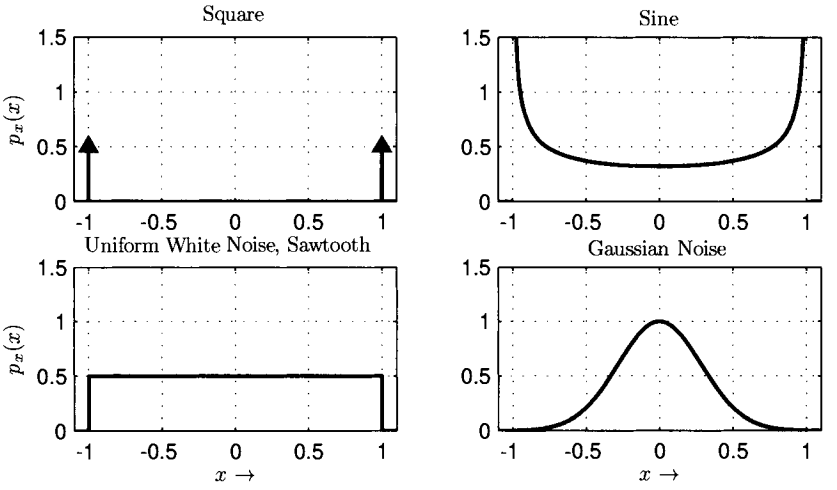

Figure 2.5 Probability density function of a square wave (top left), a sinusoidal (top right), uniform white noise or a sawtooth wave (bottom left), and Gaussian noise (bottom right)

2.1.4 Statistical Signal Description

In contrast to sinusoidal signals, noise cannot be described in the time domain with analytical functions. Statistics can help to identify some properties that describe the signal when neglecting its evolution in time. One of the most common representations is the PDF, which is a useful tool to describe stationary processes or signals that have the same properties for any point in time t.

2.1.4.1 Probability Density Function

The *PDF* $p_x(x)$ of a signal is the probability distribution of a signal. The abscissa of a PDF plot represents all possible amplitude values of the signal x and their probability is plotted on the ordinate. For example, a rectangular PDF means that all possible signal amplitudes occur with the same probability. The properties of the PDF are

$$p_x(x) \geq 0, \text{ and} \tag{2.15}$$

$$\int_{-\infty}^{\infty} p_x(x)\,dx = 1. \tag{2.16}$$

A set of prototypical PDFs is shown in Fig. 2.5. For a full-scale square wave, the PDF has only two peaks at maximum and minimum amplitudes while other amplitude values do not exist. A sine wave has a PDF in which values similar to the arithmetic mean (here: 0) are less frequent than values far from the mean. A uniform PDF can be found with noise generated with the `rand` function of different programming languages; a sawtooth has such a uniform PDF as well. A *Gaussian distribution*

$$p_g = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu_x)^2}{2\sigma^2}\right) \tag{2.17}$$

is assumed to be a typical distribution of "real-world" noise; a typical music signal has in many cases a distribution shaped similar to a *Laplace distribution* with a prominent peak

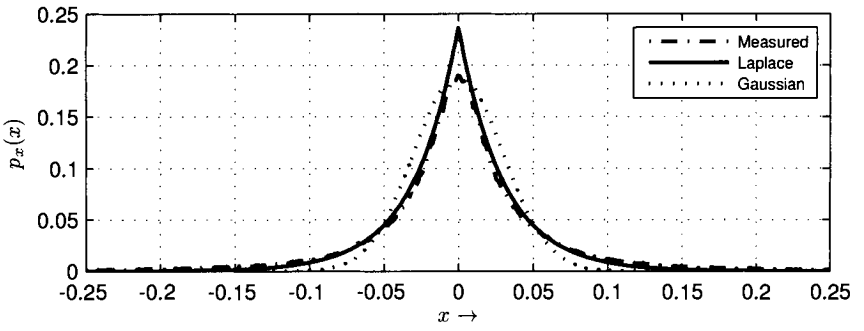

Figure 2.6 Probability density function estimated from a music signal compared to a Laplace distribution

at 0 amplitude (compare Fig. 2.6). The Laplace distribution is given by

$$p_l = \frac{1}{2\beta} \exp\left(-\frac{|x - \mu_x|}{\beta}\right). \tag{2.18}$$

If the input signal is a constant, for example, a DC offset, the PDF consists only of a single peak.

Comparing Figs. 2.5 (top right) and 2.6 makes it obvious that under the assumption that a PDF describes a signal sufficiently well, a sine wave is not a very good approximation of a real-world music or speech signal.

The PDF of quantized input signals has a limited set of amplitude classes as a quantized signal has only a limited set of amplitude values.

The PDF of a signal can be estimated from a sufficiently long block of samples by computing a histogram of signal amplitudes and dividing it by the number of observed samples. It is then sometimes referred to as *Relative Frequency Distribution (RFD)*. For the sake of simplicity the PDF and the RFD will not be differentiated in the following.

2.2 Signal Processing

The following chapters introduce methods for processing digital signals. There exists a vast amount of introductory literature on *digital signal processing* — as mentioned above we will not attempt to reiterate the theory and methods but instead give short summaries on the most important foundations for ACA.

2.2.1 Convolution

Every linear, time-invariant system can be completely described by its *impulse response* $h(i)$, which is the output of a system for an impulse as an input signal. Examples of such systems (or systems which can be approximated as such systems) are filters, speakers, microphones, rooms, etc. The output $y(i)$ of a discrete linear and time-invariant system can be computed from input signal $x(i)$ and the system's impulse response $h(i)$ of length

Convolution Properties

- Identity for convolution with the delta function $\delta(i)$:

$$x(i) = x(i) * \delta(i) \tag{2.19}$$

- Commutativity:

$$y(i) = x(i) * h(i) = h(i) * x(i) \tag{2.20}$$

- Associativity:

$$g(i) * h(i) * x(i) = \big(g(i) * h(i)\big) * x(i) = g(i) * \big(h(i) * x(i)\big) \tag{2.21}$$

- Distributivity:

$$g(i) * \big(h(i) + x(i)\big) = \big(g(i) * h(i)\big) + \big(g(i) * x(i)\big) \tag{2.22}$$

- Circularity: If $h(i)$ is periodic, then the convolution result $y(i) = h(i) * x(i)$ will also be periodic.

$\mathcal{J}$ by the *convolution* operation:

$$y(i) = x(i) * h(i) = \sum_{j=0}^{\mathcal{J}-1} h(j) \cdot x(i-j). \tag{2.23}$$

The most important properties of the convolution operation are summarized in Eqs. (2.19)–(2.22). Derivations of these properties can be found in Appendix A.

2.2.1.1 Simple Filter Examples

One typical linear system in digital signal processing is the filter. Every filtering process is a convolution, but for practical filter implementations this is only of interest for filters with a finite length impulse response, so-called *Finite Impulse Response (FIR)* filters. Filters with an infinite length impulse response are unsurprisingly referred to as *Infinite Impulse Response (IIR)* filters and have a feedback path which complicates the determination of the impulse response in comparison with FIR filters for which the impulse response simply equals its filter coefficients.

In the following, two simple and frequently used digital low-pass filters will be presented, the *Moving Average (MA)* filter as a prototypical FIR filter and the single-pole filter as a simple IIR filter.

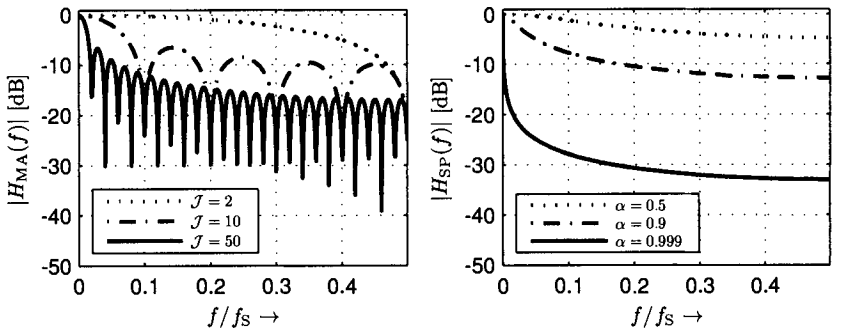

Figure 2.7 Magnitude frequency response of a moving average low-pass filter with the impulse response lengths of 2, 10, and 50 (left) and frequency response of a single-pole low-pass filter with an α of 0.5, 0.9, and 0.999 (right)

Moving Average Filter

The filter equation for an MA filter with an impulse response of length $\mathcal{J}$ is

$$y(i) = \sum_{j=0}^{\mathcal{J}-1} b(j) \cdot x(i-j). \tag{2.24}$$

The coefficients $b(j)$ of a typical MA filter have two main properties; the filter coefficients are identical:

$$b(0) = b(j) \quad \text{for } 0 \leq j \leq \mathcal{J} - 1, \tag{2.25}$$

and the sum of all coefficients is normalized to 1:

$$\sum_{j=0}^{\mathcal{J}-1} b(j) = 1. \tag{2.26}$$

Equations (2.25) and (2.26) result in the coefficients $b(j)$ being identical and normalized to $b(j) = 1/\mathcal{J}$. Alternatively they can also be weighted by an arbitrary window function. Typical window functions are symmetric and weight center samples higher than lateral samples. The window shape allows to elongate the impulse response. One of the simplest window shapes would be triangular.

Applying an MA filter with unweighted coefficients twice to the signal is equivalent to applying an MA filter with a triangular shaped impulse response of double length. Thus, the repeated use of an MA filter is similar to increasing the impulse response length and applying a window function to the coefficients.

The MA filter's cut-off frequency, the frequency at which the magnitude first drops by 3 dB, will decrease with an increasing number of coefficients as depicted in Fig. 2.7 (left). Smith gives a detailed introduction to MA filters in [15].

The number of multiply and add operations per hop increases linearly with the impulse response length of the MA filter. If no window function had been applied to the coefficients, i.e., all coefficients are identical $b(k) = b$, the number of operations can be

drastically reduced by implementing the filter recursively:

$$\begin{aligned} y(i) &= \sum_{j=0}^{\mathcal{J}-1} b \cdot x(i-j) \\ &= b \cdot \big(x(i) - x(i-\mathcal{J})\big) + \underbrace{\sum_{j=1}^{\mathcal{J}} b \cdot x(i-j)}_{y(i-1)} \\ &= b \cdot \big(x(i) - x(i-\mathcal{J})\big) + y(i-1). \end{aligned} \tag{2.27}$$

A recursive implementation is not applicable with weighted filter coefficients.

If the filter coefficients of the MA filter are symmetric with $h(i) = h(\mathcal{J}-1-i)$ (which is only possible for an FIR filter), then the filter will have a linear phase response and thus a constant group delay.

Single-Pole Low-Pass Filter

The *single-pole filter* is — mainly due to its simplicity and efficiency — one of the most frequently used smoothing and low-pass filter. The filter equation is

$$y(i) = \alpha \cdot y(i-1) + (1-\alpha) \cdot x(i) \quad \text{with } 0 \leq \alpha < 1. \tag{2.28}$$

Figure 2.7 (right) shows the magnitude of the frequency response for three different α. The coefficient α depends on the required integration time T_I; if the integration is defined to be the time required for the filter's response on a step function to rise from 10% to 90%,[2] the coefficient is

$$\alpha = \exp\left(\frac{-2.2}{f_\mathrm{S} \cdot T_\mathrm{I}}\right). \tag{2.29}$$

The cut-off frequency is then

$$\omega_0(\alpha) = \arccos\big(2 - \cosh\left(\log(\alpha)\right)\big) \tag{2.30}$$

which is only defined for $\alpha \geq e^{-\operatorname{arccosh}(3)}$.

2.2.1.2 Zero Phase Filtering with IIRs

No non-trivial causal system provides a *zero phase response*. A zero phase response means that the group delay at each frequency is zero as well. Zero phase response systems output the unmodified timing characteristics of each individual frequency group.

In the case of linear phase FIR filters a zero phase response can be reconstructed in an anti-causal way by removing the introduced delay from the filter output (which could also be interpreted as moving the impulse response to be symmetric around time 0).

In the case of IIR filters, however, this procedure is not applicable. In a non-real-time environment in which the whole input file is available, it is, however, possible to generate a filtered signal with zero phase shift by using an IIR filter. This can be done by applying the filter twice in series to the signal while using the time-reversed input signal the second time. The second filter run has the effect of leveling out the phase shifts introduced by the first run. The magnitude response of the complete filter process is then the filter's squared magnitude response. Figure 2.8 visualizes the intermediate and the final result of this process in the time domain.

[2]Note that there exist other approaches to defining the integration time that may differ significantly.

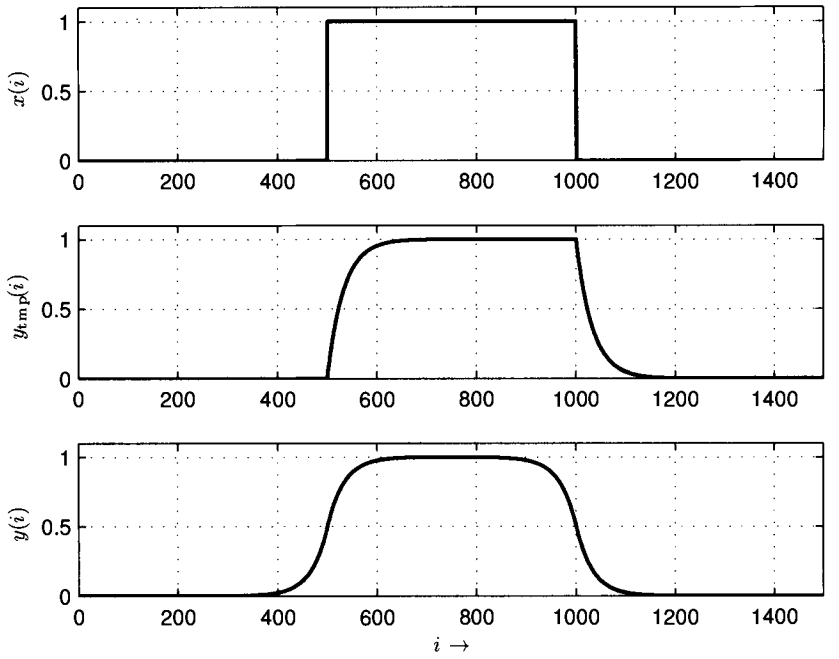

Figure 2.8 Example of zero phase filtering: input signal (top), low-pass filtered signal after the first forward pass (mid), and final result of both forward and backward paths (bottom)

To explain this behavior mathematically, the following property is of importance:

$$\mathfrak{F}\{h(-i)\} = H(-\mathrm{j}\omega), \tag{2.31}$$

meaning that the *Fourier Transform (FT)* $\mathfrak{F}$ (see Sect. 2.2.3) of a time-reversed signal equals the complex-conjugate of the transform of the original signal. The flip function is a tool which helps us in the derivation:

$$\text{flip}\big(x(i)\big) = x(-i). \tag{2.32}$$

The output signal $y(i)$ is computed via the temporary output $y_{\text{tmp}}(i) = h(i) * x(i)$ by

$$\begin{aligned} y(i) &= \text{flip}\big(h(i) * y_{\text{tmp}}(-i)\big) && (2.33)\\ &= h(-i) * y_{\text{tmp}}(i) && (2.34)\\ &= h(-i) * \big(h(i) * x(i)\big). && (2.35) \end{aligned}$$

It follows in the Fourier domain:

$$\begin{aligned} Y(\mathrm{j}\omega) &= H(-\mathrm{j}\omega) \cdot \big(H(\mathrm{j}\omega) \cdot X(\mathrm{j}\omega)\big) && (2.36)\\ &= |H(\mathrm{j}\omega)|^2 \cdot X(\mathrm{j}\omega). && (2.37) \end{aligned}$$

2.2.2 Block-Based Processing

Signal processing algorithms usually work block-based, meaning that the input signal is being split into consecutive blocks of frames with length K. These blocks are processed

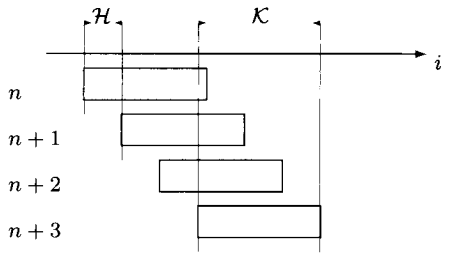

Figure 2.9 Schematic visualization of block-based processing: the input signal with the input sample index i is split into overlapping blocks of length $\mathcal{K}$ and block index n; the hop size $\mathcal{H}$ is the distance from the start of one block and the following block

one after another. In extreme cases, the block lengths can be either $\mathcal{K} = 1$ or equal the length of the processed audio file, but in most cases $\mathcal{K}$ is chosen arbitrarily between $\mathcal{K} = 32$ and $\mathcal{K} = 16,384$ (in many cases powers of 2 are an advantage). Some reasons for this *block-based processing* are

- the internal usage of block-based algorithms such as the *Discrete Fourier Transform (DFT)* (see Sect. 2.2.3 and Appendix B),
- the characteristics of the used audio *Input/Output (IO)* hardware that usually returns blocks of samples,
- reduced memory allocation (instead of the complete audio file length; only the block length has to be allocated), and
- computational efficiency compared to individual sample processing by avoiding function call overhead and vectorization with *Single Instruction Multiple Data (SIMD)* operations to optimize workload.

Systems working in *real time* have the constraint that the system has to be able to perform the processing of one block of data during the time frame this block consumes. Another condition of such systems is that they have no access to future input samples but only the present and possibly past samples. For real-time systems, the *latency* of the system, i.e., the delay between an input sample and the corresponding output value, can never be lower than the block length of the algorithm. The block length used to call the algorithm does not necessarily correspond to the algorithm's internal block length. If the input block length is not a multiple of the internal block length, then efficiency problems can arise since the computational workload cannot be distributed uniformly over the input blocks, resulting in occasional workload peaks.

Internally, most audio algorithms use overlapping blocks of data as depicted in Fig. 2.9. That means that the boundaries of subsequently processed blocks overlap, i.e. that the last sample of the first block is after the first sample of the second block. The sample indices of the block boundaries start sample $i_\mathrm{s}(n)$ and stop sample $i_\mathrm{e}(n)$ of processing block n can then be formulated as

$$i_\mathrm{s}(n) = i_\mathrm{s}(n-1) + \mathcal{H}, \tag{2.38}$$

$$i_\mathrm{e}(n) = i_\mathrm{s}(n) + \mathcal{K} - 1, \tag{2.39}$$

with $\mathcal{H}$ being the so-called *hop size* in samples. The hop size should be smaller than or equal the block length. The *block overlap ratio* can be calculated with

$$o_\mathrm{r} = \frac{\mathcal{K} - \mathcal{H}}{\mathcal{K}}. \tag{2.40}$$

A typical block overlap ratio is $o_\mathrm{r} \geq 1/2$. When only one result $v(n)$ is computed per processing block, the assigned time stamp in samples $t_\mathrm{s}(n)$ would usually be either the start frame of the block $t_\mathrm{s}(n) = {}^{i_\mathrm{s}(n)}/_{f_\mathrm{S}}$ or — more common — its middle position:

$$t_\mathrm{s}(n) = \frac{i_\mathrm{e}(n) - i_\mathrm{s}(n) + 1}{2 \cdot f_\mathrm{S}} + \frac{i_\mathrm{s}(n)}{f_\mathrm{S}} = \frac{\mathcal{K}}{2 \cdot f_\mathrm{S}} + \frac{i_\mathrm{s}(n)}{f_\mathrm{S}}. \tag{2.41}$$

If the time stamp $t_\mathrm{s}(n)$ is chosen to be in the middle of the block, the practical problem arises that the start and end time stamps always represent a shorter time range than the audio signal itself. For example, if the audio signal starts at sample index 0, the first time stamp will be at ${}^{(\mathcal{K}-1)}/_2$. This is usually avoided by padding the signal with ${}^{(\mathcal{K}-1)}/_2$ zeros at beginning and end. However, in that case the results may not be reliable for the first and the last block because the computation is done with a smaller number of audio samples.

Some algorithms do not only compute results per block of audio data but combine the extracted results per block $v(n)$ again in a *texture window*. Texture windows may in turn overlap depending on texture window length and texture window hop. In this case, the same logic as above is followed with the difference that the window boundaries are now block indices instead of frame indices. Texture windows can be used to compute, e.g., windowed statistical descriptions from the series of results $v(n)$.

2.2.3 Fourier Transform

The *DFT* is one of the most important tools for processing and analyzing audio signals. In the following, the important properties and characteristics of this widely used transform are listed. A more detailed explanation — starting with the FT of continuous signals — can be found in Appendix B. The DFT of a block of the signal $x(i)$ is defined by

$$X(\mathrm{j}\Delta\Omega) = \mathfrak{F}\{x(i)\} = \sum_{i=0}^{\mathcal{K}-1} x(i) e^{-\mathrm{j}ki\Delta\Omega} \tag{2.42}$$

with $\Delta\Omega$ being the distance between two frequency bins as angular frequency difference

$$\Delta\Omega = \frac{2\pi}{\mathcal{K} \cdot T_\mathrm{S}} = \frac{2\pi \cdot f_\mathrm{S}}{\mathcal{K}}. \tag{2.43}$$

The frequency of bin index k can then be computed from the block length $\mathcal{K}$ and sample rate f_S by

$$f(k) = \frac{\Delta\Omega}{2\pi} k = \frac{f_\mathrm{S}}{\mathcal{K}} k. \tag{2.44}$$

We will refer to the DFT of the nth block with length $\mathcal{K} = i_\mathrm{e}(n) - i_\mathrm{s}(n) + 1$ of the audio data as *Short Time Fourier Transform (STFT)*:

$$X(k, n) = \sum_{i=i_\mathrm{s}(n)}^{i_\mathrm{e}(n)} x(i) \exp\left(-\mathrm{j}k \cdot (i - i_\mathrm{s}(n)) \frac{2\pi}{\mathcal{K}}\right). \tag{2.45}$$

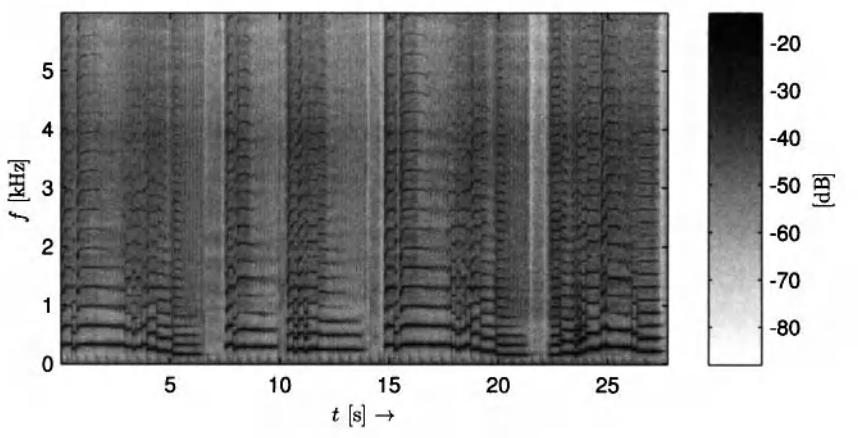

Figure 2.10 Spectrogram of a monophonic saxophone signal: each column is (the magnitude of) an STFT of a block with the time increasing from left to right; the level of each specific point in frequency and time is visualized by the darkness of the point

The most important properties of the STFT are listed on Pg. 22. See Appendix B for derivations.

A typical visualization of an audio signal that combines time and frequency components is the *spectrogram* as shown in Fig. 2.10. An STFT is calculated for each (overlapping) block of sample data; the resulting STFTs are then plotted in a pseudo-three-dimensional image in which (the magnitude of) each STFT is represented by a column and each bin is darkened according to its level.

2.2.3.1 Frequency Reassignment: Instantaneous Frequency

Frequency reassignment is a method to virtually improve the spectral resolution by mapping the frequency data to coordinates closer to its "true" coordinates. Frequency reassignment is not a fundamental frequency detection technique but can help to increase the frequency accuracy of spectral frequency analysis by utilizing the phase spectrum.

The reassignment process may basically cover both frequency reassignment as well as time reassignment. For an overview of (time-) frequency reassignment, see Fulop and Fitz [16] and Lagrange [17]. The focus will be exclusively on frequency reassignment in the following.

The basic idea of frequency reassignment is that it is possible to estimate the frequency of a signal via its phase. The frequency is the derivative of the phase $\Phi(k,t)$:

$$\omega(k,t) = \frac{\partial}{\partial t}\Phi(k,t). \tag{2.46}$$

Two common approaches are used to compute the so-called *instantaneous frequency* from the discrete STFT. The first one is to literally compute the phase difference of consecutive STFT spectra, and the second one is to compute two STFTs with different windows, namely with the second window being the derivative of the first window.

Fourier Transform Properties

- Invertibility

$$x\big(i_s(n)\ldots i_e(n)\big) = \mathfrak{F}^{-1}\{X(k,n)\} = \frac{1}{\mathcal{K}}\sum_{0}^{\mathcal{K}-1} X(k,n)\cdot\exp\left(\mathrm{j}ki\frac{2\pi}{\mathcal{K}}\right). \tag{2.47}$$

- Linearity and Superposition

$$\begin{aligned}\mathfrak{F}\{c_1\cdot x\big(i_s(n)\ldots i_e(n)\big) + c_2\cdot y\big(i_s(n)\ldots i_e(n)\big)\}\\ = c_1\cdot X(k,n) + c_2\cdot Y(k,n).\end{aligned} \tag{2.48}$$

- Periodicity

$$X(k+\mathcal{K},n) = X(k,n). \tag{2.49}$$

- Symmetry [for real $x(i)$]

$$X(\mathcal{K}-k,n) = X^*(k,n). \tag{2.50}$$

- Circularity

$$\mathfrak{F}^{-1}\{H(k,n)\cdot X(k,n)\} = h'\big(i_s(n)\ldots i_e(n)\big) * x\big(i_s(n)\ldots i_e(n)\big) \tag{2.51}$$

with h' being a *periodically extended* sequence of the signal block with the sample boundaries $i_s(n)$ and $i_e(n)$.

- Time and Frequency Shifting

$$\mathfrak{F}\{x(i-i_0)\} = X(k,n)\cdot\exp\left(-\mathrm{j}\frac{2\pi k}{\mathcal{K}}i_0\right) \tag{2.52}$$

$$\mathfrak{F}^{-1}\{X(k-k_0,n)\} = x\big(i_s(n)\ldots i_e(n)\big)\cdot\exp\left(\mathrm{j}\frac{2\pi i}{\mathcal{K}}k_0\right). \tag{2.53}$$

- Time and Frequency Scaling

$$\mathfrak{F}\{x(c\cdot i)\} = \frac{1}{|c|}X\left(\frac{k}{c},n\right). \tag{2.54}$$

- Parseval's Theorem

$$\sum_{i=i_s(n)}^{i_e(n)} |x(i)|^2 = \frac{1}{\mathcal{K}}\sum_{k=0}^{\mathcal{K}-1} |X(k,n)|^2. \tag{2.55}$$

- Duality

$$\mathfrak{F}\{X(i)\} = \frac{1}{\mathcal{K}}x(k,n). \tag{2.56}$$

Explicit Phase Difference

The derivation operation from Eq. (2.46) can be easily approximated by computing the difference of the phases of consecutive STFT phases:

$$\omega_{\mathrm{I}}(k,n) = \frac{\Delta\Phi_{\mathrm{u}}(k,n)}{\mathcal{H}} \cdot f_{\mathrm{S}}. \tag{2.57}$$

The actual computation, however, is not as trivial as it seems at first glance as $\Delta\Phi_{\mathrm{u}}(k,n)$ represents the *unwrapped* phase difference while the computed phase spectrum gives the *wrapped* phase in the range of $]-\pi;\pi]$. In order to estimate the unwrapped phase difference we first compute the estimate of the unwrapped phase of the current analysis block by

$$\hat{\Phi}(k,n) = \Phi(k,n-1) + \frac{2\pi k}{\mathcal{K}}\mathcal{H}. \tag{2.58}$$

Since the phase $\Phi(k,n)$ is in the range of $]-\pi;\pi]$, the unwrapped phase can then be formulated as

$$\Phi_{\mathrm{u}}(k,n) = \hat{\Phi}(k,n) + \mathrm{princarg}\left[\Phi(k,n) - \hat{\Phi}(k,n)\right] \tag{2.59}$$

with the princarg function being the principal argument of the phase with a resulting range of $]-\pi;\pi]$. The phase difference is then

$$\begin{aligned}\Delta\Phi_{\mathrm{u}}(k,n) &= \Phi_{\mathrm{u}}(k,n) - \Phi(k,n-1)\\ &= \hat{\Phi}(k,n) + \mathrm{princarg}\left[\Phi(k,n) - \hat{\Phi}(k,n)\right] - \Phi(k,n-1).\end{aligned}$$

The final result can be retrieved by substituting $\hat{\Phi}(k,n)$ with Eq. (2.58):

$$\Delta\Phi_{\mathrm{u}}(k,n) = \frac{2\pi k}{\mathcal{K}}\mathcal{H} + \mathrm{princarg}\left[\Phi(k,n) - \Phi(k,n-1) - \frac{2\pi k}{\mathcal{K}}\mathcal{H}\right]. \tag{2.60}$$

To improve the reliability of the phase unwrapping process the hop size $\mathcal{H}$ should be as small as possible.

Phase Difference Using Transforms

An alternative to computing the phase difference directly is to use two different windows for computing two STFTs of one analysis block. Then, the instantaneous frequency can be computed by

$$\omega_{\mathrm{I}}(k,n) = \omega(k) + \Im\left\{\frac{X_D(k,n)\cdot X^*(k,n)}{|X(k,n)|^2}\right\} \tag{2.61}$$

with $X_D(k,n)$ being the STFT computed using the derivative of the window used for the calculation of $X(k,n)$ [18]. This approach is independent of the hop size but requires the computation of two DFTs per block.

2.2.4 Constant Q Transform

The *Constant Q Transform (CQT)* was introduced to overcome the insufficient frequency resolution of the STFT at low frequencies for common STFT lengths. The CQT calculates frequency coefficients similar to the DFT but on a logarithmic frequency scale [19]:

$$X(k,n) = \frac{1}{\mathcal{K}(k)} \sum_{i=i_{\mathrm{s}}(n)}^{i_{\mathrm{e}}(n)} w_k(i-i_{\mathrm{s}}) \cdot x(i) \exp\left(\mathrm{j}2\pi \frac{\mathcal{Q}\cdot(i-i_{\mathrm{s}})}{\mathcal{K}(k)}\right). \tag{2.62}$$

$\mathcal{Q}$ adjusts the desired frequency resolution per octave c, while $\mathcal{K}(k)$ depends on both the target frequency and the *quality factor* $\mathcal{Q}$:

$$\mathcal{Q} = \frac{f}{\Delta f} = \frac{1}{2^{1/c} - 1}, \tag{2.63}$$

$$\mathcal{K}(k) = \frac{f_S}{f(k)} \mathcal{Q}. \tag{2.64}$$

The CQT can be interpreted as computing a DFT only for specific, logarithmically spaced, frequency bins. It is not invertible and can in terms of efficiency be compared to a DFT [as opposed to the *Fast Fourier Transform (FFT)*]. There are approaches to make the CQT both computationally more efficient and quasi-invertible, see, e.g., [20, 21].

2.2.5 Auditory Filterbanks

An *auditory filterbank* is frequently used in psycho-acoustical or physiological ear models. It approximates the resolution and selectivity of the human ear. The filterbank's mid-frequencies are usually computed according to the mel scale (see Sect. 5.1).

In signal processing applications the usage of auditory filterbanks is not as widespread as one might assume. The most obvious reasons are (a) the computational workload produced by a filterbank with reasonable frequency resolution and (b) the restriction to analysis tasks since the computed frequency domain representation cannot be transformed back to the time domain (except for a few carefully designed filterbanks with other limitations). In the case of ACA it has not been consistently shown that approaches using a filterbank give more reliable results than, e.g., an STFT-based analysis. One possible explanation is that in many cases the subject of investigation is not what humans are able to hear in a signal but physical properties of the signal. Still the argument that ultimately every audio analysis task is about the extraction of *perceivable* features has merit as well. Auditory modeling is a lively subject of research; Lyon et al. review different approaches in [22].

2.2.5.1 Gammatone Filterbank

A widely used auditory filterbank is the so-called *gammatone filterbank*. A gammatone filter has an impulse response of the following form [23, 24]:

$$h(i) = \frac{a \cdot (i/f_S)^{\mathcal{O}-1} \cdot \cos\left(2\pi \cdot f_c \frac{i}{f_S}\right)}{e^{2\pi i \Delta f / f_S}} \tag{2.65}$$

Figure 2.11 shows the impulse response of such a gammatone filter with the following parametrization: center frequency $f_c = 1\,\text{kHz}$, bandwidth $\Delta f = 125\,\text{Hz}$, and order $\mathcal{O} = 4$.

Slaney showed that gammatone filters can be efficiently implemented with cascaded second-order filters [25]. A survey of different gammatone filter implementations can be found in [26].

Figure 2.12 shows the frequency (magnitude) response of a gammatone filterbank as computed by the *Auditory Toolbox* [27] for 20 bands between 100 Hz and 24 kHz.

2.2.6 Correlation Function

The *correlation function* can be used to compute the similarity between two signals at a lag η. Given two signals $x(i)$ and $y(i)$, the so-called *Cross Correlation Function (CCF)* is

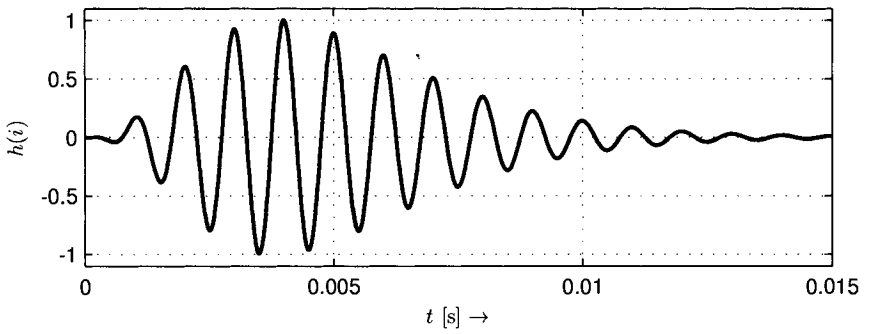

Figure 2.11 Impulse response of a gammatone filter with a center frequency $f_c = 1\,\text{kHz}$, a bandwidth $\Delta f = 125\,\text{Hz}$, and an order $\mathcal{O} = 4$

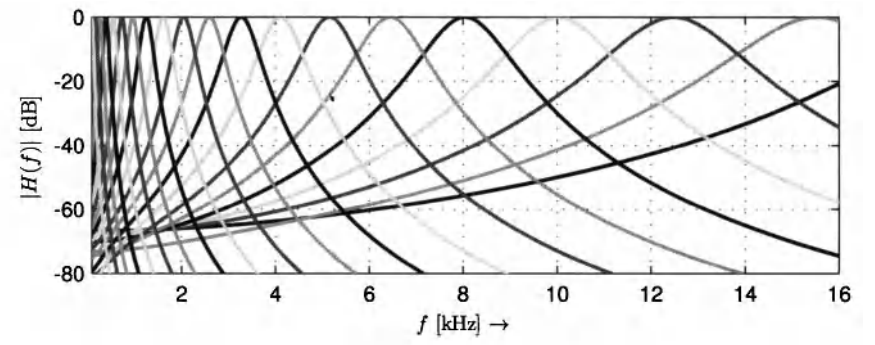

Figure 2.12 Magnitude frequency response of a gammatone filterbank with 20 bands between 100 Hz and 24 kHz

defined by

$$r_{xy}(\eta) = \sum_{i=-\infty}^{\infty} x(i) \cdot y(i+\eta). \tag{2.66}$$

Assuming the input signals are only blocks of $\mathcal{K} = i_e(n) - i_s(n) + 1$ samples, we can imagine an infinite number of zeros to the right and to the left of the actual samples; Eq. (2.66) can still be applied (although in a real application the multiplications with zero can obviously be omitted) and the CCF of two blocks of samples is thus

$$r_{xy}(\eta, n) = \sum_{i=i_s(n)}^{i_e(n)-\eta} x(i) \cdot y(i+\eta) \tag{2.67}$$

and will equal zero for all lags larger than $\mathcal{K} = i_e(n) - i_s(n) + 1$:

$$r_{xy}(\eta, n) = 0 \quad \text{if } |\eta| \geq \mathcal{K}. \tag{2.68}$$

Another possible interpretation of the block-wise CCF is the infinite correlation function of signals multiplied with a rectangular window $w_{\mathrm{R}}(i)$:

$$r_{xy}(\eta, n) = \sum_{i=-\infty}^{\infty} x(i) w_{\mathrm{R}}(i) \cdot y(i+\eta) w_{\mathrm{R}}(i+\eta). \tag{2.69}$$

2.2.6.1 Normalization

The correlation function is usually multiplied by a normalization factor λ_{c}. If only a block of samples of the signals $x(i)$ and $y(i)$ is being considered, then

$$\lambda_{\mathrm{c}} = \frac{1}{\sqrt{\left(\sum_{i=i_{\mathrm{s}}(n)}^{i_{\mathrm{e}}(n)} x^2(i)\right) \cdot \left(\sum_{i=i_{\mathrm{s}}(n)}^{i_{\mathrm{e}}(n)} y^2(i)\right)}}. \tag{2.70}$$

The number of non-zero multiplications decreases linearly with increasing lag until it is only one for $\eta = K - 1$. Therefore, the resulting correlation function is shaped triangular with the shape's maximum at lag $\eta = 0$. This triangular weighting can be avoided by taking into account the current lag η for the normalization:

$$\lambda_{\mathrm{c}}(\eta) = \frac{K}{(K - |\eta|) \cdot \sqrt{\left(\sum_{i=i_{\mathrm{s}}(n)}^{i_{\mathrm{e}}(n)} x^2(i)\right) \cdot \left(\sum_{i=i_{\mathrm{s}}(n)}^{i_{\mathrm{e}}(n)} y^2(i)\right)}}. \tag{2.71}$$

For large lags η the results will, however, become unreliable due to the limited amount of samples from which they are calculated.

A different way of avoiding the triangular shape of the correlation function is to use signal blocks of different sizes, namely a block length of $3K$ for the signal x and a block length of K samples for the signal y. Typically, the number of samples for x is increased by appending K preceding and succeeding samples, respectively. While this leads to a extended theoretical non-zero range of r_{xy} for $|\eta| \leq 2K$, the results for $|\eta| > K$ are usually discarded. The normalization procedure is in this case unclear, although three options present themselves:

- *normalization per lag* requiring the computation of the denominator of Eq. (2.70) for every individual lag,
- *overall normalization* by computing the denominator from unequal length sequences, and
- *two-stage normalization* by first finding the maximum of the unnormalized cross correlation and then computing the denominator for two short length sequences around the maximum.

If only the current block of samples from signal x is available, this block may also be copied and pasted to the start and end of this block, resulting in the same amount of $3K$ samples. If the block of samples from signal x would theoretically be pasted an infinite number of times, then the signal would be periodic with K, which in turn would result in the periodicity of the correlation function: $r_{xy}(\eta) = r_{xy}(\eta + K)$. This is then called the *Circular Correlation Function (CiCF)*; the circularity aspects have to be taken into account when the CCF is computed in the frequency domain as described in Sect. 2.2.6.4.

Autocorrelation Function Properties

- *Autocorrelation Function (ACF)* at lag 0:
  $r_{xx}(0, n) = 1$ when normalized according to Eq. (2.70), otherwise it is the *Root Mean Square (RMS)* (see Sect. 4.3.1) of the block.
- Maximum:
  $|r_{xx}(\eta, n)| \leq r_{xx}(0, n)$ since the signal can at no lag be more similar to itself than without lag ($\eta = 0$).
- Symmetry:
  $r_{xx}(\eta, n) = r_{xx}(-\eta, n)$. The ACF is symmetric around lag $\eta = 0$. This means that the calculation of negative lags can be omitted for reasons of computational efficiency.
- Periodicity:
  The ACF of a periodic signal will be periodic with the period length of the input signal.

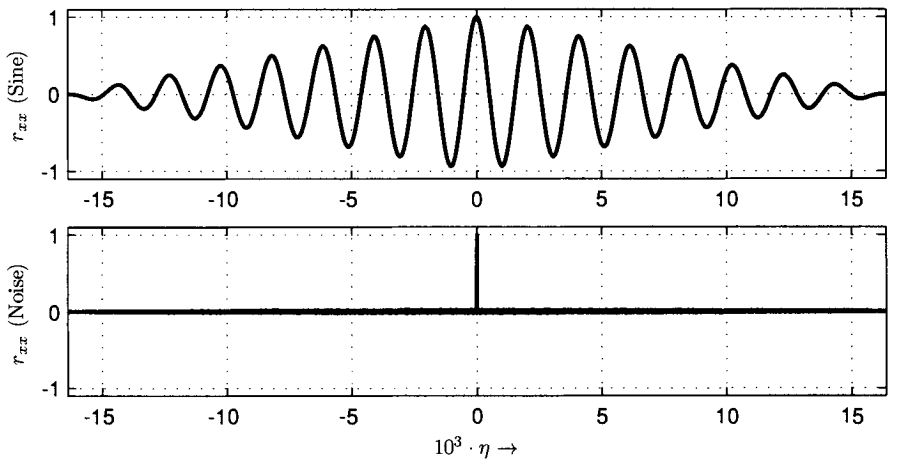

Figure 2.13 ACF of a sinusoid (top) and white noise (bottom)

2.2.6.2 Autocorrelation Function

The ACF is a special case of the correlation function with identical input signals $x(i) = y(i)$ and is therefore referred to as $r_{xx}(\eta)$. It is a measure of self-similarity, and its properties are summarized in the box above.

Figure 2.13 shows the ACF computed with a block of a sinusoid and a block of white noise.

2.2.6.3 Applications

The CCF can be used to find a specific latency or offset between a signal and the delayed signal. The lag of the maximum of the CCF indicates the delay between the signals. A

typical example would be a radar signal with its echo coming back after being reflected by an obstacle. Finding the maximum of the CCF between the sent and the reflected signal should indicate the transmission time and thus the distance.

A CCF is sometimes also computed between a real signal and a constructed signal to find out whether the original signal shows similarities to the synthetic signal. An example would be to compute the CCF of the envelope of a drum loop with a series a weighted peaks to find a pattern in the signal.

The ACF is usually used to find periodicities within the signal. A periodic input signal will lead to a period ACF with peaks at the length of the period and its integer multiples. The ACF can be thus used to find a (fundamental) frequency (see Sect. 5.3.3.2) or to find the time interval between beats in music.

2.2.6.4 Calculation in the Frequency Domain

Rewriting either the CCF or the ACF as a convolution operation reveals an important property:

$$\begin{aligned} r_{xx}(\tau) &= \int_{-\infty}^{\infty} x(\tau) \cdot x(t+\tau)\, d\tau \\ &= x(\tau) * x(-\tau). \end{aligned} \tag{2.72}$$

Using Eqs. (B.7) and (B.25) it can be deduced that

$$\begin{aligned} \mathfrak{F}\{r_{xx}(\tau)\} &= R_{xx}(\mathrm{j}\omega) \\ &= \mathfrak{F}\{x(\tau) * x(-\tau)\} \\ &= X(\mathrm{j}\omega) \cdot X^*(\mathrm{j}\omega) \\ &= |X(\mathrm{j}\omega)|^2. \end{aligned} \tag{2.73}$$

This relation is called the *Wiener-Khinchin theorem.*

The computation of the ACF in the time domain can thus be replaced by a simple multiplication in the frequency domain preceded and followed by an FT and an inverse FT, respectively. For long blocks this will save computing cycles and reduce computational workload when using an FFT algorithm. Note that with blocks of sampled data the result will be a *circular* ACF if the transformed block of data has not been correctly padded with zeros before transforming it. More specifically, computing the CCF of two blocks of length $\mathcal{K}$ requires a minimum FFT length of $2\mathcal{K}$.

Frequency Domain Compression

In certain applications such as auditory processing it might be of interest to apply a nonlinear compression function to the magnitude spectrum before transforming it back. Tolonen and Karjalainen named such a combined approach the *generalized ACF* to be computed by (see Sect. 5.3.4.2, [28])

$$r_{xx}^{\beta}(\eta, n) = \mathfrak{F}^{-1}\left\{|X(\mathrm{j}\omega)|^{\beta}\right\}. \tag{2.74}$$

A value $\beta = 2$ would result in the normal ACF.

2.2.7 Linear Prediction

The idea of *linear prediction* is to use preceding (sample) values to estimate (or *predict*) future values. In its most common form, the predicted value $\hat{x}(i)$ is a linear combination

of the preceding samples

$$\hat{x}(i) = \sum_{j=1}^{\mathcal{O}} b_j \cdot x(i-j). \tag{2.75}$$

The filter order $\mathcal{O}$ can vary between only a few coefficients and up to thousands of coefficients. To allow adaption to a time-variant signal $x(i)$, the predictor filter is usually updated for every sample block to ensure good predictive qualities. In that case the process is called *adaptive linear prediction.*

In order to estimate the coefficients b_j from the signal, the usual approach is to minimize the power of the *prediction error*

$$e_{\mathrm{P}}(i)^2 = (x(i) - \hat{x}(i))^2 \tag{2.76}$$

$$= \left(x(i) - \sum_{j=1}^{\mathcal{O}} b_j \cdot x(i-j) \right)^2 . \tag{2.77}$$

The minimum of this power can be found by minimizing $\partial e_{\mathrm{P}}^2(i)/\partial b_j$. The prediction coefficients b_j can be computed by solving the following system of equations:

$$\sum_{j=1}^{\mathcal{O}} b_j \cdot r_{xx}(|j-\eta|) = -r_{xx}(\eta), \quad 1 \leq \eta \leq \mathcal{O}. \tag{2.78}$$

An efficient numeric solution to this system of equations has been presented by Levinson and Durbin [29, 30].

A signal can only be predicted well if there are statistical relations between the individual samples. Noise is an example of a signal type that cannot be predicted at all; purely periodic signals, however, can theoretically be perfectly predicted. For music signals this means that tonal parts of the signal can be predicted well while noisy and non-periodic parts such as initial transients cannot be predicted.

Figure 2.14 plots the original signal (top), the predicted signal (middle), and the prediction error (bottom) for an adaptive predictor of order $\mathcal{O} = 20$ and a block length of 2048 samples. The source signal is an excerpt of a song of popular music. The plot illustrates two noteworthy properties of predicted signals. First, the overall power of the prediction error is significantly lower than the power of the input signal. This is used in predictive waveform coding algorithms such as *Adaptive Differential Pulse Code Modulation (ADPCM)*, which basically translates the lower power into a bit rate reduction [31]. Secondly, the amplitude of the prediction error is low during the quasi-periodic states of the input signal and spikes at transient and thus noisy segments of the signal.

A detailed introduction to linear prediction and autoregressive modeling can be found in [32].

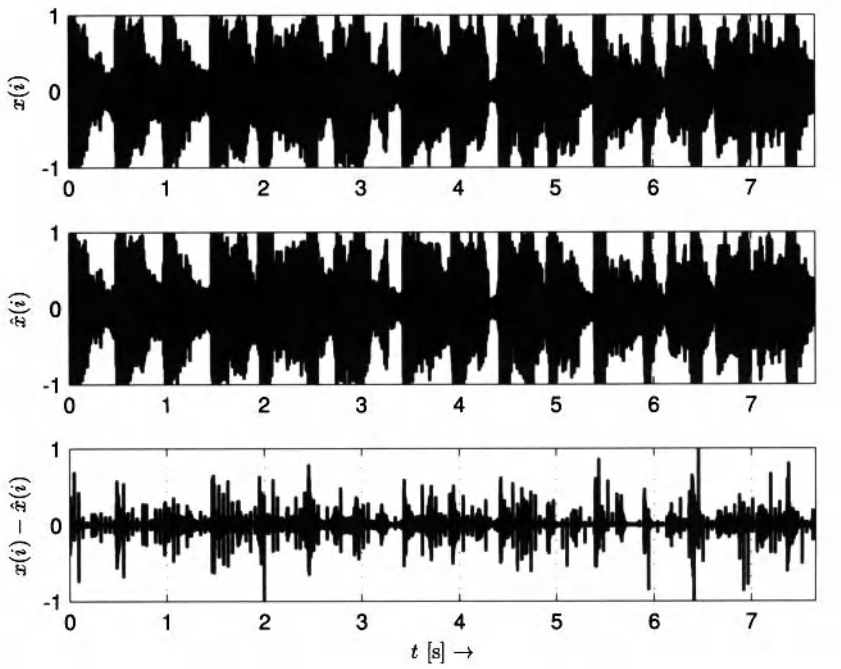

Figure 2.14 Plot of the original signal (top), the signal estimated with linear prediction (mid), and the resulting prediction error (bottom)

CHAPTER 3

INSTANTANEOUS FEATURES

Over the last few decades, a set of different widely used features has established itself for audio content analysis. Many of these features will be presented in this chapter. However, the choice of the specific features used in an algorithm will in the end always be driven by the task at hand; this can make the modification of well-known features as well as the design of new features advantageous or even mandatory. Therefore, the number of possible features used in audio content analysis is probably limitless and only a more or less representative set can be presented in the following. The various pre- and post-processing options closely related to feature extraction will be covered by this chapter as well.

The term *instantaneous feature*, *short-term feature*, or *descriptor* is generally used for measures that generate one value per (short) block of audio samples. An instantaneous feature is not necessarily musically, musicologically, or perceptually meaningful all by itself, and it is frequently referred to as a *low-level feature*. A low-level feature can serve as a building block for the construction of higher level features describing more semantically meaningful properties of the (music) signal (such as tempo, key, melodic properties, etc.).

A good example for an instantaneous feature is the magnitude or power level of the audio signal extracted on a block-per-block basis. It represents a widely known reduced representation of the audio signal. Although simple to extract, it may show characteristics usable for the extraction of higher level information. As Fig. 3.1 illustrates with three envelope excerpts of length 15 s, the signal categories speech, chamber music, and pop music show some variation that may be easily identified by an experienced user. This is, of course, not necessarily true for every possible excerpt, but it tells us that some kind of

An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics, First Edition. Alexander Lerch.

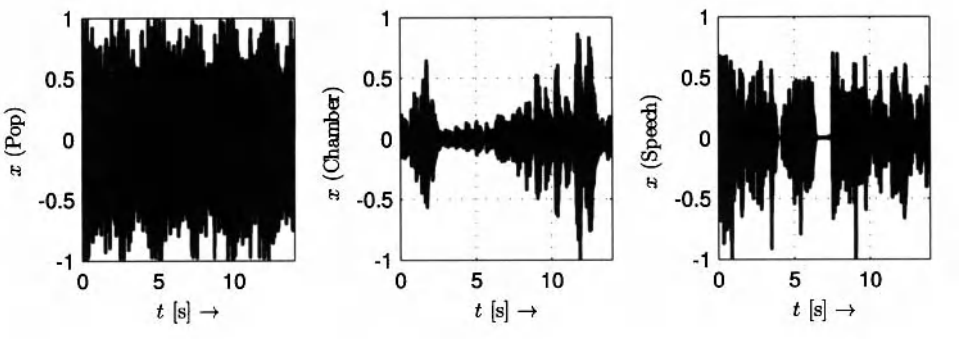

Figure 3.1 Envelope of excerpts from a typical speech recording (left), a string quartet recording (mid), and a pop recording (right) with a length of 15 s

envelope characteristic may be useful in automatic detection of a signal type or musical genre classification.

Instantaneous features can be categorized using different taxonomies. Probably the most obvious categorization is in the computational domain, which in the case of audio signals is usually either the *time domain* or the *frequency domain*. In several applications, features are calculated from other features, so the *feature domain* may be added as well. In the MPEG-7 standard, the following feature categories have been used: *basic*, *basic spectral*, *signal parameters*, *temporal timbral*, *spectral timbral*, and *spectral basis representations* [33]. Peeters categorizes features into the classes *temporal shape* (e.g., waveform-based envelope), *temporal* (e.g., zero crossing rate, ACF coefficients), *energy* (e.g. global or tonal energy), *spectral shape* (STFT-based features), *harmonic* (e.g., harmonic/noise ratio, tonalness), and *perceptual* (e.g., modeling human hearing) features [34]. Eisenberg uses the three categories *time domain*, *spectral domain*, and *harmonic* features [35].

As can be seen from these examples it is difficult to find a simple and consistent yet practically useful feature categorization. One feature may, for example, fit into more than one category because the categories may be overlapping. However, in the end this discussion is in itself only of limited use, so we will just boldly use the following categories for our instantaneous features in the following:

- *statistical properties*: features that are commonly used in statistical signal description such as standard deviation etc. (see Sect. 3.2),
- *spectral shape*: features describing the shape of the (magnitude spectrum of the) STFT (see Sect. 3.3),
- *technical/signal properties*: features that describe specific technical properties of the signal and cannot be categorized in other domains (see Sect. 3.4), and
- *intensity properties*: features closely related to the amplitude or intensity of the audio signal such as volume and loudness (see Chap. 4).

3.1 Audio Pre-Processing

The raw audio data is frequently pre-processed before computing instantaneous features from the data. The motivation for this pre-processing step is to reduce the amount of audio data to be analyzed by omitting unnecessary information or to minimize the impact of unwanted information on the extracted features. The standard approaches differ mainly in terms of whether the designed algorithm works in

- *real time*, meaning that only the current and past samples or blocks of samples are known, or
- *offline*, meaning that all upcoming samples of an audio file are known as well.

3.1.1 Down-Mixing

In most of the analysis problems the information of interest can be represented by one single audio channel as well as by multiple input channels. For example, the tempo or information on the musical style should be extractable from a mono-recording as well as from stereo or multi-channel recordings.

Down-mixing is usually done by simply computing the arithmetic mean as defined by Eq. (3.10) over all input channel signals $x_c(i)$ per sample i. The number of input channels is $\mathcal{C}$.

$$x(i) = \frac{1}{\mathcal{C}} \sum_{c=0}^{\mathcal{C}-1} x_c(i). \tag{3.1}$$

It is also possible to apply different weights to different channels; surround channels may, for example, have a lower weight than front channels. An alternative to down-mixing could be to use only one pre-selected audio channel, however, this may result in loss of information if the channels have been mixed to produce a wide spatial image. Some audio applications also apply a phase shift of 90° to one channel before down-mixing a stereo signal to mono. The reason is to avoid a level boost of components present in all audio channels (mono-components).

3.1.2 DC Removal

A DC offset — shown by a signal's arithmetic mean significantly different from zero — usually does not provide any useful information and may have unwanted impact on the feature results.

3.1.2.1 Offline

If all audio data $x_{\mathrm{DC}}(i)$ of length $\mathcal{I}$ is available, it is possible to simply compute the arithmetic mean and subtract it from every sample:

$$x(i) = x_{\mathrm{DC}}(i) - \frac{1}{\mathcal{I}} \sum_{i=0}^{\mathcal{I}-1} x_{\mathrm{DC}}(i). \tag{3.2}$$

3.1.2.2 Real Time

A self-evident method to remove the DC part from your signal is to apply a high-pass filter to it. Basically any high-pass filter can be used. The simplest method to do so in real time

is to apply a differentiator:

$$x(i) = x_{\mathrm{DC}}(i) - x_{\mathrm{DC}}(i-1). \tag{3.3}$$

A differentiator, however, has significant impact on higher frequency components as well. This effect can be lessened by low-pass filtering the difference; the resulting DC filter would be

$$x(i) = (1-\alpha) \cdot \big(x_{\mathrm{DC}}(i) - x_{\mathrm{DC}}(i-1)\big) + \alpha \cdot x(i-1). \tag{3.4}$$

Another possibility is to use a long MA filter (see Sect. 2.2.1.1) of a length $\mathcal{O}$ which provides a sufficiently reliable estimate of the arithmetic mean to be subtracted from the input signal:

$$x(i) = x_{\mathrm{DC}}(i) - \frac{1}{\mathcal{O}} \sum_{j=i-\mathcal{O}/2}^{i+\mathcal{O}/2-1} x_{\mathrm{DC}}(j). \tag{3.5}$$

3.1.3 Normalization

In order to extract features independently of the amplitude scaling of the input signal, the signal can be *normalized* to have a pre-defined (maximum) amplitude or power.

3.1.3.1 Offline

A simple and frequently used method to normalize an audio signal is to detect the overall maximum of its absolute sample values and scale the signal so that this maximum's absolute value is mapped to 1:

$$x(i) = \frac{x_s(i)}{\max\limits_{\forall i} \big(|x_s(i)|\big)}. \tag{3.6}$$

This approach results in a normalized magnitude but does not warrant equal loudness of different input files. Furthermore, the *normalization* may be influenced by signal distortions such as the clicks and crackles of a vinyl recording. In this case, the normalization to the maximum click amplitude will result in an "incorrect" scaling of the music data.

The alternative is to use some other reference than the maximum for the derivation of the scaling factor such as an RMS (see Sect. 4.3.1) or a loudness measurement with large integration time (see Sect. 4.5). In this case, additional processing may be necessary in order to avoid potential clipping of the scaled signal.

3.1.3.2 Real Time

Normalizing a signal in a real-time context is difficult. It can be done with algorithms for automatic gain control or compressors and limiters monitoring the instantaneous input signal characteristics (e.g., the RMS or the peak level) and aiming to adjust a time-variant gain value according to it.

3.1.4 Down-Sampling

Down-sampling requires the application of a *sample rate conversion* algorithm which converts the input sample rate f_{S} to a lower sample rate f_{d}. Down-sampling thus reduces both the amount of audio samples and the bandwidth of the resulting audio signal.

The easiest way to down-sample is to reduce the sample rate by an integer factor l. The output sample rate is

$$f_{\mathrm{d}} = \frac{f_{\mathrm{S}}}{l}. \tag{3.7}$$

If we just pick every lth sample, then the down-sampled signal x_{d} is

$$x_{\mathrm{d}}(i) = x(l \cdot i). \tag{3.8}$$

Taking into account the sampling theorem as given in Eq. 2.9 and the time scaling property of the spectrum as given in Eq. (2.54) it becomes clear the aliasing will occur if the input signal $x(i)$ contains frequency components higher than $f_{\mathrm{d}}/2$. In order to ensure artifact-free down-sampling a low-pass filter has to be applied to the input signal to remove any frequency components higher than $f_{\mathrm{S}}/2l$. Sample rate conversion with non-integer factors is based on the same principles. The factor can be written as the ratio of two integer factors s/l so that the signal can first be up-sampled by factor s and the down-sampled by factor l. In between the two resampling steps it is required to apply a low-pass filter which ensures both the reconstruction of the up-sampled signal and the suppression of aliasing artifacts in the down-sampled signal. It is possible to use various combinations of interpolation algorithms and low-pass filters for down-sampling a signal. One example for such a *band-limited* interpolation has been presented by Smith and Gosset and is usually referred to as *sinc* interpolation [36].

3.1.5 Other Pre-Processing Options

As with the selection of appropriate features itself, the pre-processing options will always be adjusted to the application in mind. Every pre-processing which improves the algorithm's accuracy, its stability, or minimizes its computational workload is beneficial. In addition to the presented pre-processing options it is, for example, also common to attenuate the level of any unwanted frequency region by applying a filter to the signal.

3.2 Statistical Properties

Various methods describing the properties of a (time-invariant) properties of a signal are well-established and frequently used. These measures can be applied to both, the time-domain signal block as well as the spectrum. While the definitions below use $x(i)$ as input signal, it could be substituted by $X(k, n)$, by a series of feature values $v(n)$ or by any other signal of interest.

Theoretically, the statistical properties presented below require a signal of infinite length, however, in practical applications they can be assumed to be sufficiently accurate if the block is long enough. The block length will be denoted as

$$\mathcal{K} = i_{\mathrm{e}}(n) - i_{\mathrm{s}}(n) + 1. \tag{3.9}$$

In order to simplify definitions, there will be no differentiation between the theoretically correct property or measure (for the signal with infinite length) and its estimate (from a finite length signal block).

3.2.1 Arithmetic Mean

The *arithmetic mean* is the average of the input signal (block). It is computed by

$$\mu_x(n) = \frac{1}{\mathcal{K}} \sum_{i=i_s(n)}^{i_e(n)} x(i). \tag{3.10}$$

The result of the arithmetic mean is a value between the minimum and maximum input signal value. The unit corresponds to the unit of the input signal. For symmetric PDFs, the arithmetic mean will be the (abscissa) position of the symmetric axis. For example, a time domain audio signal usually has a mean value of approximately 0; if the signal has a DC offset, the mean will indicate the amount of the DC offset. When the PDF is not symmetric, then the calculation of the arithmetic mean is of limited use. In this case, the computation of the median (see Sect. 3.2.10) or the centroid (see Sect. 3.2.5) of the PDF might be more meaningful measures of the average.

3.2.2 Geometric Mean

The *geometric mean* is an average measure for sets of positive numbers that are ordered on a logarithmic scale. It can be computed with

$$M_x(0,n) = \sqrt[\mathcal{K}]{\prod_{i=i_s(n)}^{i_e(n)} x(i)} \tag{3.11}$$

$$= \exp\left(\frac{1}{\mathcal{K}} \sum_{i=i_s(n)}^{i_e(n)} \log\left[x(i)\right]\right). \tag{3.12}$$

Equation (3.12) is equivalent to Eq. (3.11) but avoids problems with computational accuracy for long blocks of data and large values at the computational cost of applying a logarithm to each signal value. The result of the geometric mean is a value between the minimum and maximum input signal value. The unit corresponds to the unit of the input signal.

3.2.3 Harmonic Mean

The *harmonic mean* is an average measure appropriate for averaging rates. It is

$$M_x(-1,n) = \frac{\mathcal{K}}{\sum_{i=i_s(n)}^{i_e(n)} 1/x(i)}. \tag{3.13}$$

3.2.4 Generalized Mean

A generalized expression for the calculation of different measures of mean is

$$M_x(\beta,n) = \sqrt[\beta]{\frac{1}{\mathcal{K}} \sum_{i=i_s(n)}^{i_e(n)} x^\beta(i)}. \tag{3.14}$$

Different values of β then lead to different measures:

- $\beta = 1$: arithmetic mean, (Sect. 3.2.1)
- $\beta = 2$: quadratic mean, or RMS (see Sect. 4.3.1)
- $\beta = -1$: harmonic mean (Sect. 3.2.3)
- $\beta \to 0$: geometric mean, (Sect. 3.2.2)
- $\beta \to -\infty$: minimum
- $\beta \to \infty$: maximum

3.2.5 Centroid

The *centroid* computes the *Center of Gravity (COG)* of a block of input values. It is computed by the index-weighted sum of the values divided by their unweighted sum:

$$v_{\mathrm{C}}(n) = \frac{\sum_{i=i_{\mathrm{s}}(n)}^{i_{\mathrm{e}}(n)} \big(i - i_{\mathrm{s}}(n)\big) \cdot x(i)}{\sum_{i=i_{\mathrm{s}}(n)}^{i_{\mathrm{e}}(n)} x(i)}. \tag{3.15}$$

The result will be a value in the range of $0 \leq v_{\mathrm{C}}(n) \leq \mathcal{K} - 1$. For more information, see Sect. 3.3.3 on the spectral centroid.

3.2.6 Variance and Standard Deviation

Both the *variance* and the *standard deviation* measure the spread of the input signal $x(i)$ around its arithmetic mean. The variance σ_x^2 is defined by

$$\sigma_x^2(n) = \frac{1}{\mathcal{K}} \sum_{i=i_{\mathrm{s}}(n)}^{i_{\mathrm{e}}(n)} \big(x(i) - \mu_x(n)\big)^2. \tag{3.16}$$

Strictly speaking this is the so-called *biased* estimate of the variance in contrast to the *unbiased* estimate

$$\sigma_{x,\mathrm{b}}^2(n) = \frac{1}{\mathcal{K} - 1} \sum_{i=i_{\mathrm{s}}(n)}^{i_{\mathrm{e}}(n)} \big(x(i) - \mu_x(n)\big)^2. \tag{3.17}$$

In case of $\mu_x(n) = 0$, the variance equals the power of the observed block of samples.

The standard deviation σ_x can be computed directly from the variance

$$\sigma_x(n) = \sqrt{\sigma_x^2(n)}. \tag{3.18}$$

In case of $\mu_x(n) = 0$, the standard deviation equals the RMS of the observed block of samples (see Sect. 4.3.1). The result of the standard deviation is in the range

$$0 \leq \sigma_x(n) \leq \max_{i \in [i_{\mathrm{s}}(n); i_{\mathrm{e}}(n)]} |x(i)|.$$

Table 3.1 Spectral skewness for the three prototypical spectral shapes *silence* (zero magnitude at all bins), *flat* (same amplitude at all bins), and *peak* (all bins except one have zero magnitude)

Spectral Shape	v_{SSk}
silence	not def.
flat mag.	not def.
single peak (@ k_s)	not def.

The standard deviation is 0 for silent or constant input signals. Its unit corresponds to the input signal's unit.

Although the computation of standard deviation or variance of audio samples might be of interest in specific cases, it is more common in ACA to compute them from a series of features.

3.2.7 Skewness

The *skewness* is referred to as the third central moment of a variable divided by the cube of its standard deviation. It is defined by

$$v_{\mathrm{Sk}}(n) = \frac{1}{\sigma_x^3(n) \cdot \mathcal{K}} \sum_{i=i_s(n)}^{i_e(n)} \big(x(i) - \mu_x(n)\big)^3. \tag{3.19}$$

The skewness is a measure of the asymmetry of the PDF. It will be 0 for symmetric distributions, negative for distributions with their mass centered on the right (left-skewed), and positive for distributions with their mass centered on the left (right-skewed). Note that while every symmetric distribution has zero skewness the converse is not necessarily true. The range is unrestricted. It is not defined for signals with a standard deviation of 0.

3.2.7.1 Spectral Skewness

The *spectral skewness* measures the symmetry of the distribution of the spectral magnitude values around their arithmetic mean. It is defined by

$$v_{\mathrm{SSk}}(n) = \frac{2 \sum_{k=0}^{\mathcal{K}/2-1} \big(|X(k,n)| - \mu_{|X|}\big)^3}{\mathcal{K} \cdot \sigma_{|X|}^3}. \tag{3.20}$$

For all features computed from the spectrum we will present the feature output for three prototypical spectral shapes in a table; these shapes are

- *silence*: $|X(k,n)| = 0$
- *white noise* $|X(k,n)| = const$ and
- a *single spectral peak*

$$|X(k,n)| = 0|_{k \neq k_s} \vee |X(k,n)| = A|_{k=k_s}. \tag{3.21}$$

The spectral skewness is, unfortunately, not a good first example as it is not defined for either of these three signals (see Table 3.1).

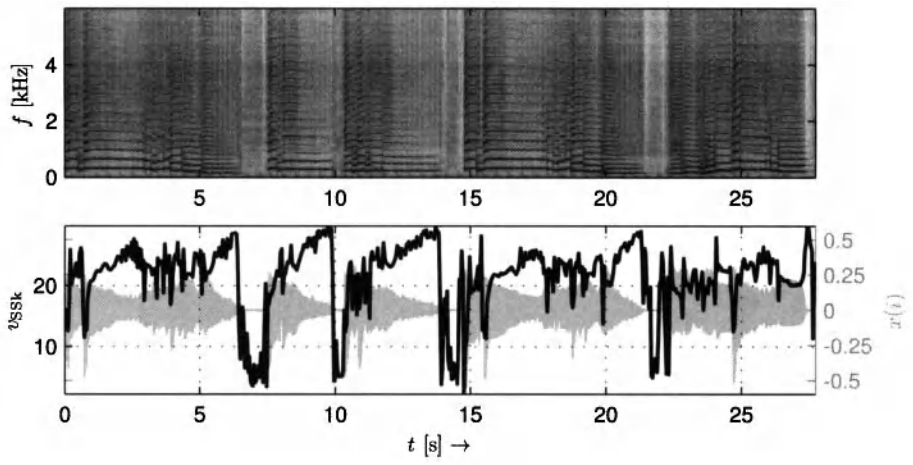

Figure 3.2 Spectrogram (top), waveform (bottom background), and spectral skewness (bottom foreground) of a saxophone signal

Figure 3.2 shows the spectral skewness for an example signal. During signal pauses the spectral skewness drops since the spectral magnitudes are similar, while at positions with high magnitudes at the fundamental frequency the magnitude spectrum is significantly skewed. The spectral skewness increases, for example, in the region between 12 s and 15 s due to the strong decrease of the higher harmonics compared to the lower harmonics.

3.2.8 Kurtosis

The *kurtosis* is referred to as the fourth central moment of a variable divided by the fourth power of the standard deviation:

$$v_K(n) = \frac{1}{\sigma_x^4(n) \cdot \mathcal{I}} \sum_{i=i_s(n)}^{i_e(n)} \Big(x(i) - \mu_x(n)\Big)^4 - 3. \tag{3.22}$$

The kurtosis is a measure of "non-Gaussianity" of the PDF; more specifically, it indicates the flatness (and peakiness, respectively) of the input values' distribution compared to the Gaussian distribution. It equals 0 for a Gaussian distribution (*mesokurtic*), is negative for a flatter distribution with a wider peak (*platykurtic*), and positive for distributions with a more acute peak (*leptokurtic*).

The range is not restricted, and similar to the skewness, the kurtosis is not defined for signals with a standard deviation of 0.

3.2.8.1 Spectral Kurtosis

The *spectral kurtosis* measures whether the distribution of the spectral magnitude values is shaped like a Gaussian distribution or not. It is defined by

$$v_{SK}(n) = \frac{2 \sum_{k=0}^{\kappa/2-1} \big(|X(k,n)| - \mu_{|X|}\big)^4}{\mathcal{K} \cdot \sigma_{|X|}^4} - 3. \tag{3.23}$$

Table 3.2 Spectral kurtosis for the three prototypical spectral shapes *silence* (zero magnitude at all bins), *flat* (same amplitude at all bins), and *peak* (all bins except one have zero magnitude)

Spectral Shape	v_{SK}
silence	not def.
flat mag.	not def.
single peak (@ k_s)	not def.

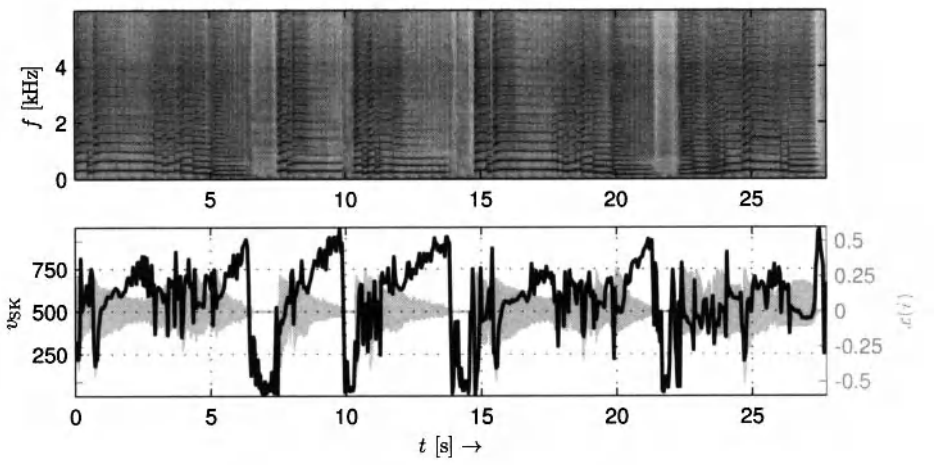

Figure 3.3 Spectrogram (top), waveform (bottom background), and spectral kurtosis (bottom foreground) of a saxophone signal

Table 3.2 shows that the spectral kurtosis is, similar to the spectral skewness, not defined for the three prototype signals.

Figure 3.3 shows the spectral kurtosis for a saxophone signal. While during the notes high values can be observed, indicating a very peaked distribution, the spectral kurtosis drops significantly during pauses.

3.2.9 Generalized Central Moments

The measures variance, skewness, and kurtosis are directly derived from the so-called central moments of different order. A central moment of order $\mathcal{O}$ is defined by

$$\gamma_{x,\beta}(n) = \sum_{i=i_s(n)}^{i_e(n)} \Big(x(i) - \mu_x(n)\Big)^{\mathcal{O}}. \tag{3.24}$$

3.2.10 Quantiles and Quantile Ranges

Quantiles can be computed from the PDF and can be used to divide the PDF into (equal sized) subsets. They are helpful in the description of asymmetric distributions or distributions with so-called outliers, occasional untypical values far from the median (see below).

If the PDF is divided into two quantiles, it is split into two parts each containing 50% of the overall number of observations. The position of the border between those two quantiles will be referred to as the *median* $Q_x(0.5)$:

$$Q_x(0.5) = x \Big|_{\int_{-\infty}^{x} p_x(y)\,dy = 0.5} \tag{3.25}$$

which equals the arithmetic mean in the case of symmetric distributions. Similarly, if the PDF is partitioned in four quantiles (so-called *quartiless*), the quantile borders would be $Q_x(0.25)$, $Q_x(0.5)$, and $Q_x(0.75)$.

In many cases, *quantile ranges* are of specific interest for the simplified description of a distribution's shape. The range spanned by 90% of the samples can be computed by $\Delta Q_x(0.9) = Q_x(0.95) - Q_x(0.05)$. This should be a good measure of the signal's range while discarding infrequent outliers, namely the upper and lower 5%.

The overall range of a signal is

$$\Delta Q_x(1.0) = \min\big(Q_x(1.0)\big) - \max\big(Q_x(0)\big). \tag{3.26}$$

3.3 Spectral Shape

Most of the features describing the spectral shape of an audio signal are closely related to the timbre of this signal. The *timbre* of a sound is referred to as its *sound color*, its *quality*, or its *texture*. Besides pitch and loudness, timbre is considered as "the third attribute of the subjective experience of musical tones" [37]. Timbre can be explained by two closely related phenomena, which will be referred to as *timbre quality* and *timbre identity*. The timbre quality allows humans to group together different sounds originating from the same source such as two recordings made with the same instrument. Timbre identity enables the differentiation of two sounds with the same tone characteristics (loudness, pitch if available) played on two instruments. Thus, the quality represents general timbre properties of a sound ("sounds like a violin"), while the timbre identity refers to instrument specifics ("one violin sounds different from the other").

Loudness and pitch are unidimensional properties, as sounds with different loudness or pitch can be ordered on a single scale from quiet to loud and low to high, respectively. Timbre is a multi-dimensional property [38, 39]; this complicates its definition. A good summary over the various attempts of the definition of the term timbre has been compiled by Sandell.[1] The most prominent example is probably the definition of the American Standards Association from 1960 that defined timbre as "that attribute of auditory sensation in terms of which a listener can judge that two sounds similarly presented and having the same loudness and pitch are dissimilar" [40]. This definition has been criticized repeatedly by researchers mainly because according to Bregman [41] it

- does not attempt to explain what timbre is, but only what timbre is *not*, i.e., loudness and pitch, and
- implies that timbre only exists for sounds with a pitch, implicating that, for example, percussive instruments do not have a timbre.

[1] Sandell, Greg: *Definitions of the word "Timbre"*. http://www.zainea.com/timbre.htm. Last retrieved on Nov. 16, 2011.

Early uses of the term *timbre* can be found in Blumenbach [42]. Helmholtz was probably the first to detect the dependency between the timbre of a sound and the relative amplitudes of the harmonics during the second half of the 19th century [43]. Although he noted that other influences play a role in defining the quality of a tone such as the "beginning" and "ending" of a sound, he restricted his definition of timbre ("Klangfarbe") to the harmonic amplitude distribution only.

Stumpf extended the definition of timbre by two more attributes [44]. He named the relative amplitude of harmonics, the form and length of the attack time and note endings, and additional sounds and noise as the third timbre-determining component.

Seashore restricted the term timbre during the first half of the 20th century to the harmonic structure that "is expressed in terms of the number, distribution, and relative intensity of its partials," but he additionally introduced the term *sonance*, referring to "the successive changes and fusions which take place within a tone from moment to moment" [45]. This distinction did, however, not find broad acceptance in the research community.

Nowadays, timbre is understood as the phenomenon that takes into account both spectral patterns and temporal patterns [39, 46]. Timbre perception is obviously influenced by numerous parameters of both the onset properties such as rise time, inharmonicities during the onset, etc. and numerous steady-state effects such as vibrato, tremolo, pitch instability, etc. [37]. In the following, we will restrict ourselves to measures of spectral shape since most of the features describing "temporal timbre" only work for individual monophonic notes as opposed to complex time-variant mixtures of signals. The presented features represent spectral shape and are technically motivated; therefore, there is not necessarily a direct relation to the human perception of timbre.

3.3.1 Spectral Rolloff

The *spectral rolloff* is a measure of the bandwidth of the analyzed block n of audio samples. The spectral rolloff $v_{\mathrm{SR}}(n)$ is defined as the frequency bin below which the accumulated magnitudes of the STFT $X(k,n)$ reach a certain percentage κ of the overall sum of magnitudes:

$$v_{\mathrm{SR}}(n) = i \left| \sum_{k=0}^{i} |X(k,n)| = \kappa \cdot \sum_{k=0}^{K/2-1} |X(k,n)| \right. \tag{3.27}$$

with common values for κ being 0.85 (85%) or 0.95 (95%).

The result of the spectral rolloff is a bin index in the range $0 \leq v_{\mathrm{SR}}(n) \leq K/2 - 1$. It can be converted either to Hz with Eq. (2.44) or to a parameter range between zero and one by dividing it by the STFT size $K/2 - 1$. Low results indicate insignificant magnitude components at high frequencies and thus a low audio bandwidth.

Table 3.3 shows the results for the spectral rolloff for three prototype spectral shapes. The behavior of the spectral rolloff at pauses in the input signal may require special consideration. While the result will equal zero for absolute silence, it may be quite large for noise, including pauses with low-level noise.

Table 3.3 Spectral rolloff for the three prototypical spectral shapes *silence* (zero magnitude at all bins), *flat* (same amplitude at all bins), and *peak* (all bins except one have zero magnitude)

Spectral Shape	v_{SR}
silence	0
flat mag.	$\kappa \cdot \kappa/2 - 1$
single peak (@ k_s)	k_s

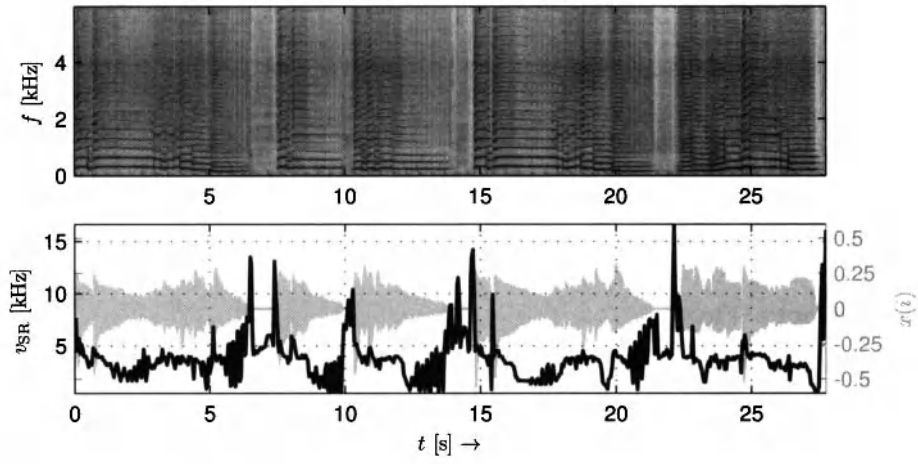

Figure 3.4 Spectrogram (top), waveform (bottom background), and spectral rolloff (bottom foreground) of a saxophone signal

Figure 3.4 shows the spectral rolloff for an example signal. It is comparably low in the presence of a tone and higher — although somewhat erratic — during the noise-filled pauses.

3.3.1.1 Common Variants

Spectral bins representing very low or very high frequencies may in many cases be considered to be unnecessary or unwanted for the analysis. Therefore, both sums in Eq. (3.27) may start and stop at pre-defined frequency boundaries f_{min}, f_{max}:

$$v_{\mathrm{SR},\Delta\mathrm{f}}(n) = i \left| \sum_{k=k(f_{\mathrm{min}})}^{i} |X(k,n)| = \kappa \cdot \sum_{k=k(f_{\mathrm{min}})}^{k(f_{\mathrm{max}})} |X(k,n)| \right| . \tag{3.28}$$

It is also common to use the power spectrum instead of the magnitude spectrum:

$$v_{\mathrm{SR},\mathrm{Pow}}(n) = i \left| \sum_{k=0}^{i} |X(k,n)|^2 = \kappa \cdot \sum_{k=0}^{\kappa/2-1} |X(k,n)|^2 \right| . \tag{3.29}$$

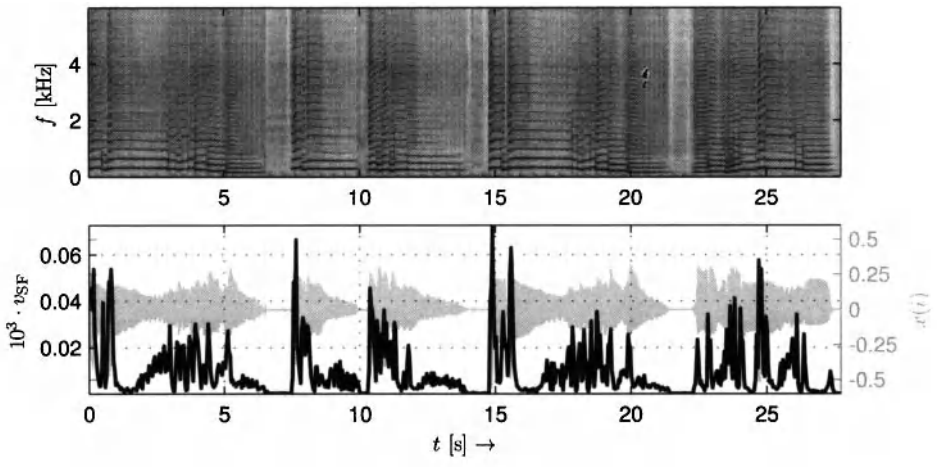

Figure 3.5 Spectrogram (top), waveform (bottom background), and spectral flux (bottom foreground) of a saxophone signal

3.3.2 Spectral Flux

The *spectral flux* measures the amount of change of the spectral shape. It is defined as the average difference between consecutive STFT frames:

$$v_{\mathrm{SF}}(n) = \frac{\sqrt{\sum_{k=0}^{K/2-1} \left(|X(k,n)| - |X(k,n-1)|\right)^2}}{K/2}. \tag{3.30}$$

The spectral flux can be interpreted as a rudimentary approximation to the sensation *roughness* which according to Zwicker and Fastl can be modeled as quasi-periodic change or a modulation in the excitation pattern levels [47].

The result of the spectral flux is a value within the range $0 \leq v_{\mathrm{SF}}(n) \leq A$ with A representing the maximum possible spectral magnitude. Thus, its output range depends on the normalization of the audio signal and the frequency transform. Low results indicate steady-state input signals or low input levels.

Figure 3.5 shows the spectral flux for an example signal. It is low during the stationary parts of the signal, such as during a note or a pause, and spikes at pitch changes and at the beginning of a new note.

3.3.2.1 Common Variants

The definition of the spectral flux above is the Euclidean distance of the two spectra as given in Eq. (5.59). The distance norm can also be generalized:

$$v_{\mathrm{SF}}(n,\beta) = \frac{\sqrt[\beta]{\sum_{k=0}^{K/2-1} \left(|X(k,n)| - |X(k,n-1)|\right)^\beta}}{K/2}. \tag{3.31}$$

Typical values for β range between $[0.25; 3]$, with $\beta = 1$ [manhattan distance, Eq. (5.60)] and $\beta = 2$ [Euclidean distance, Eq. (5.59)] being the most common.

In some applications such as note onset detection, only an increase in spectral energy is of interest; in these cases, the difference magnitude spectrum is computed[2]

$$\Delta X(k,n) = |X(k,n)| - |X(k,n-1)| \tag{3.32}$$

and all negative differences $\Delta X(k,n) < 0$ can be set to zero before summation while positive differences will be left unaltered. This is called *Half-Wave Rectification (HWR)*. Mathematically the HWR of a signal x is

$$\mathrm{HWR}(x) = \frac{x + |x|}{2}. \tag{3.33}$$

Alternative approaches can be used to derive a measure of spectral change. An example is the computation of the standard deviation of the difference magnitude spectrum $\Delta X(k,n)$:

$$v_{\mathrm{SF},\sigma}(n) = \sqrt{\frac{2}{K}\sum_{k=0}^{K/2-1}(\Delta X(k,n) - \mu_{\Delta X})^2}. \tag{3.34}$$

Another variant is to compute the logarithmic difference. This has the advantage of making the resulting feature independent of magnitude scaling but the disadvantage of zero-mean frames having to be handled individually:

$$v_{\mathrm{SF},\log}(n) = \frac{2}{K}\sum_{k=0}^{K/2-1}\log_2\left(\frac{|X(k,n)|}{|X(k,n-1)|}\right). \tag{3.35}$$

3.3.3 Spectral Centroid

The *spectral centroid* represents the COG of spectral energy (compare Sect. 3.2.5 for the time domain centroid). It is defined as the frequency-weighted sum of the power spectrum normalized by its unweighted sum:

$$v_{\mathrm{SC}}(n) = \frac{\sum_{k=0}^{K/2-1} k \cdot |X(k,n)|^2}{\sum_{k=0}^{K/2-1} |X(k,n)|^2}. \tag{3.36}$$

In the literature, numerous indications can be found that this position of energy concentration is well correlated with the timbre dimension *brightness* or *sharpness* [48–54].

The result of the spectral centroid is a bin index within the range $0 \leq v_{\mathrm{SC}}(n) \leq K/2-1$. It can be converted either to Hz by using Eq. (2.44) or to a parameter range between zero and one by dividing it by the STFT size $K/2 - 1$. Low results indicate significant low-frequency components and insignificant high frequency components and low "brightness."

Table 3.4 shows the results for the spectral centroid for three prototype spectral shapes. The behavior of the spectral centroid at pauses in the input signal requires special consideration as it is not defined for silence and will be comparably large for (low-level) noise.

Figure 3.6 shows the spectral centroid for an example signal. In the case of this monophonic signal one can see how the spectral centroid moves with the fundamental frequency during tonal parts, spikes at initial transients, and is high during pauses.

[2] Another distance measure can be used as well.

Table 3.4 Spectral centroid for the three prototypical spectral shapes *silence* (zero magnitude at all bins), *flat* (same amplitude at all bins), and *peak* (all bins except one have zero magnitude)

Spectral Shape	v_{SC}
silence	not def.
flat mag.	$\frac{K/2-1}{2}$
single peak (@ k_s)	k_s

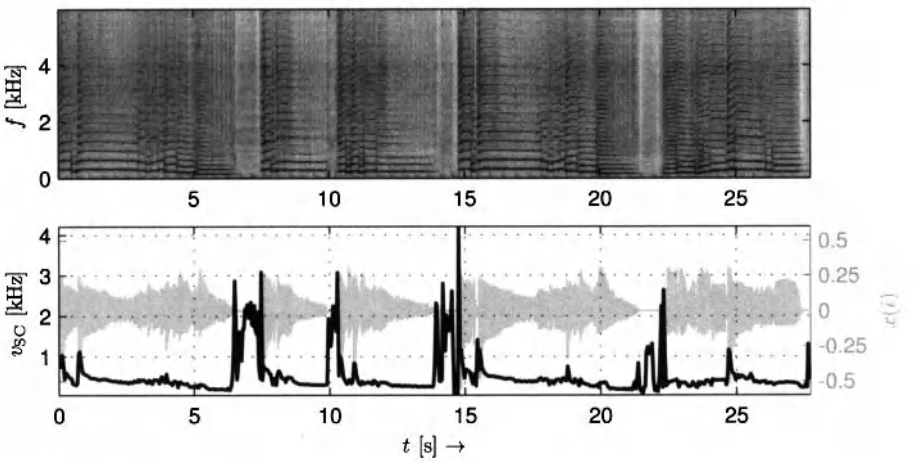

Figure 3.6 Spectrogram (top), waveform (bottom background), and spectral centroid (bottom foreground) of a saxophone signal

3.3.3.1 Common Variants

The magnitude spectrum may be used instead of the power spectrum:

$$v_{\mathrm{SC,m}}(n) = \frac{\sum_{k=0}^{K/2-1} k \cdot |X(k,n)|}{\sum_{k=0}^{N/2-1} |X(k,n)|}. \tag{3.37}$$

Zwicker and Fastl presented a psycho-acoustic model of sharpness that uses the excitation patterns to compute the sharpness [47]. It differs from Eq. (3.36) mainly in two points. First, it is computed on a non-linear bark scale, namely the so-called critical band rate which models the non-linearity of human frequency perception (compare Sect. 5.1.1.2). Second, it utilizes a psycho-acoustic loudness measure instead of the spectral power; this loudness model takes into account masking and other perceptual effects. While ignoring the difference between loudness and power, the idea of a "more human" non-linear frequency scale has been adapted for the definition of the spectral centroid in the MPEG-7 standard [33]. The critical band rate is approximated by applying a logarithm to the

Table 3.5 Spectral spread for the three prototypical spectral shapes *silence* (zero magnitude at all bins), *flat* (same amplitude at all bins), and *peak* (all bins except one have zero magnitude)

Spectral Shape	v_{SS}
silence	not def.
flat mag.	$4/\kappa \cdot \sum_{k=0}^{\kappa/4-1} (\kappa/4 - k)^2$
single peak (@ k_s)	0

frequencies with a reference point of $f_{ref} = 1000\,\text{Hz}$:

$$v_{SC,log}(n) = \frac{\sum_{k=k(f_{min})}^{\kappa/2-1} \log_2\left(\frac{f(k)}{f_{ref}}\right) \cdot |X(k,n)|^2}{\sum_{k=k(f_{min})}^{N/2-1} |X(k,n)|^2}. \tag{3.38}$$

In this specific MPEG-definition, all bins corresponding to frequencies below 62.5 Hz are combined to one band with a mid-frequency of 31.25 Hz.

3.3.4 Spectral Spread

The *spectral spread*, sometimes also referred to as *instantaneous bandwidth*, describes the concentration of the power spectrum around the spectral centroid and is a rather technical description of spectral shape. It can be interpreted as the standard deviation of the power spectrum around the spectral centroid. Its definition is

$$v_{SS}(n) = \sqrt{\frac{\sum_{k=0}^{\kappa/2-1} (k - v_{SC}(n))^2 \cdot |X(k,n)|^2}{\sum_{k=0}^{\kappa/2-1} |X(k,n)|^2}}. \tag{3.39}$$

There are indications that the spectral spread is of some relevance in describing the perceptual dimensions of timbre [52].

The result of the spectral spread is a bin range of $0 \leq v_{SS}(n) \leq \kappa/4$. It can be converted either to Hz by using Eq. (2.44) or to a parameter range between zero and one by dividing it by $\kappa/4$. Low results indicate the concentration of the spectral energy at a specific frequency region. As the spectral centroid, the spectral spread is not defined for audio blocks with no spectral energy (silence) and will result in high values if the input signal contains (low-level) white noise.

Table 3.5 shows the results for the spectral spread for three prototype spectral shapes. Figure 3.7 shows the spectral spread for an example signal. Most prominent are the high feature values during pauses and at transients; the spectral spread is low during notes for this monophonic signal. When the higher harmonics slowly disappear between 12 and 15 s, the spread of the signal decreases accordingly.

3.3.4.1 Common Variants

The definition of the spectral spread has to conform with the definition of the spectral centroid. If the spectral centroid has been calculated from the magnitude spectrum instead

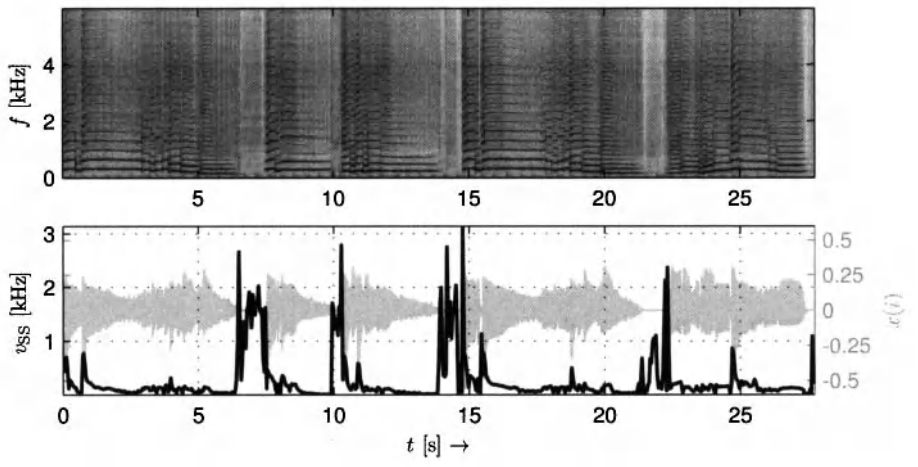

Figure 3.7 Spectrogram (top), waveform (bottom background), and spectral spread (bottom foreground) of a saxophone signal

of the power spectrum, then the spectral spread should use the magnitude spectrum as well. In the case of the MPEG-7 definition, a logarithmic frequency scale has to be used for the calculation of the spectral spread as well:

$$v_{\mathrm{SS,log}}(n) = \sqrt{\frac{\sum_{k=k(f_{\min})}^{\kappa/2-1} \left(\log_2\left(\frac{f(k)}{1000\,\mathrm{Hz}}\right) - v_{\mathrm{SC}}(n)\right)^2 \cdot |X(k,n)|^2}{\sum_{k=k(f_{\min})}^{\kappa/2-1} |X(k,n)|^2}}. \tag{3.40}$$

3.3.5 Spectral Decrease

The *spectral decrease* estimates the steepness of the decrease of the spectral envelope over frequency. It is defined by [34]

$$v_{\mathrm{SD}}(n) = \frac{\sum_{k=1}^{\kappa/2-1} \frac{1}{k} \cdot \left(|X(k,n)| - |X(0,n)|\right)}{\sum_{k=1}^{\kappa/2-1} |X(k,n)|}. \tag{3.41}$$

The result of the spectral decrease is a value $v_{\mathrm{SD}}(n) \leq 1$. Low results indicate the concentration of the spectral energy at bin 0. The spectral decrease is not defined for audio blocks with no spectral energy (silence).

Table 3.6 Spectral decrease for the three prototypical spectral shapes *silence* (zero magnitude at all bins), *flat* (same amplitude at all bins), and *peak* (all bins except one have zero magnitude)

Spectral Shape	v_{SD}
silence	not def.
flat mag.	0
single peak (@ k_s)	$1/k_s$

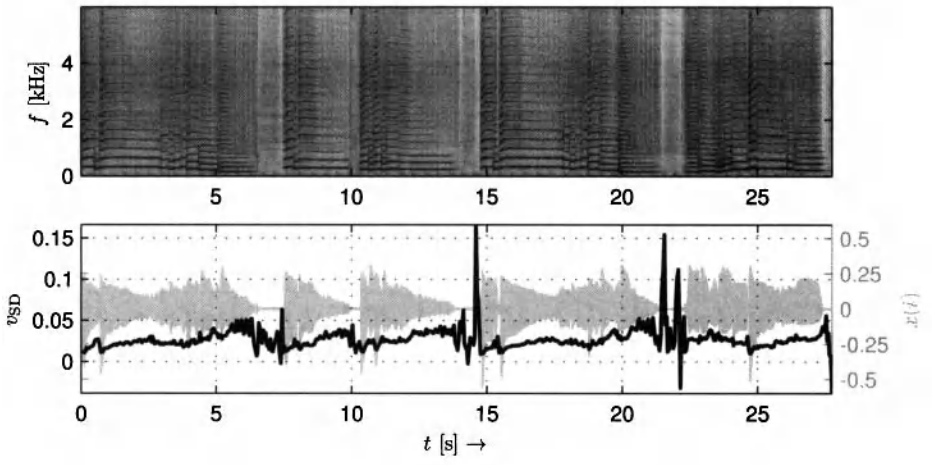

Figure 3.8 Spectrogram (top), waveform (bottom background), and spectral decrease (bottom foreground) of a saxophone signal

Table 3.6 shows the results for the spectral decrease for three prototype spectral shapes.

Figure 3.8 shows the spectral decrease for an example signal. It is difficult to draw any conclusions from the graph except that the feature behaves erratically during the pauses.

3.3.5.1 Common Variants

Reducing the spectral analysis range might lead to more meaningful results in some cases. This can be done by using a lower and upper bound k_{l} and k_{u}, respectively:

$$v_{\mathrm{SD}}(n) = \frac{\sum_{k=k_{\mathrm{l}}}^{k_{\mathrm{u}}} \frac{1}{k} \cdot \left(|X(k,n)| - |X(k_{\mathrm{l}}-1,n)|\right)}{\sum_{k=k_{\mathrm{l}}}^{k_{\mathrm{u}}} |X(k,n)|}. \tag{3.42}$$

3.3.6 Spectral Slope

The *spectral slope* is — similar to the spectral decrease — a measure of the slope of the spectral shape. It is calculated using a linear approximation of the magnitude spectrum; more specifically, a linear regression approach is used. In the presented form, the linear function is modeled from the magnitude spectrum. Its slope is then estimated with the equation

Table 3.7 Spectral slope for the three prototypical spectral shapes *silence* (zero magnitude at all bins), *flat* (same amplitude at all bins), and *peak* (all bins except one have zero magnitude, the magnitude at the bin k_s equals A)

Spectral Shape	v_{SSl}
silence	0
flat mag.	0
single peak (@ k_s)	$\frac{(k_s - \mathcal{K}/4) \cdot (A - 2A/\mathcal{K})}{\sum_{k=0}^{\mathcal{K}/2-1}(k - \mu_k)^2}$

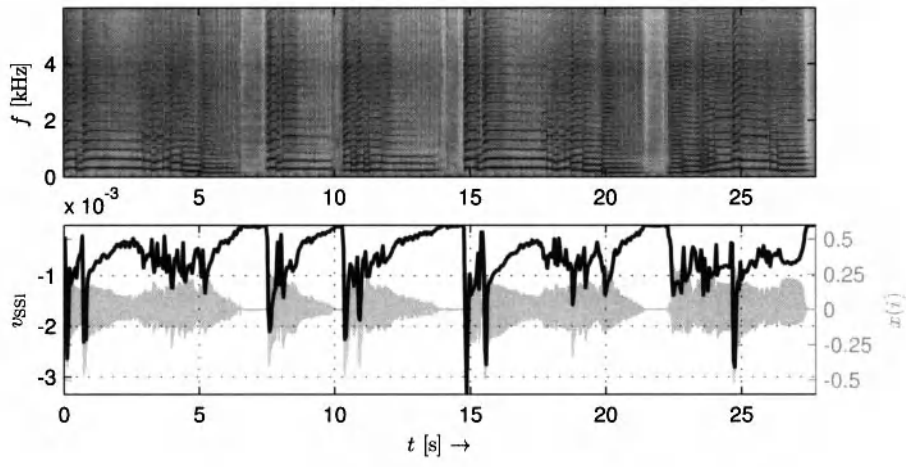

Figure 3.9 Spectrogram (top), waveform (bottom background), and spectral slope (bottom foreground) of a saxophone signal

$$v_{\mathrm{SSl}}(n) = \frac{\sum_{k=0}^{\mathcal{K}/2-1}(k - \mu_k)(|X(k,n)| - \mu_{|X|})}{\sum_{k=0}^{\mathcal{K}/2-1}(k - \mu_k)^2} \tag{3.43}$$

$$= \frac{\mathcal{K}\sum_{k=0}^{\mathcal{K}/2-1} k \cdot |X(k,n)| - \sum_{k=0}^{\mathcal{K}/2-1} k \cdot \sum_{k=0}^{\mathcal{K}/2-1} |X(k,n)|}{\mathcal{K} \cdot \sum_{k=0}^{\mathcal{K}/2-1} k^2 - \left(\sum_{k=0}^{\mathcal{K}/2-1} k\right)^2}. \tag{3.44}$$

The result of the spectral slope depends on the amplitude range of the spectral magnitudes.

Table 3.7 shows the results for the spectral slope for three prototype spectral shapes.

Figure 3.9 shows the spectral slope for an example signal. It is maximal for the noisy pauses and increases with disappearing higher harmonics.

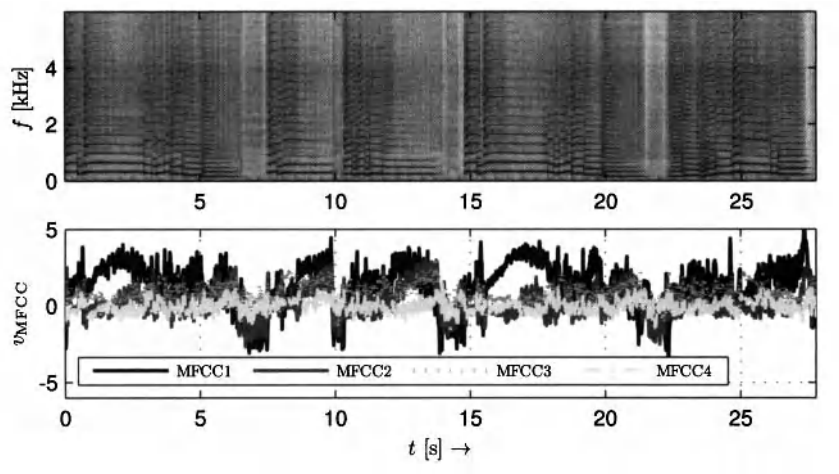

Figure 3.10 Spectrogram (top) and mel frequency cepstral coefficients 1–4 (bottom) of a saxophone signal

3.3.7 Mel Frequency Cepstral Coefficients

The *Mel Frequency Cepstral Coefficients (MFCCs)* can be seen as a compact description of the shape of the spectral envelope of an audio signal. The jth coefficient $v^j_{\mathrm{MFCC}}(n)$ can be calculated with

$$v^j_{\mathrm{MFCC}}(n) = \sum_{k'=1}^{\mathcal{K}'} \log\left(|X'(k',n)|\right) \cdot \cos\left(j \cdot \left(k' - \frac{1}{2}\right)\frac{\pi}{\mathcal{K}'}\right) \tag{3.45}$$

with $|X'(k',n)|$ being the mel-warped magnitude spectrum at the signal block. The calculation is based on the following steps:

1. computation of the mel-warped (see Sect. 5.1.1.1) spectrum with a bank of overlapping band-pass filters,
2. taking the logarithm of the magnitude of each resulting band, and
3. calculating the *Discrete Cosine Transform (DCT)* on the resulting bands. The DCT equals the real (cosine) part of an FT.

The MFCCs have been widely used in the field of speech signal processing since their introduction in 1980 [55] and have been found to be useful in music signal processing applications as well [56–59]. In the context of audio signal classification, it has been shown that a small subset of the resulting MFCCs as shown in Fig. 3.10 already contains the principal information [60, 61] — in most cases the number of used MFCCs varies in the range from 4 to 20.

The calculation is closely related to the calculation of the cepstrum as introduced in Sect. 5.3.3.7 as it transforms a logarithmic spectral representation. The main difference to the standard cepstrum is the use of a non-linear frequency scale (the mel scale) to model the non-linear human perception of pitch and the use of the DCT instead of a DFT. The mel-warped basis functions for the DCT are displayed in Fig. 3.11.

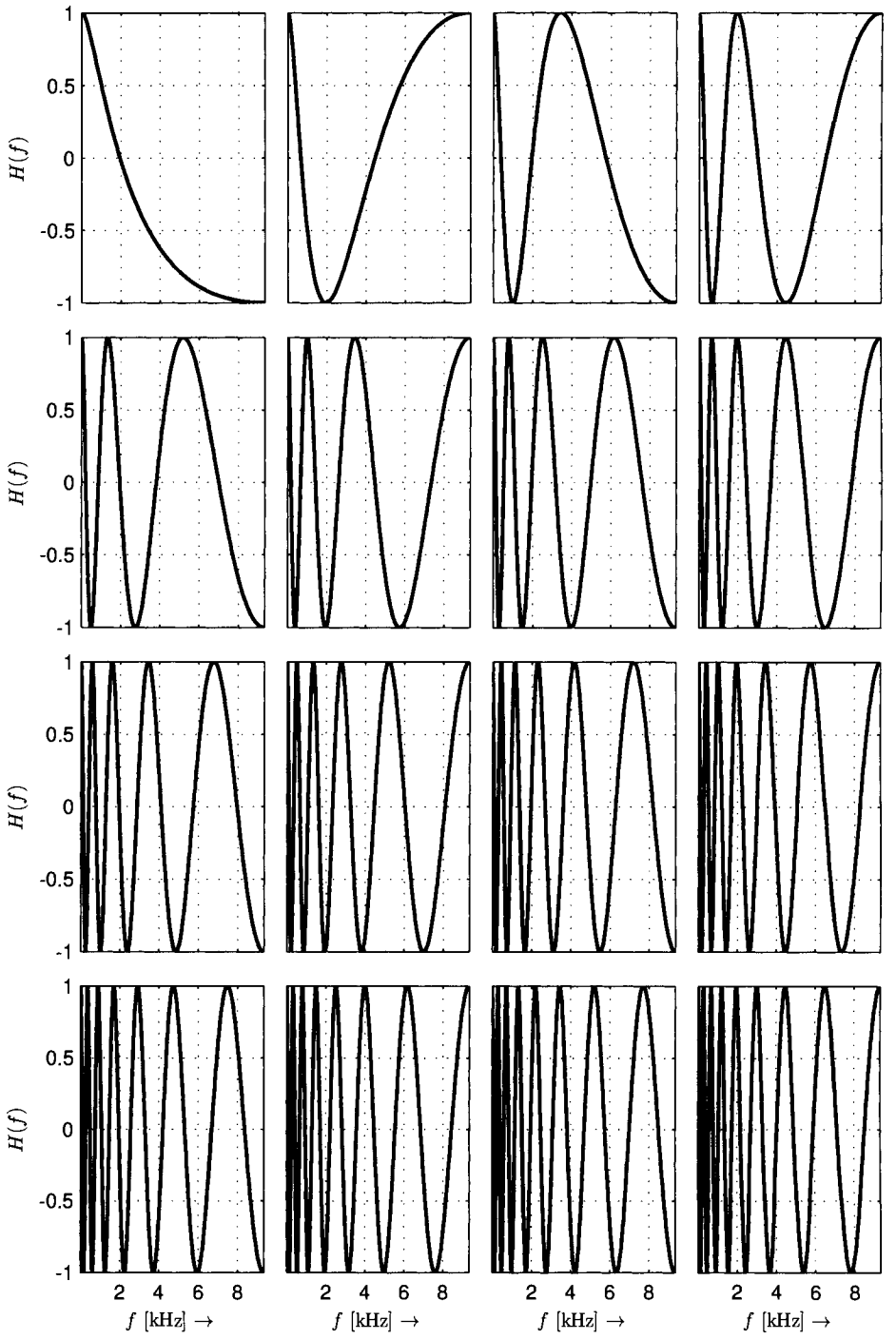

Figure 3.11 Warped cosine-shaped transformation basis functions for the computation of MFCC coefficients (order increases from left to right and from top to bottom)

Table 3.8 Properties of three popular MFCC implementations

Property	*DM*	*HTK*	*SAT*
Num. filters	20	24	40
Mel scale	lin/log	log	lin/log
Freq. range	[100; 4000]	[100; 4000]	[200; 6400]
Normalization	Equal height	Equal height	Equal area

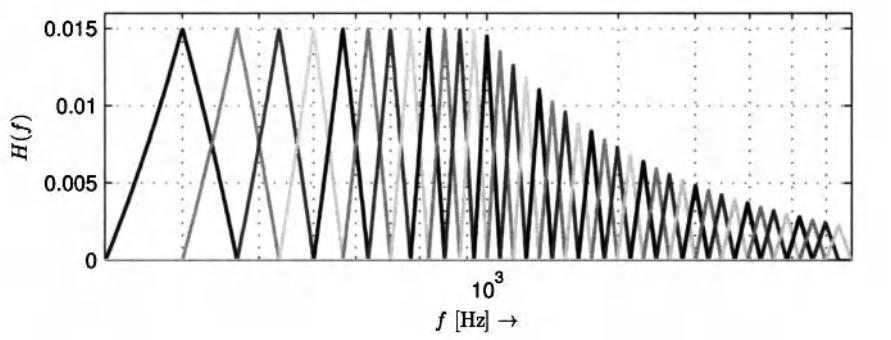

Figure 3.12 Magnitude transfer function of the filterbank for MFCC computation as used in Slaney's *Auditory Toolbox*

The mel warping of the spectrum frequently leads to the conclusion that the MFCCs are a "perceptual" feature. This is only partly true as there is no psycho-acoustic evidence to motivate the application of the DCT. Also, there is no direct correlation between the MFCCs and known perceptual dimensions.

The result of the MFCCs depends on the amplitude range of the spectral power. The zeroth MFCC $v^0_{\mathrm{MFCC}}(n)$ is usually ignored as it has no relevance in describing the timbre. It is simply a scaled measure of the energy in decibel. The MFCCs are not defined for silence as input signal.

The first four coefficients are shown in Fig. 3.10. Despite their proven usefulness it is difficult to identify non-trivial relationships to the input signal.

3.3.7.1 Common Variants

The differences between MFCC implementations can be found mainly in the computation of the mel-warped spectrum, i.e., in number, spacing, and normalization of the filters. Table 3.8 shows the differences between the three most popular MFCC implementations, the original introduced by Davis and Mermelstein (DM) [55], the implementation in the *HMM Toolkit (HTK)* software [62], and the implementation in Slaney's *Auditory Toolbox* (SAT) [27].

Figure 3.12 shows the triangular filter shapes as used in the Slaney's *Auditory Toolbox*.

Section 5.1.1.1 lists the typical mel scale models used for the non-linear frequency warping. It is also possible to use other filter shapes or to compute MFCCs directly from the power spectrum by using warped cosine basis functions as shown in Fig. 3.11 [63]. The power spectrum might also be approximated by other means such as through linear prediction coefficients.

Table 3.9 Spectral crest factor for the three prototypical spectral shapes *silence* (zero magnitude at all bins), *flat* (same amplitude at all bins), and *peak* (all bins except one have zero magnitude)

Spectral Shape	v_{Tsc}
silence	not def.
flat mag.	$2/\kappa$
single peak (@ k_s)	1

3.4 Signal Properties

3.4.1 Tonalness

Measures of *tonalness* estimate the amount of *tonal* components in the signal as opposed to noisy components. Tonalness is thus a measure related to sound quality. The somewhat unusual term *tonalness* is used here to distinguish this measure from the musical term *tonality* which describes a specific harmonic or key context. No specific measure of tonalness has itself established as de-facto standard, meaning that there are various different approaches to measure the tonalness of a signal. They have in common that for a signal considered to be tonal, they expect a high amount of periodicity and a low amount of noisy components. In that sense, the most tonal signal is a sinusoidal signal, and the most non-tonal signal is (white) noise. As an alternative to measuring the tonalness one could also find a feature for the noisiness which would be an "inverse" tonalness measure.

3.4.1.1 Spectral Crest Factor

A very simple measure of tonalness compares the maximum of the magnitude spectrum with the sum of this magnitude spectrum, a measure which will be referred to as *spectral crest factor*. It is defined by

$$v_{\mathrm{Tsc}}(n) = \frac{\max\limits_{0 \le k \le \kappa/2-1} |X(k,n)|}{\sum\limits_{k=0}^{\kappa/2-1} |X(k,n)|}. \tag{3.46}$$

The result of the spectral crest factor is a value between $2/\kappa \le v_{\mathrm{Tsc}}(n) \le 1$. Low results indicate a flat magnitude spectrum and high results indicate a sinusoidal. The spectral crest factor is not defined for audio blocks with no spectral energy (silence).

Table 3.9 shows the results for the spectral crest factor for three prototype spectral shapes.

Figure 3.13 shows the spectral crest factor for an example signal. It is low for noisy parts during the pauses and higher during the tonal passages. With decreasing amplitude of higher harmonics the spectral crest factor increases as the spectral energy is more and more concentrated at a single spectral bin.

Common Variants

A common variant is to replace the sum in the denominator by the arithmetic mean of the magnitude spectrum. This scales the range of the spectral crest factor.

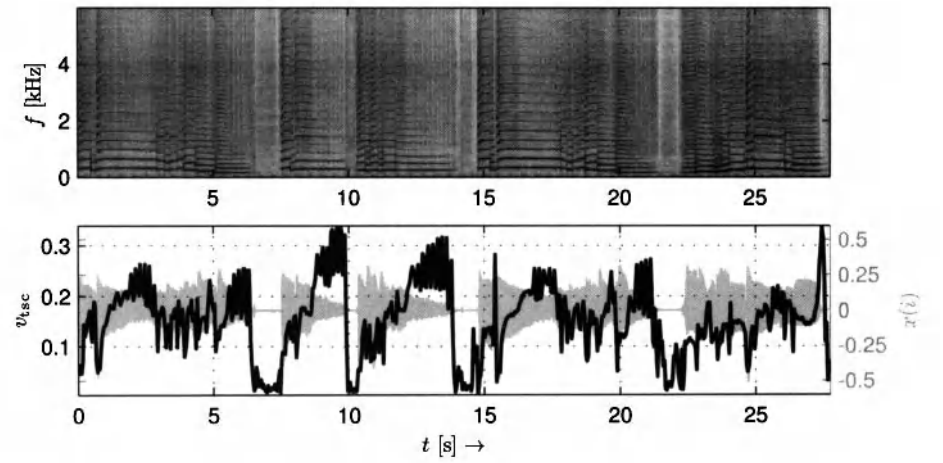

Figure 3.13 Spectrogram (top), waveform (bottom background), and tonalness feature spectral crest factor (bottom foreground) of a saxophone signal

Table 3.10 Spectral flatness for the three prototypical spectral shapes *silence* (zero magnitude at all bins), *flat* (same amplitude at all bins), and *peak* (all bins except one have zero magnitude)

Spectral Shape	v_{Tf}
silence	not def.
flat mag.	1
single peak (@ k_s)	0

3.4.1.2 Spectral Flatness

The *spectral flatness* is the ratio of geometric mean and arithmetic mean of the magnitude spectrum. It is defined by [64]

$$v_{\mathrm{Tf}}(n) = \frac{\sqrt[\kappa/2]{\prod_{k=0}^{\kappa/2-1} |X(k,n)|}}{2/\kappa \cdot \sum_{k=0}^{\kappa/2-1} |X(k,n)|} = \frac{\exp\left(2/\kappa \cdot \sum_{k=0}^{\kappa/2-1} \log\left(|X(k,n)|\right)\right)}{2/\kappa \cdot \sum_{k=0}^{\kappa/2-1} |X(k,n)|}. \tag{3.47}$$

The latter formulation uses the arithmetic mean of the logarithmic magnitude spectrum in the numerator in order to avoid problems with computing accuracy.

The result of the spectral flatness is a value larger than 0. The upper limit depends on the maximum spectral magnitude. Low results hint toward a non-flat — possibly a tonal — spectrum, while high results indicate a flat (or noisy) spectrum. The spectral flatness is thus a measure of noisiness as opposed to tonalness. However, as soon as only the magnitude at one individual bin equals 0, v_{Tf} will be zero as well.

Table 3.10 shows the results for the spectral flatness for three prototype spectral shapes. The behavior of the spectral flatness at pauses in the input signal requires special consideration as it is not defined for silence and will be comparably large for (low-level) noise.

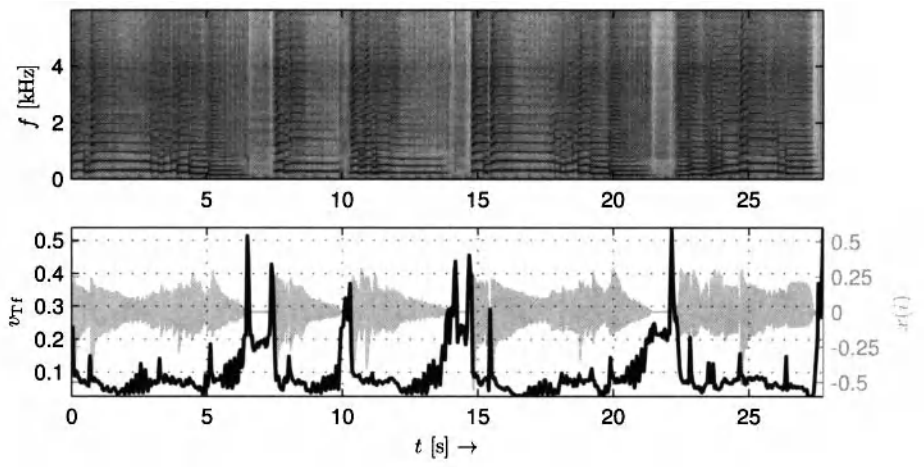

Figure 3.14 Spectrogram (top), waveform (bottom background), and tonalness feature spectral flatness (bottom foreground) of a saxophone signal

Figure 3.14 shows the spectral flatness for an example signal. It is low during tonal passages, high in noisy pauses, and produces spikes at transients.

Common Variants

It is common to use the power spectrum instead of the magnitude spectrum in order to emphasize peaks.

To avoid problems with individual zero magnitudes having too large an impact on the overall result, the magnitude spectrum can be smoothed. One typical approach is to compute the arithmetic mean of a group of neighboring spectral coefficients, which is basically the same as applying an MA filter to the magnitude spectrum. However, the length of the filter might also increase with frequency to take into account the lower frequency resolution of the human ear at higher frequencies.

In many cases, more useful information can be gathered if the spectral flatness calculation takes only magnitudes within a pre-defined frequency range into account, as opposed to computing it from the whole spectrum. The MPEG-7 standard recommends a frequency range of from 250 Hz to 16 kHz, divided into 24 slightly overlapping frequency bands with quarter-octave bandwidth [33]. Since the spectral flatness is then computed for each individual frequency band, the result per STFT is a vector of spectral flatness results.

3.4.1.3 Tonal Power Ratio

A straightforward way of computing the tonalness of a signal is to compute the ratio of the tonal power $E_{\mathrm{T}}(n)$ to the overall power:

$$v_{\mathrm{Tpr}} = \frac{E_{\mathrm{T}}(n)}{\sum_{i=0}^{K/2-1} |X(k,n)|^2}. \tag{3.48}$$

The interesting part in this case is naturally the estimation of the power of the tonal components. An approach to detecting tonal components in the spectrum is outlined in

Table 3.11 Tonal power ratio for the three prototypical spectral shapes *silence* (zero magnitude at all bins), *flat* (same amplitude at all bins), and *peak* (all bins except one have zero magnitude)

Spectral Shape	v_{Tpr}
silence	not def.
flat mag.	0
single peak (@ k_s)	1

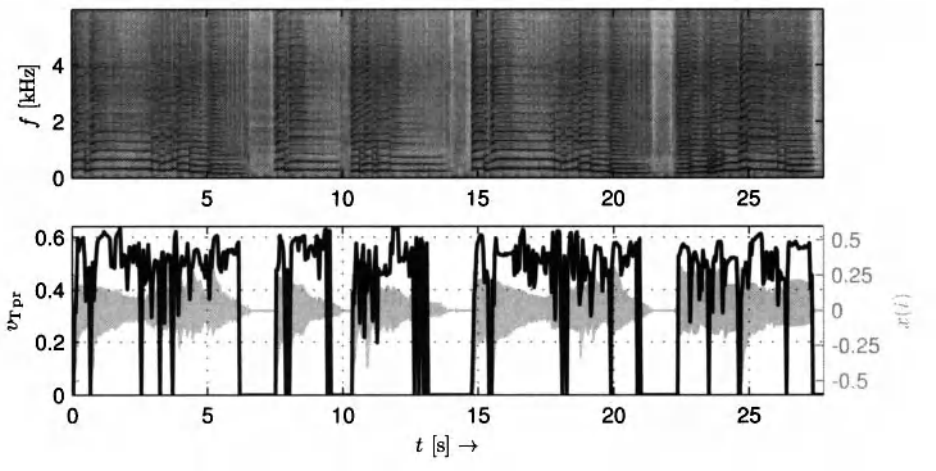

Figure 3.15 Spectrogram (top), waveform (bottom background), and tonalness feature tonal power ratio (bottom foreground) of a saxophone signal

Sect. 5.3.2.2. A simpler approximation to estimating the tonal energy is summing all bins k which

- are a local maximum: $|X(k-1,n)|^2 \leq |X(k,n)|^2 \geq |X(k+1,n)|^2$ and
- lie above a threshold G_T.

The result of the tonal power ratio is a value between $0 \leq v_{\mathrm{Tpr}} \leq 1$. Low results hint toward a flat (noisy) spectrum or a block with low input level while high results indicate a tonal spectrum.

Table 3.11 shows the results for the tonal power ratio for three prototype spectral shapes. The behavior of the spectral flatness at pauses in the input signal requires special consideration as it is not defined for silence and will be comparably large for (low-level) noise.

Figure 3.15 shows the tonal power ratio for an example signal. It is zero in noisy pauses, high for tonal passages and, in the case of this simple saxophone signal, drops distinctively at the initial transients at note beginnings.

3.4.1.4 Maximum of Autocorrelation Function

The ACF of a time signal yields local maxima where the ACF lag matches the wavelengths of the signal-inherent periodicities (see Sect. 2.2.6.2). The less periodic and therefore less tonal the signal is, the lower is the value of such maxima. The absolute value of the overall *ACF maximum* is therefore a simple estimate of the signal's tonalness:

$$v_{\mathrm{Ta}}(n) = \max_{0 \leq \eta \leq K-1} |r_{xx}(\eta, n)|. \tag{3.49}$$

Values in the main lobe of the ACF around lag $\eta = 0$ have to be discarded to ensure more reliable results. Different approaches can be used to ignore the main lobe:

- *Minimum lag*: Assuming that a maximum of interest will not be found at high frequencies (small lags and period lengths, respectively), the search for the maximum can be started at a pre-defined lag, ignoring values at smaller lags. The lower the expected maximum frequency is, the larger can the minimum lag can be. Depending on the task at hand and the sample rate, the maximum frequency might be too high to correspond to a reasonably large minimum lag. For example, at a sample rate of 48 kHz a frequency of 9.6 kHz corresponds to a lag of 5 samples, a frequency of 4.8 kHz to a lag of 10 samples, and a frequency of 1920 Hz to a lag of 25 samples.

- *Minimum magnitude threshold*: Maxima are only detected at lags larger than the lag η_{r}. This lag is the smallest lag at which r_{xx} crosses a pre-defined threshold G_{r}

$$\eta_{\mathrm{r}} = \operatorname*{argmin}_{0 \leq \eta \leq K-1} \left(r_{xx}(\eta) < G_{\mathrm{r}} \right). \tag{3.50}$$

  Theoretically, however, the threshold might never be crossed; this case has to be considered in the implementation.

- *Search range from the first local minimum*: Only consider maxima at lags larger than the lag of the "first" local minimum. The idea is to avoid the detection of "insignificant" local maxima in the main lobe around lag $\eta = 0$, but depending on the signal, a local minimum might be detected at a very low lag.

The best solution is to combine these approaches and possibly to find additional ways fitted to the problem to ensure meaningful results.

The result is a value between $0 \leq v_{\mathrm{Ta}}(n) \leq 1$. This ACF-based feature will work more reliable for monophonic signals or signals with a limited number of fundamental frequencies. Low results indicate a non-periodic signal and high results a periodic signal.

Table 3.12 Tonalness feature acf maximum for the three prototypical signal types *silence* (zero magnitude at all samples), *white noise* and a sinusoidal signal

Spectral Shape	v_{Ta}
silence	0
white noise	0
sinusoidal	1

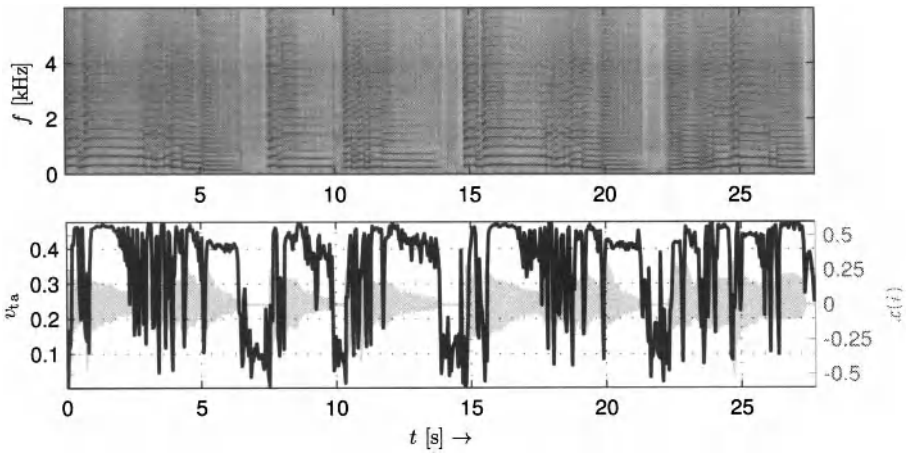

Figure 3.16 Spectrogram (top), waveform (bottom background), and tonalness feature ACF maximum (bottom foreground) of a saxophone signal

Table 3.12 shows the results for the ACF maximum for three prototype signals.

Figure 3.16 shows the ACF maximum for an example signal. As expected, there is the tendency of giving low values at noisy pause segments and higher values for tonal segments.

3.4.1.5 Predictivity Ratio

The *predictivity ratio* is a measure of how well the audio signal can be predicted by $\mathcal{O}$-order linear prediction (see Sect. 2.2.7). Each sample i is predicted using the preceding sample values and the prediction coefficients b_j:

$$\hat{x}(i) = \sum_{j=1}^{\mathcal{O}} b_j \cdot x(i-l). \tag{3.51}$$

The less noisy the signal is, the smaller the error e_{P} between the original and the predicted signal will be. Periodic and thus tonal signals will yield small prediction errors while noisy signals will result in high prediction errors. The power of the prediction error is therefore a measure of tonalness or more precisely a measure of noisiness as it will approach 0 for

Table 3.13 Tonalness feature predictivity ratio for the three prototypical signal types *silence* (zero magnitude at all samples), *white noise*, and a sinusoidal signal

Spectral Shape	v_{Tp}
silence	not def.
white noise	high
sinusoidal	$\rightarrow 0$

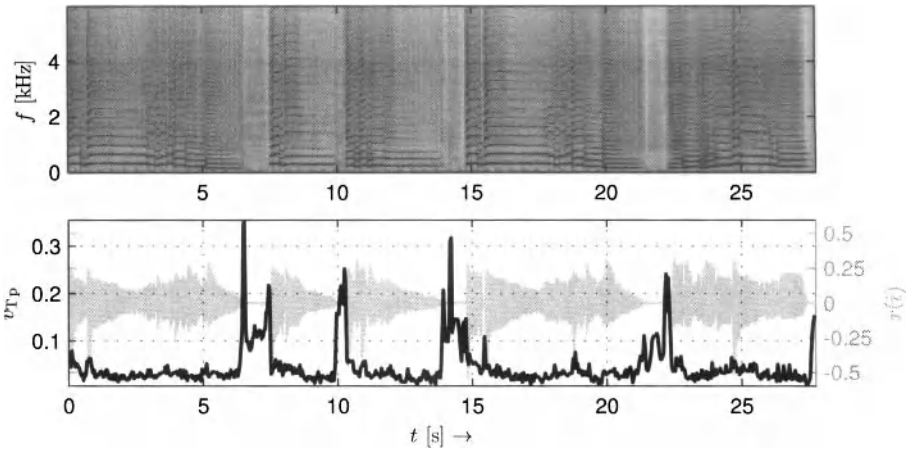

Figure 3.17 Spectrogram (top), waveform (bottom background), and tonalness feature predictivity ratio (bottom foreground) of a saxophone signal

tonal signals:

$$v_{\mathrm{Tp}}(n) = \sqrt{\frac{\sum_{i=i_s(n)}^{i_e(n)} \left(x(i) - \hat{x}(i)\right)^2}{\sum_{i=i_s(n)}^{i_e(n)} x^2(k)}}. \tag{3.52}$$

The result is a value larger or equal to 0. Low results indicate a periodic signal and high results a non-periodic signal.

Table 3.13 shows the results for the predictivity ratio for three prototype signals.

Figure 3.17 shows the predictivity ratio for an example signal with a predictor length of 12 coefficients. It clearly separates the noisy pause segments (high values) from the tonal segments (low values); the individual tonal parts cannot be distinguished with the feature.

3.4.1.6 Spectral Predictivity

The *spectral predictivity* is a measure of tonalness computed from overlapping STFTs. The magnitude and phase of each spectral bin are predicted with a simple predictor with fixed coefficients:

$$|\hat{X}(k,n)| = 2 \cdot |X(k,n-1)| - |X(k,n-2)|, \tag{3.53}$$

$$\hat{\phi}_X(k,n) = 2 \cdot \Phi_X(k,n-1) - \Phi_X(k,n-2). \tag{3.54}$$

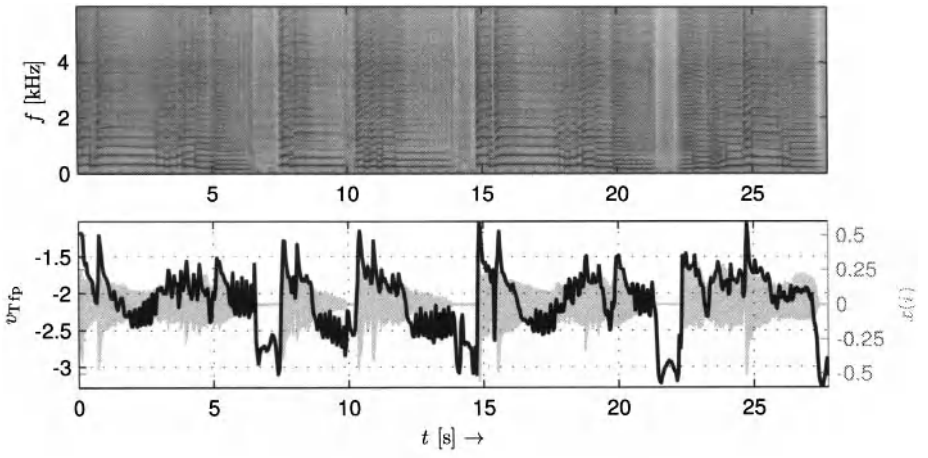

Figure 3.18 Spectrogram (top), waveform (bottom background), and tonalness feature spectral predictivity (bottom foreground) of a saxophone signal

The prediction error $e_{\mathrm{Tfp}}(k,n)$ is then defined by

$$e_{\mathrm{Tfp}}(k,n) = \left| \frac{|X(k,n)|e^{\Phi_X(k,n)} - |\hat{X}(k,n)|e^{\hat{\phi}_X}(k,n)}{|X(k,n)| \cdot |\hat{X}(k,n)|} \right| \tag{3.55}$$

and the resulting tonalness

$$v_{\mathrm{Tfp}}(n) = \frac{c_1 + c_2 \cdot \sum_{k=0}^{K/2-1} \log\left(e_{\mathrm{Tfp}}(k,n)\right)}{K/2} \tag{3.56}$$

with c_1 and c_2 as constants to be arbitrarily selected (MPEG: $c_1 = -0.299$, $c_2 = -0.43$).

The spectral predictivity as shown in Fig. 3.18 is used in the psycho-acoustic model II of the audio coding standards by the *Motion Picture Experts Group (MPEG)* [65].

3.4.2 Autocorrelation Coefficients

Infrequently, the first *autocorrelation coefficients* are used directly to describe statistical properties of the signal

$$v_{\mathrm{ACF}}^{\eta}(n) = r_{xx}(\eta, n) \quad \text{with } \eta = 1, 2, 3, \ldots \tag{3.57}$$

with $r_{xx}(\eta, n)$ being the ACF as defined in Sect. 2.2.6.2.

The number of used coefficients usually varies between 6 and 64 depending on requirements and sample rate. Each coefficient is in the range of $-1 \leq r_{xx}(\eta, n) \leq 1$. The faster the coefficients decrease with increasing lag, the "whiter" the signal can be assumed to be.

Table 3.14 shows typical results for the autocorrelation coefficient for three prototype signals.

Figure 3.19 shows the autocorrelation coefficient at $\eta = 20$ for an example signal. The lower the input frequency and the more tonal the signal is, the higher the coefficient is. During the pauses, it drops toward 0.

Table 3.14 Autocorrelation coefficient for the three prototypical signal types *silence* (zero magnitude at all samples), *white noise*, and a sinusoidal signal

Input Signal	v_{ACF}
silence	not def.
white noise	≈ 0
sinusoidal	high

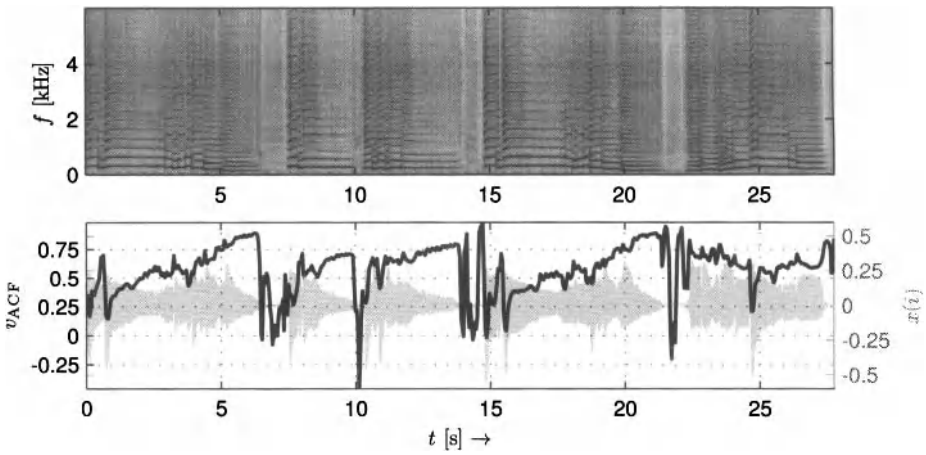

Figure 3.19 Spectrogram (top), waveform (bottom background), and autocorrelation coefficient 20 (bottom foreground) of a saxophone signal

3.4.3 Zero Crossing Rate

The number of changes of sign in consecutive blocks of audio samples — the *zero crossing rate* — is a low-level feature that has been used for decades in speech and audio analysis due to its simple calculation:

$$v_{\mathrm{ZC}}(n) = \frac{1}{2 \cdot \mathcal{K}} \sum_{i=i_s(n)}^{i_e(n)} \left| \operatorname{sign}\left[x(i)\right] - \operatorname{sign}\left[x(i-1)\right] \right| \tag{3.58}$$

with the sign function being defined by

$$\operatorname{sign}\left[x(k)\right] = \begin{cases} 1, & \text{if } x(i) > 0 \\ 0, & \text{if } x(i) = 0 \\ -1, & \text{if } x(i) < 0 \end{cases} \tag{3.59}$$

and $x(i-1) = 0$ used as initialization if $x(i-1)$ does not exist.

The output will be a value in the range of $0 \leq v_{\mathrm{ZC}}(n) \leq 1$. The more often the signal changes its sign, the more high-frequency content can be assumed to be in the signal. Furthermore, the more the zero crossing rate varies over blocks, the less periodic the signal can be assumed to be. The concept of the zero crossing rate is based on input signals with an arithmetic mean of approximately 0.

Table 3.15 Zero crossing rate for the three prototypical signal types *silence* (zero magnitude at all samples), *white noise*, and a sinusoidal signal

Input Signal	v_{ZC}
silence	0
white noise	high
sinusoidal	period lengths per block times 2

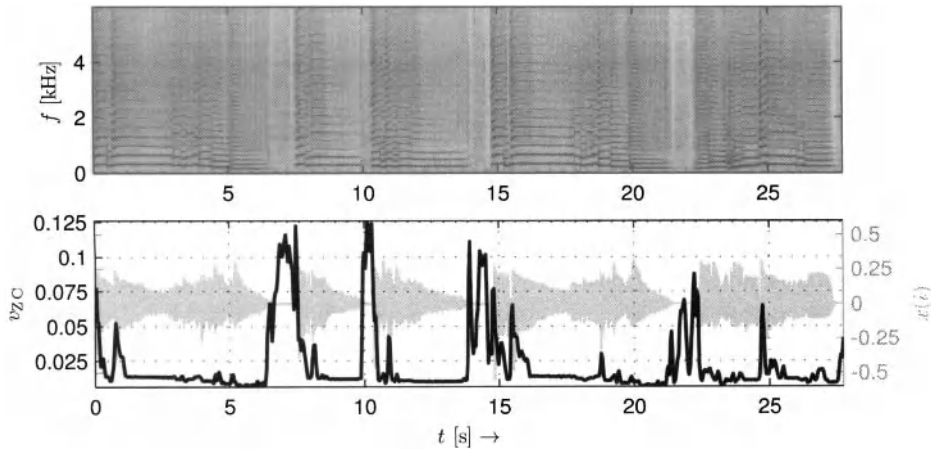

Figure 3.20 Spectrogram (top), waveform (bottom background), and zero crossing rate (bottom foreground) of a saxophone signal

Table 3.15 shows the results for the zero crossing rate for three prototype signals.

Figure 3.20 shows the zero crossing rate for an example signal. Unsurprisingly, it is high for noisy parts and low for tonal parts. Its usability for fundamental frequency detection (see Sect. 5.3.3.1) is indicated by the long constant values during constant pitches.

3.4.3.1 Common Variants

The zero crossing rate has been used for both measuring the tonalness of a signal as introduced in Sect. 3.4.1 and estimating its fundamental frequency by assuming a sinusoidal input signal and then relating the number of zero crossings directly to the fundamental frequency. To improve the robustness of such attempts, the input signal can be low-pass filtered to suppress high-frequency content interaction with the feature result. The cut-off frequency of the low-pass filter then should be chosen as low as the highest expected fundamental frequency to ensure maximum suppression of high-frequency content.

3.5 Feature Post-Processing

The result of the feature extraction process is a series of feature values that can — dependent on the use case — be processed, transformed, and selected.

It is quite common to compute a large number of features. Formally, they can be represented in a matrix:

$$\begin{aligned} \boldsymbol{V} &= [\boldsymbol{v}(0)\ \boldsymbol{v}(1)\ \ldots\ \boldsymbol{v}(\mathcal{N}-1)] \\ &= \begin{bmatrix} v_0(0) & v_0(1) & \ldots & v_0(\mathcal{N}-1) \\ v_1(0) & v_1(1) & \ldots & v_1(\mathcal{N}-1) \\ \vdots & \vdots & \ddots & \vdots \\ v_{\mathcal{F}-1}(0) & v_{\mathcal{F}-1}(1) & \ldots & v_{\mathcal{F}-1}(\mathcal{N}-1) \end{bmatrix} \end{aligned} \tag{3.60}$$

with the number of rows being the number of features $\mathcal{F}$ and the number of columns being the number of blocks $\mathcal{N}$. Each vector $\boldsymbol{v}(n)$ consists of $\mathcal{F}$ feature values at block n and is called an *observation*.

3.5.1 Derived Features

It is possible to generate new features from the previously extracted features. These new derived features do not have to replace the original features; they can be added as complementary features. Sometimes these derived features are called subfeatures, but we will use the term *subfeature* in a slightly different context as described in Sect. 3.5.3.

In some cases the detection of (sudden) changes of feature values is of special interest as it may mark the start or end of important segments, for example, note onsets and structural boundaries. A simple way of analyzing these changes is to compute the difference between consecutive feature results (which would be called *derivative* if it were a continuous function):

$$v_{j,\Delta}(n) = v_j(n) - v_j(n-1). \tag{3.61}$$

The resulting series $v_{j,\Delta}(n)$ is either one value shorter than the corresponding series $v_j(n)$ or an appropriate initialization for $v_j(-1)$ has to be defined. The time stamp $t_{\mathrm{s},\Delta}(n)$ of $v_{j,\Delta}(n)$ would be

$$t_{\mathrm{s},\Delta}(n) = \frac{t_{\mathrm{s}}(n) + t_{\mathrm{s}}(n-1)}{2} \tag{3.62}$$

with $t_s(n)$ being the time stamp of $v_j(n)$.

Computing the derivative has the character of a high-pass filter; it is also common to do the opposite, namely to smooth out $v_i(n)$ with a low-pass filter. This allows us to focus on the long-term variations of the feature. In general it is beneficial if the used filter has a zero phase or linear phase response in order to ensure correct timing properties, therefore either an MA filter as introduced in Sect. 2.2.1.1 can be used or any IIR filter applied twice forward and backward on the series to produce the low-pass filtered series $v_{j,\mathrm{LP}}(n)$ (see Sect. 2.2.1.2).

The features usually are the input of the second processing step of an ACA system, namely a classification system, a distance measure, or some other system for feature interpretation. Depending on the classifier and feature selection algorithm used and given a training database (compared to the number of features), it may be of interest to have as many raw input features as possible or at least to have a large feature data set from which to choose. While linear combinations of features are frequently already covered by the selection algorithm (see below), non-linear combinations such as the multiplication of two series of features can theoretically improve results in certain cases. An example would be

$$v_{jl}(n) = v_j(n) \cdot v_l(n). \tag{3.63}$$

The usage of such derived features is frequently met with only limited success as most of the information of interest can already be found in the original features.

3.5.2 Normalization and Mapping

When different features are combined into vectors $\boldsymbol{v}(n)$, their different output ranges and distributions might become a problem. This is, for example, the case when computing the Euclidean distance because one feature may have more impact on the result than another. Consider two identical features with identical distribution except for an amplitude scale factor λ. Each observation contains the two features. When computing the Euclidean distance, the second dimension's distance will have the weight λ^2 while the first dimension will be weighted with 1. Large λ will thus let the second feature dominate the distance while small λ will render the second dimension superfluous.

A common approach to normalize features if they all have *symmetric* distributions with *identical shape* is to remove their mean value and scale them to a variance of 1 (see, e.g., [66]):

$$v_{j,\mathrm{N}}(n) = \frac{v_j(n) - \mu_{v_j}}{\sigma_{v_j}}. \tag{3.64}$$

However, if the distributions do not have identical shape this normalization is not applicable. In that case, other approaches are necessary to ensure a correct combination of features.

3.5.2.1 Feature Distribution

To get similar feature distributions, a target distribution has to be chosen to which all the features will be transformed if necessary. In most cases, a Gaussian distribution is chosen as target.

Several approaches exist to transform a given distribution into a Gaussian distribution; widely used is the *Box-Cox transform* [67]. One example of this transform is

$$v^{(\lambda)} = \begin{cases} \frac{v^\lambda - 1}{\lambda}, & \lambda \neq 0 \\ \log(v), & \lambda = 0 \end{cases} \tag{3.65}$$

with the parameter λ to be estimated. The Box-Cox transform only considers a limited class of transformation functions and thus does not guarantee that any arbitrary distribution can be mapped to a Gaussian distribution.

There are also numerical methods of finding appropriate feature transformations. One example is the work of Albada and Robinson who transform arbitrary distributions to the normal distribution [68]. The transformation function for every feature has then to be stored numerically.

In many practical applications it is sufficient to transform selected features in a way that results only in roughly approximated Gaussian distributions. Only features failing a test for Gaussianity have to be subjected to a transformation. Statistical procedures to test a distribution for Gaussianity are, for example, the Kolmogorov-Smirnov test, the Lilliefors test, the Shapiro-Wilk test [69], and the Anderson-Darling test [70]. Thode gives a good introductory overview [71]. Unfortunately, the result of these statistical tests is only of limited use in ACA because these tests nearly always tend to fail for large numbers of observations. In these cases it is more practical to compute both the skewness (see Sect. 3.2.7)

and possibly the kurtosis (see Sect. 3.2.8) of the features to determine how *Gaussian* their distribution is. As a rule of thumb, distributions with a skewness smaller than 2 are not significantly skewed and can thus be assumed to be symmetric [72].

3.5.2.2 Feature Normalization

The standard approach to feature *normalization* has been given in Eq. (3.64). When the feature distribution is not shaped like a Gaussian distribution, it can make more sense to normalize it with respect to its median $Q_v(0.5)$ as described in Eq. (3.25). The normalized feature is then

$$v_{j,\mathrm{N}}(n) = \frac{v_j(n) - Q_{v_j}(0.5)}{s_{v_j}} \tag{3.66}$$

with s_{v_j} being the root mean squared deviation from the median

$$s_{v_j} = \sqrt{\frac{1}{\mathcal{N}} \sum_{n=0}^{\mathcal{N}-1} \left(v_j(n) - Q_{v_j}(0.5)\right)^2}. \tag{3.67}$$

The normalization of multi-dimensional features requires special consideration and depends on both the feature characteristics as well as the specific use case.

3.5.3 Subfeatures

Each feature still represents a time series, and it is common to compute a time-independent summary of each feature per sliding texture window (the term *texture window* has been introduced in Sect. 2.2.2). The summary feature is frequently called *subfeature*. The most common subfeatures would be the arithmetic mean and the standard deviation of the feature or, more general, basically every statistical measure presented in Sect. 2.1.4. Furthermore, it is basically possible to use each single feature presented above as a description of the feature time series by computing the feature of a feature, although certainly not every combination would make sense. A relatively large number of possible meaningful subfeatures in the context of musical genre classification was presented by Mörchen et al. [73].

3.5.4 Feature Dimensionality Reduction

Although large numbers of features can easily be extracted from the audio data, it is unclear a priori which features will help the final interpretation or classification stage of an ACA system. In other words, it is not known which features are of relevance.

The accuracy of a pattern recognition system or classifier will not necessarily improve with an increasing number of features. A large number of features increases the likelihood of *overfitting* to occur during the training. Overfitting means that the classifier starts to learn training set specific characteristics which cannot be generalized to the use case. The classifier will perform poorly on unknown input data. Therefore, the number of features has to be reduced and fitted to the amount of available training data.[3] The optimal number of features is difficult to determine; typically it is simply the number of features that maximize

[3]There exist classifiers which are relatively robust against such dimensionality issues while other classifiers are where susceptible. The danger of overfitting should be carefully considered for each combination of classifier type, the number of features, and the size of the training data set.

the classification accuracy of the training set using *N-fold cross validation* (see Sect. 8.1.3). To estimate the minimum feature performance, it is helpful to add an additional feature consisting of random noise to the feature set.

We can define the following criteria for a feature to be helpful:

- *high "discriminative" or descriptive power* since the feature should be suitable to the task at hand,
- *non-correlation to other features* because each feature should add new information to avoid redundancy,
- *invariance to irrelevancies* to allow the feature to be robust against, e.g., linear transformations of the input audio signal such as scaling and filtering operations (low-pass filtering, reverberation), the addition of signals such as (background) noise, coding artifacts as well as the application of non-linear operations such as distortion and clipping (see Wegener et al. for an example evaluation of feature robustness [74]), and
- *reasonable computational complexity* to ensure that the feature is able to be computed on the target platform (such as a mobile device) and for the required application, respectively.

We will mainly focus on the first two criteria; the third criterion can be easily tested by adding modified audio files to the test set, and the fourth criterion is too application-specific to be dealt with in a general way.

There are two different approaches to reduce the dimensionality of the feature space:

- *feature subset selection* to discard specific features, and
- *feature space transformation* to transform the features to a lower dimensional space.

In the first case the most promising feature subset is chosen; in the latter case only those dimensions in the transformed feature space are discarded that contribute the least information for the target application.

Note that the requirement of feature dimensionality reduction strongly depends on the classification algorithm used; furthermore, it is still under discussion whether complex methods of dimensionality reduction really outperform simpler ones [75].

3.5.4.1 Feature Subset Selection

The aim of *feature subset selection* is to reduce the number of used features by discarding the least powerful features. Formally, the available feature set

$$\mathcal{V} = v_j|_{j=1,\ldots,\mathcal{F}} \tag{3.68}$$

should be reduced to the feature subset

$$\mathcal{V}_\mathrm{s} = v_j|_{j=1,\ldots,\mathcal{F}_\mathrm{s}} \tag{3.69}$$

with $\mathcal{F}_\mathrm{s} < \mathcal{F}$; the subset is chosen to optimize a given objective function $J(\mathcal{V}_\mathrm{s})$.

Feature selection algorithms are called *wrapper methods* if the objective function is the classifier itself and *filter methods* if the objective function $J(\mathcal{V}_\mathrm{s})$ is independent of the classification system used. Filter methods select features based on properties a good feature set is presumed to have and are usually computationally less expensive than wrapper methods.

This section will only provide a short introduction to the most common approaches to feature subset selection. More in-depth surveys of this topic have been published by Guyon and Elisseeff [76] and Cantú-Paz et al. [77].

Examples for wrapper methods are

- *Brute Force Subset Selection*

  The most obvious way of finding the optimal feature subset is to compute the classification accuracy for all possible combinations of features and to select the subset that performed best. The disadvantage of this approach is that the number of subsets to test, i.e., to train and evaluate, will be $2^{\mathcal{F}}$. This renders this method impractical for large numbers of features $\mathcal{F}$.

- *Single Variable Classification*

  A simple feature ranking can be obtained by calculating the classification accuracy for each individual feature. This enables the identification of features performing very poorly individually. While discarding the features that perform worst seems to be an intuitive solution, there are two problems with selecting or discarding features this way:

  1. The feature ranking contains no information on the *correlation* of two or more features. Consider the case of two features with identical values. They will both have the same ranking which is possibly high, but leaving both in the selected feature subset cannot improve classifier performance (and might even harm it in the case of simple classification algorithms).
  2. The feature ranking contains no information on the *combined usefulness* of features. A feature that adds no information individually might be able to add information in combination with other features.

- *Sequential Forward Selection*

  Sequential forward selection starts with an empty subset of features. In the first iteration, it considers all feature subsets with only one feature similar to the *single variable classification* approach method. The subset with the highest classification accuracy is used as the basis for the next iteration. The iterative algorithm can be structured into the following processing steps:

  1. Start with an empty feature subset $\mathcal{V}_{\mathrm{s}} = \emptyset$.
  2. Find the one feature v_j not yet included in the feature subset that maximizes the objective function
  $$v_j = \underset{\forall j | v_j \notin \mathcal{V}_{\mathrm{s}}}{\operatorname{argmax}} J(\mathcal{V}_{\mathrm{s}} \bigcup v_j). \tag{3.70}$$
  3. Add feature v_j to $\mathcal{V}_{\mathrm{s}}$.
  4. Go to step 2 and repeat the procedure until the required number of features has been selected or the required classification accuracy has been reached.

- *Sequential Backward Elimination*

  Sequential backward elimination works in an analogous way to sequential forward selection but starts with a full subset of features and iteratively removes features from the subset. It is computationally less efficient than sequential forward selection. This is particularly true for large feature sets. Sequential backward elimination can be argued to give better results since sequential forward selection does not assess the importance of features in combination with other not yet included features.

There exist many filter methods for finding variable rankings. The methods include, for example, chi-square statistics or any arbitrary class separability measure. One common example of a filter yielding an implicit feature ranking is *Principal Component Analysis (PCA)*. The concepts of PCA are summarized in Appendix C. It can be used for feature selection by examination of the transformation matrix $\boldsymbol{T}$. The idea is to keep the features with major influence on the principal components and to eliminate features with major influence on the components with low variance.

A simple rule for feature elimination is to start with the component with the smallest eigenvector, discard the feature which contributes most to this component, proceed to the next-smallest eigenvector, and repeat the procedure until all features have been ranked. Then, an arbitrary number of features can be discarded.

3.5.4.2 Feature Space Transformation

The objective of *feature space transformation* is to reduce the number of used features by transforming them into a lower dimensional space.

The disadvantage of using transformations for dimensionality reduction is that the transformed features cannot be interpreted as easily as the original features since the transformed features are linear combinations of the original features. Examples of tools for feature space transformation are

- *Principal Component Analysis*

  Transforming the data set with PCA (see Appendix C) does not reduce the dimensionality of the data by itself. However, the new dimensions can be sorted according to the variance they contribute to the data which can be seen as a measure of importance. Discarding the components that account for low variance is therefore a viable approach.

  A widely used systematic criterion to decide how many components can be discarded is based on the eigenvalue. This approach is based on the assumption that every component with an eigenvalue lower than 1 can be discarded. This criterion is equivalent of a threshold of $1/\mathcal{F}$ for the relative variance for which a component accounts.

  In many cases, this criterion leaves more components in the data set than useful. A slightly more "hands-down" approach is to identify either the index after which the eigenvalues are significantly lower or the index after which eigenvalues tend to be very similar to each other.

- *Other Transformation Methods*

  Other transformation methods can be used for feature space transformation but will not be explained in detail in this book. Typical approaches that can be found in the literature are *Independent Component Analysis (ICA)* and *Singular Value Decomposition (SVD)*. *Linear Discriminant Analysis (LDA)* also transforms the feature space, however, the number of output components cannot be chosen freely in this case. The main distinction of LDA and PCA is that PCA maximizes the variance and LDA maximizes class separability.

CHAPTER 4

INTENSITY

Intensity, magnitude, and loudness-related features constitute one of the most commonly used classes for the description of audio content. Most audio editors and digital audio workstations illustrate the audio signal in its waveform view, its amplitude variation over time. There also exists a variety of instruments for level, volume, and loudness measurements frequently used in recording studio environments.

Many of the presented features are instantaneous features similar to the features introduced in Chap. 3. Therefore the same post-processing options (see Sect. 3.5) can be applied to most of the features introduced in the following.

4.1 Human Perception of Intensity and Loudness

It is important to distinguish the meaning of the terms *intensity* and *loudness*. Intensity means a physical, measurable entity such as the magnitude of a sound while loudness refers to a perceptual entity that can only be measured via responses of human observers [78]. Unfortunately, the term *loudness* is also used for *algorithmic models* of the perceived loudness.

Human perception of intensity is related to the magnitude of the audio signal in a way that if the signal's magnitude is scaled up, the perceived loudness will increase as well. It has been discovered very early that this relationship is non-linear; a linear increase in the signal's magnitude or power will not result in a linear increase in perceived loudness. An approximately linear relation could be found by using the pseudo-unit *decibel* (dB) which

An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics, First Edition. Alexander Lerch.

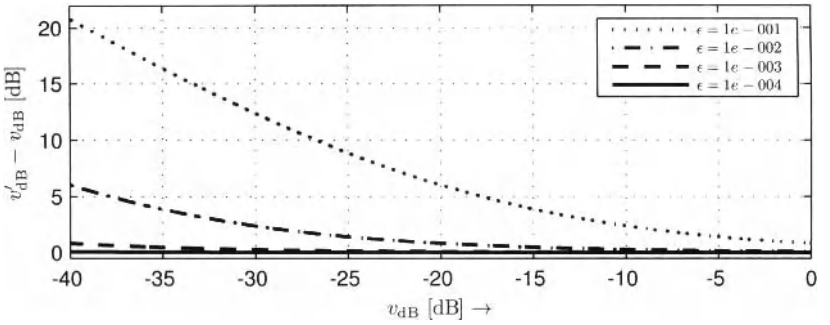

Figure 4.1 Level error introduced by adding a small constant ϵ to the argument of a logarithm

is computed by taking the logarithm of the intensity feature $v(n)$ (computed from a block of samples, see below):

$$v_{\mathrm{dB}}(n) = 20 \cdot \log_{10}\left(\frac{v(n)}{v_0}\right) \tag{4.1}$$

with v_0 representing a reference constant. In the digital domain, dealing with audio amplitudes in the range of $[-1; 1]$, it is commonly set to $v_0 = 1$. The resulting level unit is then referred to as dBFS (dB *full scale*) and has a range of $-\infty \leq v_{\mathrm{dB}}(n) \leq 0\,\mathrm{dB}$. The scaling factor 20 has been chosen to scale the non-linear function so that 1 dB roughly represents the level difference a human can easily recognize. This is only a rough approximation as the actual so-called *Just Noticeable Difference in Level (JNDL)* depends on the stimulus level and partly on stimulus frequency and masking effects [47].

Computing the logarithm of the feature v is not possible for silence $v(n) = 0$. The calculation of $\log(0)$ is commonly avoided by adding a small constant ϵ, resulting in

$$v_{\mathrm{dB}}(n) = 20 \cdot \log_{10}(v(n) + \epsilon). \tag{4.2}$$

The choice of ϵ determines the measurement accuracy at low-level inputs. The measurement error increases for decreasing $v(n)$ approaching ϵ. It will be 6 dB for $v(n) = \epsilon$ and increase with lower levels. Figure 4.1 visualizes the error amount for different ϵ. Alternatively, the input feature values may be truncated at ϵ to yield a correct value for magnitudes greater or equal than ϵ at the cost of an additional `if` statement per feature value:

$$v_{\mathrm{trunc}}(n) = \begin{cases} v(n), & \text{if } v(n) \geq \epsilon \\ \epsilon, & \text{otherwise} \end{cases}. \tag{4.3}$$

The decibel scale is not a loudness scale since equal-sized steps on the decibel scale are not perceived as equal-sized loudness steps by human listeners: doubling the level in dB does not result in doubling the perceived loudness of a sound. Stevens, summarizing several loudness perception studies, proposed a simple rule of thumb stating that a doubling of the perceived loudness corresponds to a level increase of 10 dB [79].

More accurate models of the human perception of loudness take both the input signal's frequency and the cochlea's frequency resolution into account as has been shown by many researchers during the last century. The most important researchers in the history of loudness perception are probably Fletcher and Munson [80, 81], Stevens [78, 79, 82], Moore [83], and Zwicker and Fastl [47].

4.2 Representation of Dynamics in Music

In a traditional musical score, loudness-related performance instructions are rather vague. Usually only five to eight different dynamic steps are used to describe musical dynamics (e.g., ***pp***: *pianissimo*, ***p***: *piano*, ***mf***: *mezzoforte*, ***f***: *forte*, ***ff***: *fortissimo* for the dynamic range from "very soft" to "very strong"), complemented by indications of smooth loudness transitions (e.g., *crescendo* or *decrescendo* for increasing and decreasing loudness, respectively) and dynamic accents (e.g., ***sf***: *sforzando*). These written instructions do not directly refer to absolute loudness as it would also depend on a number of other influencing factors such as instrumentation, timbre, number of voices, performance, and musical tension and musical context. The indifference to absolute loudness may also be illustrated by the fact that while listening to a recording on a hi-fi system, the reproduction volume may be manipulated without loosing the *piano* or *forte* character of the performance. Still, Nakamura has shown that measures of intensity or loudness can to a certain degree be used as indications of musical dynamics [84]. A loudness-related performance attribute is the *tremolo*, the periodic modulation of loudness over time. It usually appears in combination with a vibrato.

A more technical representation of dynamics in music is the *velocity* as standardized in the MIDI protocol [3]. It consists of 128 volume steps with the highest representing maximal intensity. But although the number of velocity steps is standardized, there is no standardized relationship between MIDI velocity and intensity: Goebl and Bresin found that for the Yamaha Disklavier and the Bösendorfer SE System (both pianos allowing for the monitoring of performance data), the relationship between MIDI velocity and (logarithmic) sound pressure level is nearly linear when disregarding very low and high values [85]. Dannenberg investigated the RMS peak level of various synthesizers and software instruments and found great differences among different synthesizers [86]. He identified a general trend for the velocity to be related to the square root of the RMS peak instead of its logarithm. Using one single electronic instrument, Taguti measured the A-weighted sound pressure level dependent on velocity and key [87]. The results, displayed over various keys for different input velocities, showed non-systematic deviations of up to 10 dB from a constant level among keys.

4.3 Features

The features presented in this section can be roughly structured into three categories: physical measures of sound intensity, approaches to measure intensity in recording studio environments, and psycho-acoustically motivated features modeling the human perception of loudness.

4.3.1 Root Mean Square

The *RMS* is one of the most common intensity features and is sometimes directly referred to as the sound intensity. It is calculated from a block of audio samples by

$$v_{\mathrm{RMS}}(n) = \sqrt{\frac{1}{\mathcal{K}} \sum_{i=i_s(n)}^{i_e(n)} x(i)^2}. \tag{4.4}$$

Table 4.1 RMS for the three prototypical signal types *silence* (zero magnitude at all samples), *white noise* with a rectangular PDF and a peak amplitude A, and a sinusoidal signal with the same peak amplitude

Input Signal	v_{RMS}
silence	0
rect. white noise (ampl. A)	$A/\sqrt{3}$
sinusoidal (ampl. A)	$A/\sqrt{2}$

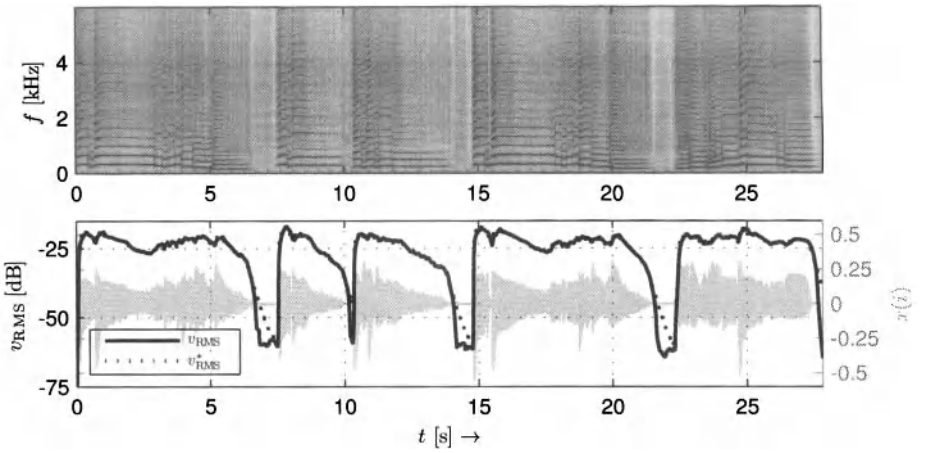

Figure 4.2 Spectrogram (top), waveform (bottom background), and RMS (bottom foreground) of a saxophone signal

Typical block lengths for the RMS calculation are in the range of several hundred milliseconds. The length in seconds is the so-called *integration time*.

The result of the calculation is a value within the range $0 \leq v_{\mathrm{RMS}}(n) \leq 1$ (as long as the amplitude 1 represents full scale. It will equal 0 if the input is silence and will approach 1 for both a square wave with maximum amplitude and a constant DC offset at ± 1. Sharp transients in the signal will be smoothed out by an RMS measure due to the comparably long integration time. Table 4.1 shows the RMS results for three signal prototypes.

Figure 4.2 shows the RMS of an example signal for two implementations, one calculated as shown above and the other with an approximation explained below. The RMS is a measure of the power of the signal; the two implementations shown are roughly equivalent except during sudden signal pauses in which the low-pass filtered variant only slowly decreases.

4.3.1.1 Common Variants

The calculation of the RMS can be computationally inefficient for large block lengths and small hop sizes $\mathcal{H}$. If the hop size is one sample, it is possible to reduce the number of computations by using the method of recursive implementation introduced in the context of the MA filter in Sect. 2.2.1.1:

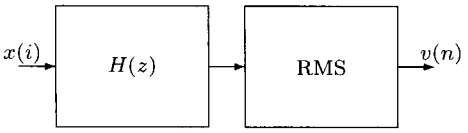

Figure 4.3 Flowchart of the frequency-weighted RMS calculation

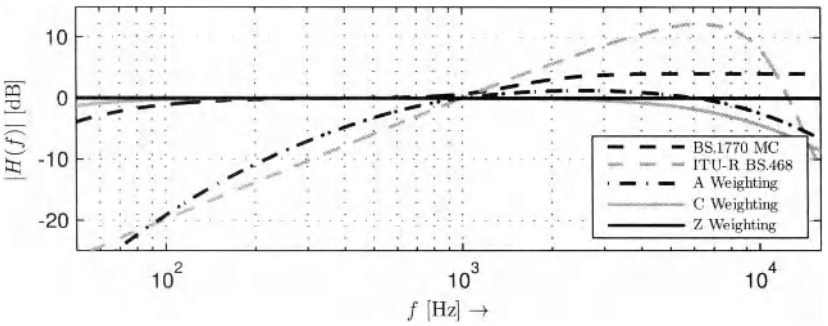

Figure 4.4 Frequency weighting transfer functions applied before RMS measurement

$$v_{\mathrm{RMS}}^2(n) = \frac{x(i_{\mathrm{e}}(n))^2 - x(i_{\mathrm{s}}(n-1))^2}{i_{\mathrm{e}}(n) - i_{\mathrm{s}}(n) + 1} + v_{\mathrm{RMS}}^2(n-1), \tag{4.5}$$

$$v_{\mathrm{RMS}}(n) = \sqrt{v_{\mathrm{RMS}}^2(n)}. \tag{4.6}$$

This implementation is computationally efficient but still requires a significant amount of memory to be allocated for large block lengths. An approximation of the RMS v_{RMS}^* with hop size $\mathcal{H} = 1$ can be implemented with a single-pole filter (compare Sect. 2.2.1.1):

$$v_{\mathrm{tmp}}(i) = \alpha \cdot v_{\mathrm{tmp}}(i-1) + (1-\alpha) \cdot x(i)^2 \tag{4.7}$$

$$v_{\mathrm{RMS}}^*(i) = \sqrt{v_{\mathrm{tmp}}(i)}. \tag{4.8}$$

The filter coefficient α can be estimated from the integration time (or block length) with Eq. (2.29).

The RMS computation is often preceded by a weighting filter. Figure 4.3 shows typical transfer functions for such weighting filters. The transfer function of a weighting filter is usually modeled after an inverse *equal-loudness contour* which measures the level for which a listener perceives equal loudness at different frequencies [80]. Thus, the weighting filter amplifies frequency regions in which the human ear is sensitive and attenuates other regions.

The specific weighting filter transfer functions shown in Fig. 4.4 are:

- *A weighting*: weighting function to be used for sounds at low level [88],
- *C weighting*: weighting function to be used for sounds at medium level [88],

- *Z weighting*: flat frequency weighting function (no weighting) [88],
- *RLB weighting*: weighting function according to ITU-R BS.1770 (plus high-frequency emphasis for multichannel signals) [89], and
- *CCIR weighting*: weighting function according to ITU-R BS.468 [90].

When frequency weighted RMS measures are used as models of loudness, the integration time is usually several seconds in order to ignore short-term variations of the signal.

4.3.2 Peak Envelope

The *peak envelope* of an audio signal can be extracted in different ways. The simplest way of extracting the envelope is to find the absolute maximum per block of audio samples:

$$v_{\text{Peak}}(n) = \max_{i_s(n) \le i \le i_e(n)} |x(i)|. \tag{4.9}$$

Using a so-called *Peak Program Meter (PPM)*, the envelope is extracted on a sample-per-sample basis. The PPM, frequently used in recording studio environments, operates with different integration times for attack (*attack time*) and release (*release time*). Typically, the attack time is significantly shorter than the release time (e.g., attack time: 10 ms, release time: 1500 ms) which means that the output reflects an increase in level faster than a decrease. This originates in requirements of recording engineers: the attack time has to be short in order to allow the systems to detect short peaks while the longer release time gives humans more time to actually see those peaks.

The structure of a digital PPM as described by Zölzer is shown in Fig. 4.5 [91]. The filter coefficient representing the attack time is named α_{AT}. The release time coefficient α_{RT} is not always in use and is being represented by λ in the figure. More specifically, λ has the following two states depending on the input magnitude

$$\lambda = \begin{cases} \alpha_{\text{RT}}, & \text{if } |x(i)| \le v_{\text{PPM}}(i-1) \\ 0, & \text{otherwise} \end{cases}. \tag{4.10}$$

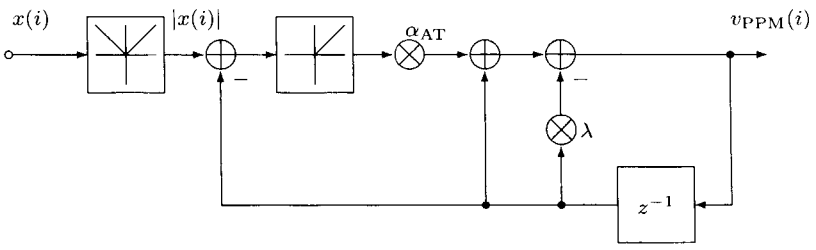

Figure 4.5 Flowchart of a Peak program meter

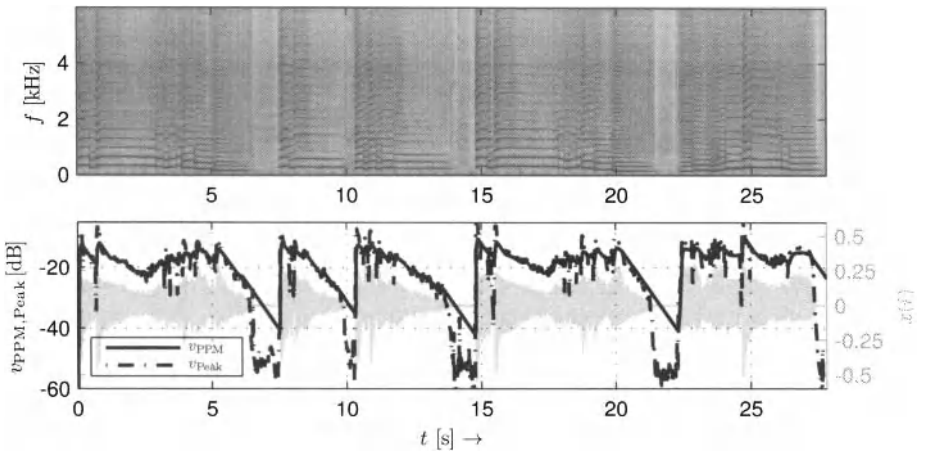

Figure 4.6 Spectrogram (top), waveform (bottom background), and PPM output compared to the magnitude maximum per block (bottom foreground) of a saxophone signal

The two states may be referred to as *release state* and *attack state*. The corresponding output equations are

- *Release state*:

$$\begin{aligned} v_{\mathrm{PPM}}(i) &= v_{\mathrm{PPM}}(i-1) - \alpha_{\mathrm{RT}} \cdot v_{\mathrm{PPM}}(i-1) \\ &= (1-\alpha_{\mathrm{RT}}) \cdot v_{\mathrm{PPM}}(i-1), \end{aligned} \tag{4.11}$$

- *Attack state*:

$$\begin{aligned} v_{\mathrm{PPM}}(i) &= \alpha_{\mathrm{AT}} \cdot \big(|x(i)| - v_{\mathrm{PPM}}(i-1)\big) + v_{\mathrm{PPM}}(i-1) \\ &= \alpha_{\mathrm{AT}} \cdot |x(i)| + (1-\alpha_{\mathrm{AT}}) \cdot v_{\mathrm{PPM}}(i-1). \end{aligned} \tag{4.12}$$

The result of both v_{Peak} and v_{PPM} is a value within the range $0 \leq v_{\mathrm{PPM}} \leq 1$. It will equal 0 if the input is silence and approach 1 for certain full-scale signals.

Figure 4.6 shows the results of both envelope measures. The comparison with the two RMS results as shown in Fig. 4.2 reveals similar behavior of RMS and peak measures as the calculation is relatively similar. The dotted peak maximum shows faster and more pronounced changes as can be clearly seen during the pauses with the PPM's constant decrease due to the long release time.

4.3.3 Psycho-Acoustic Loudness Features

There exist complex loudness measurements based on either psycho-acoustic properties of human loudness perception or physiological models of the human ear (or both). These are not frequently used for ACA, as (a) they are comparably costly to implement and to compute and (b) there are indications that they are in many cases equally meaningful as simpler loudness approximations such as the algorithm described in *International Telecommunication Union (ITU)* recommendation BS.1770, a weighted RMS solution [92].

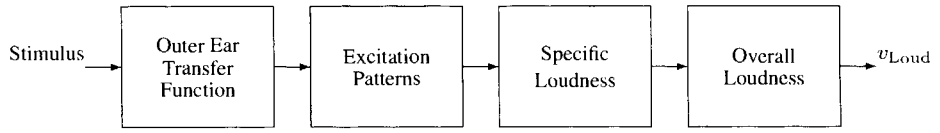

Figure 4.7 Flowchart of Zwicker's model for loudness computation

A widely known and psycho-acoustically motivated loudness measurement has been proposed by Zwicker [47, 93]. Figure 4.7 presents the flowchart of this loudness calculation. The signal is transformed into the bark domain (see Sect. 5.1) by a filterbank or a similar frequency transform. The so-called *excitation patterns* are computed from the filterbank outputs by taking into account frequency sensitivity and masking effects. Then, the *specific loudness* is calculated from the excitation patterns per band. The resulting loudness is the sum of the specific loudness over all bands.

4.3.3.1 EBU R128

One relatively recent recommendation for measurement of the loudness of a program or a file has been published by the *European Broadcasting Union (EBU)* [94]. The loudness itself is measured in compliance to ITU recommendation BS.1770, an RMS measure with an *RLB weighting*. The required block length is 3.0 s with a block overlap ratio of at least 66%. Blocks with very low loudness are discarded with a gating threshold.

The EBU recommendation also requires the following additional values to be extracted from the audio signal:

- clipped values according to an up-sampled true peak meter, and
- the loudness range computed from a histogram of all loudness block results as the difference $Q_v(0.95) - Q_v(0.10)$.

CHAPTER 5

TONAL ANALYSIS

Tonal aspects play an important role in understanding and analyzing music, as can be seen from the vast number of pitch-related publications in music theory. Pitches are the basic building blocks of key, melody, and harmony of a piece of music.

5.1 Human Perception of Pitch

The human perception of *pitch* is directly related to the frequency of a signal in a way that higher frequencies will lead to the perception of a higher pitch. If the signal is a combination of sinusoidal components with the frequencies $f_0, 2f_0, 3f_0, \ldots$ (which is a reasonable approximation for tonal sounds produced by many musical instruments), then the *fundamental frequency* f_0 dominates the pitch perception. As a matter of fact, humans will usually even perceive the same pitch for this combination of harmonics if the fundamental frequency f_0 has low power or is missing.

5.1.1 Pitch Scales

The relation between the fundamental frequency and the perceived pitch is non-linear; at higher frequencies, two pitches with the same perceived pitch distance will have a larger frequency distance than at lower frequencies. Simply speaking, the non-linearity is tied to the frequency resolution of the human cochlea. There are different approaches to measure this cochlear frequency map and thus there exist different ways of describing it. The

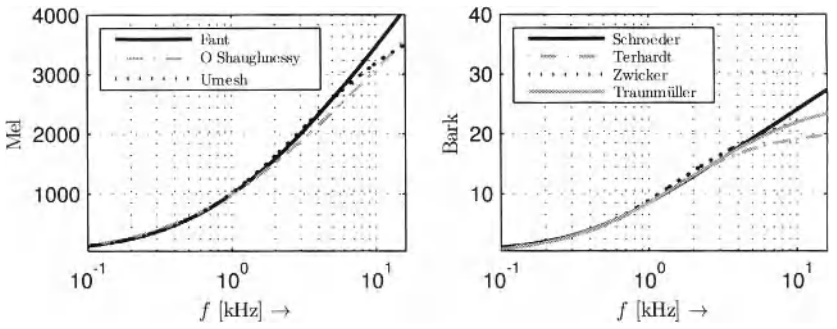

Figure 5.1 Different models for the non-linear mapping of frequency to mel (left) and bark (right)

most common models are the *mel scale* and the *critical band rate*, also called *bark scale* (Fig. 5.1). There has been a number of proposals for analytical functions to approximate the measurement data resulting from listening test for these scales. In most practical audio analysis applications, the use of one or another approximation is apparently circumstantial (compare [95, 96]). Nevertheless, several of these approximations will be presented below to illustrate the number of options.

5.1.1.1 Mel Scale

The term *mel* was introduced in 1937 by Stevens et al. as the name of a subjective pitch unit [82]. The *mel scale* is a measure of tone height. The empirical data used to build the numerous analytical models of the mel scale stems from only a limited number of psychological experiments: the most important test results have been presented by Stevens et al. [82], Stevens and Volkmann [97], and Siegel [98]. Three models will be presented here; the two older models by Fant [99]

$$\mathrm{m_F}(f) = 1000 \cdot \log_2 \left(1 + \frac{f}{1000\,\mathrm{Hz}}\right) \tag{5.1}$$

and O'Shaughnessy [100]

$$\mathrm{m_S}(f) = 2595 \cdot \log_{10} \left(1 + \frac{f}{700\,\mathrm{Hz}}\right) \tag{5.2}$$

are most commonly used. Note that the latter model is sometimes also referenced in the form

$$\mathrm{m_S}(f) = 1127 \cdot \log \left(1 + \frac{f}{700\,\mathrm{Hz}}\right). \tag{5.3}$$

The third model proposed by Umesh et al. [101] appears in this list mainly to demonstrate the variety of models and is not as widely known as the two other models:

$$\mathrm{m_U}(f) = \frac{f}{2.4 \cdot 10^{-4} f + 0.741}. \tag{5.4}$$

5.1.1.2 Bark Scale

The *bark scale* or *critical band rate* is constructed from the bandwidth of measured frequency groups, the critical bands. Zwicker and Fastl suggest that the bark scale is related to the mel scale by $1\,\mathrm{bark} = 100\,\mathrm{mel}$ [47].

The most prominent models of the bark scale have been proposed by Schroeder et al. [102]

$$\mathfrak{z}_{\mathrm{S}}(f) = 7 \cdot \operatorname{arcsinh}\left(\frac{f}{650\,\mathrm{Hz}}\right), \tag{5.5}$$

Terhardt [103]

$$\mathfrak{z}_{\mathrm{T}}(f) = 13.3 \cdot \arctan\left(0.75 \cdot \frac{f}{1000\,\mathrm{Hz}}\right), \tag{5.6}$$

and Zwicker and Terhardt [104]

$$\mathfrak{z}_{\mathrm{Z}}(f) = 13 \cdot \arctan\left(0.76 \cdot \frac{f}{1000\,\mathrm{Hz}}\right) + 3.5 \cdot \arctan\left(\frac{f}{7500\,\mathrm{Hz}}\right). \tag{5.7}$$

Traunmüller's model [105] is not as well known but is simple to calculate

$$\mathfrak{z}_{\mathrm{TM}}(f) = \frac{26.81}{1 + {}^{1960}/_{f}} - 0.53. \tag{5.8}$$

5.1.1.3 Other Models

Many more models have been proposed over the years for the non-linear transformation of frequency to perceptual frequency groups, pitch height, and position on the human *cochlea*.

Moore's model for the *Equivalent Rectangular Bandwidth (ERB)* can be seen as a model "competing" to the critical band rate [83]

$$\mathfrak{e}(f) = 9.26 \log\left(1 + \frac{f}{228.7}\right). \tag{5.9}$$

Terhardt introduced a function, which he named *SPINC*, as an alternative to the mel scale [106]:

$$\mathfrak{s}(f) = 1414 \arctan\left(\frac{f}{1414\,\mathrm{Hz}}\right), \tag{5.10}$$

and Greenwood proposed the following equation to compute the position (normed to the range of $[0; 1]$) of a specific frequency on the cochlea [107]:

$$\mathfrak{x}(f) = \frac{1}{2.1} \log_{10}\left(\frac{f}{165.4} + 1\right). \tag{5.11}$$

5.1.2 Chroma Perception

There is an additional facet to human pitch perception: not only do we perceive pitch height from low to high but we tend to group pitches with specific frequency ratios [108, 109]. More specifically, humans perceive frequencies with a frequency ratio of a power of 2 (such as $f_0, 2f_0, 4f_0, 8f_0, \ldots$) as very similar and closely related to each other. This phenomenon is usually called *chroma perception*.

Figure 5.2 visualizes this in a helix plot. On the one hand, the frequency is monotonically increasing on the z axis, modeling the tone or pitch height. On the other hand, points with the same (x, y) coordinates share a frequency ratio of a power of 2 — they share the same scale degree and are in the same pitch class (see Sect. 5.2.1), respectively. The circle which appears when looking directly on the (x, y) plane would thus encompass all pitch classes.

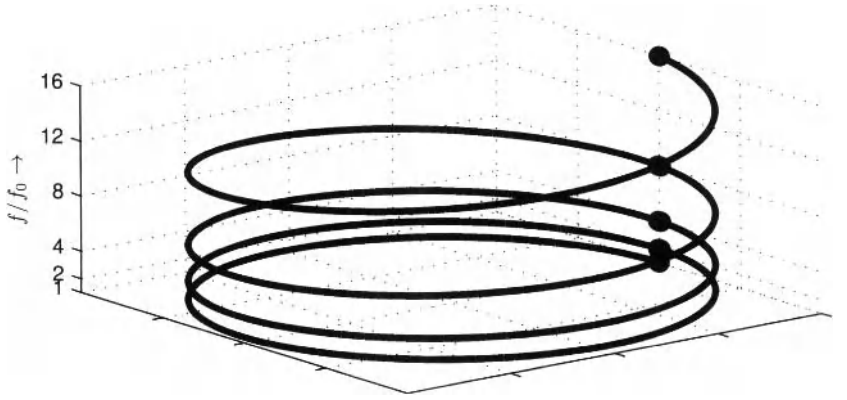

Figure 5.2 Helix visualizing the two facets of pitch perception: pitch height and chroma

5.2 Representation of Pitch in Music

Much can be said about pitch-related properties of music, covering not only the frequency mapping of specific pitches but also interaction of pitches in chords and melodies, harmony progression, and musical key. It is not the intention of this section to give a comprehensive overview on the (music) theory involved; instead, some basics will be covered in a simplified manner since the understanding of some theoretical background can be of help in the successful design of analysis algorithms.

5.2.1 Pitch Classes and Names

The concept of musical pitch in western music theory closely follows the pitch helix (see Fig. 5.2) in that each *octave*, i.e., each range with boundaries with a frequency ratio of $2:1$, is divided into the same chunks: the 12 *pitch classes*.

The common labels of these octave-independent pitch classes are shown in Table 5.1.

Table 5.1 Pitch class indices and corresponding names of the chromatic pitch classes

0	1	2	3	4	5	6	7	8	9	10	11
C	$C\sharp/D\flat$	D	$D\sharp/E\flat$	E	F	$F\sharp/G\flat$	G	$G\sharp/A\flat$	A	$A\sharp/B\flat$	B

Seven of those pitch classes form the so-called *diatonic scale*.

Table 5.2 shows a diatonic scale, more specifically the major mode (see Sect. 5.2.3) based on a root note C with pitch class index, pitch class name, Solfège name and distance to the previous pitch class in semi-tones.

As can be seen from the table, the distance between two neighboring pitches is in most cases two semi-tones; twice, however, it is only one semi-tone. Any pitch between the presented diatonic pitches can be constructed be raising the pitch (e.g., $C \rightarrow C\sharp$) or lowering it (e.g., $E \rightarrow E\flat$). For now it will be assumed that, e.g., $C\sharp$ equals $D\flat$ (a relation that is

Table 5.2 Names and distance in semi-tones ΔST of diatonic pitch classes

Index	*Name*	*Solfège Name*	ΔST
0	C	Do	1
2	D	Re	2
4	E	Mi	2
5	F	Fa	1
7	G	Sol	2
9	A	La	2
11	B	Si	2

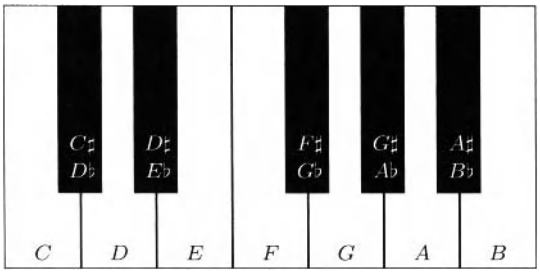

Figure 5.3 One octave on a piano keyboard with annotated pitch class names

referred to as *enharmonic equivalence*), resulting in 12 pitch classes per octave as shown in Table 5.1.

Figure 5.3 depicts these pitch classes on a piano keyboard; the white keys form the mode $C\ Major$ mode as shown in Table 5.2. Figure 5.4 displays the pitch classes in one octave in musical score notation.

A common convention for naming musical pitches used in the following is simply the pitch class name followed by an octave index, e.g., $C2$ or $A4$. Each new octave starts with a C by convention.

5.2.2 Intervals

The distance between two pitches is the musical *interval*. It is used for both the distance of simultaneously sounding pitches and pitches sounding one after another. Table 5.3 names commonly used intervals and their corresponding distance in semi-tones.

Figure 5.5 displays the most important intervals (rising from pitch $C4$) in musical score notation.

Humans will hear the same interval for different pairs of pitches if the ratio between their fundamental frequencies is the same.

5.2.3 Root Note, Mode, and Key

The musical *key* of a tonal piece of music is defined by both its *mode* and a *root note*.

The root note is the most important pitch class in a specific key. It is also referred to as the *first scale degree* and will usually appear most frequently in a piece of music.

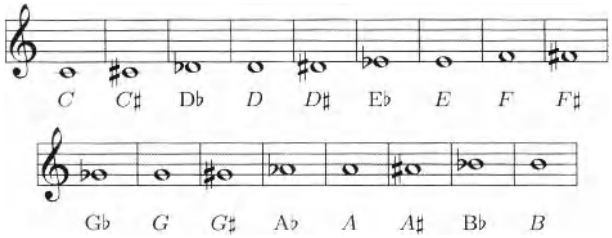

Figure 5.4 Musical pitches $C4 \ldots B4$ in musical score notation; enharmonically equivalent pitches are displayed twice

Table 5.3 Names of musical intervals, their enharmonic equivalents, and their pitch distance in semi-tones

Interval	*Enharmonic Equivalent*	ΔST
Unison	Diminished Second	0
Minor Second	Augmented Unison	1
(Major) Second	Diminished Third	2
Minor Third	Augmented Second	3
Major Third	Diminished Fourth	4
(Perfect) Fourth	Augmented Third	5
Augmented Fourth	Diminished Fifth/Tritone	6
(Perfect) Fifth	Diminished Sixth	7
Minor Sixth	Augmented Fifth	8
Major Sixth	Diminished Seventh	9
Minor Seventh	Augmented Sixth	10
Major Seventh	Diminished Octave	11
(Perfect) Octave	Augmented Seventh	12

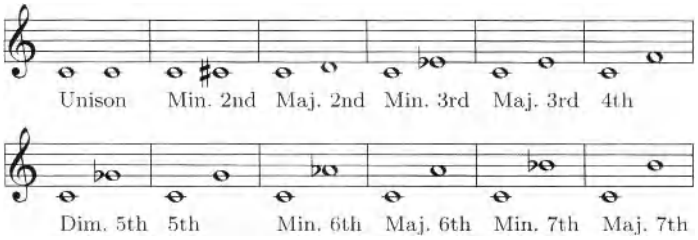

Figure 5.5 Musical intervals in musical score notation

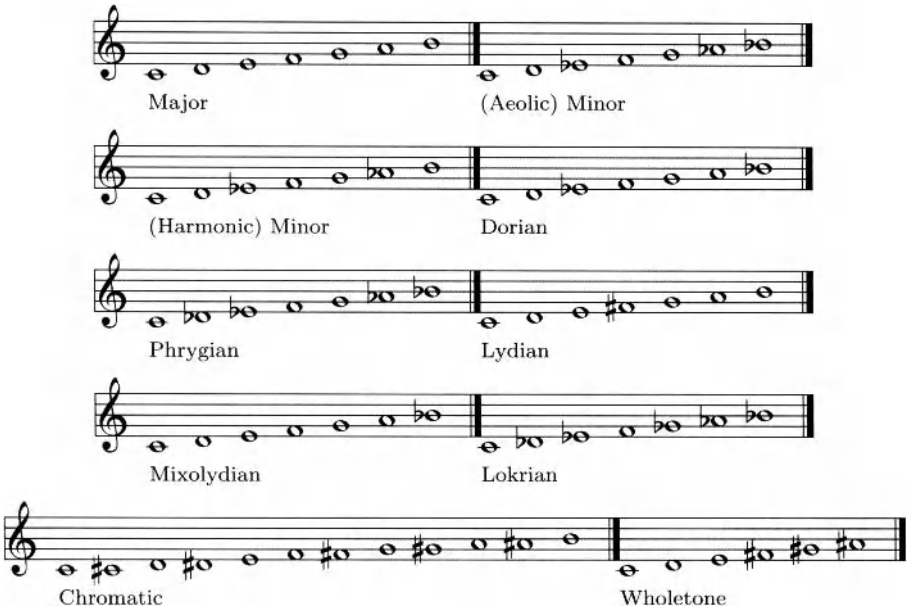

Figure 5.6 Different modes in musical score notation starting at the root note C

The *mode* defines a set of relative pitch relationships; an example would be: the distance between first and second scale degree is a major second, between first and third scale degree is a major third, etc. The most common modes are the *major mode* and the *minor mode*. Figure 5.6 displays an example set of different modes, all starting from the root note C. All modes except major mode and the two minor modes — aeolic mode and harmonic mode — are only of interest in specific musical styles and are usually not very important in the context of ACA.

The key thus defines the set of pitch classes which are used to construct the tonal aspects of a piece of music. In popular music it is common for a piece to have exactly one key. There are many exceptions to this rule: the key can change within a piece (when a so-called *modulation* occurs) and non-key pitches may be used for musical reasons.

Depending on the current root note and mode, up to six different accidentals have to be used to raise or lower the pitches in the musical score. The notational convention allows writing all key-inherent accidentals at the begin of a staff — the *key signature* — and to add only accidentals within the score where non-key-inherent pitches are used. Figure 5.7 shows major modes with their key signatures starting from all possible root notes.

As can be seen from Fig. 5.7, the root notes of keys that differ only in one accidental are always spaced by a fifth (F/C, C/G and G/D, etc.). This relationship can be visualized by the so-called *circle of fifths* (Fig. 5.8). Strictly speaking, this circle is only closed in the case of enharmonic equivalence when $F\sharp$ equals $G\flat$ (see Sect. 5.2.5.2).

Neighboring keys on the circle of fifths have all pitch classes but one in common; for example, $G\ Major$ has the same pitches as $C\ Major$ except for the F which is raised to an $F\sharp$ in $G\ Major$.

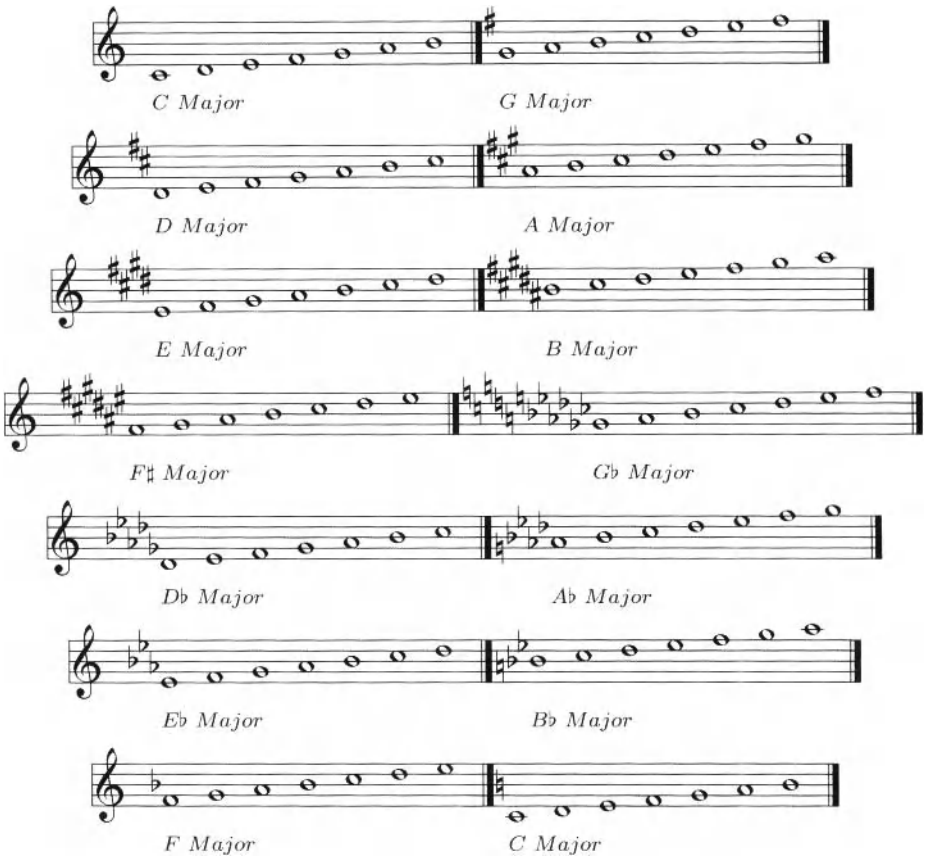

Figure 5.7 The twelve major keys in musical score notation, notated in the 4th octave

The circle of fifths also exists for the (aeolian) minor keys with $a\ minor$ being the key without accidentals.

Since the circle of fifths shows the relation of different keys, it hints also at what modulations are more or less likely. To give an example, the most likely key changes from $F\ Major$ would be either $C\ Major$, $B\flat\ Major$, or $d\ minor$. The circle of fifths can thus be understood as a model for a distance map between keys.

Keys construed from the same set of pitch classes such as $C\ Major$ and $a\ minor$ are called parallel keys; in the circle of fifths their distance is 0. In order to build an analytical model for key distances with a non-zero distance between parallel keys, the circle of fifths can be enhanced to a three-dimensional model. This model would feature two parallel planes, one containing the circle for major keys and the other for minor keys. Parallel keys thus share the same (x, y) coordinates but have a different z coordinate (which is then the distance between the two parallel keys).

Approaches to automatic key detection from audio signals can be found in Sect. 5.5.

5.2.4 Chords and Harmony

The simultaneous use of (usually no less than three) different pitches creates a chord. Chords built of three pitches are referred to as triads. The most common chord types are

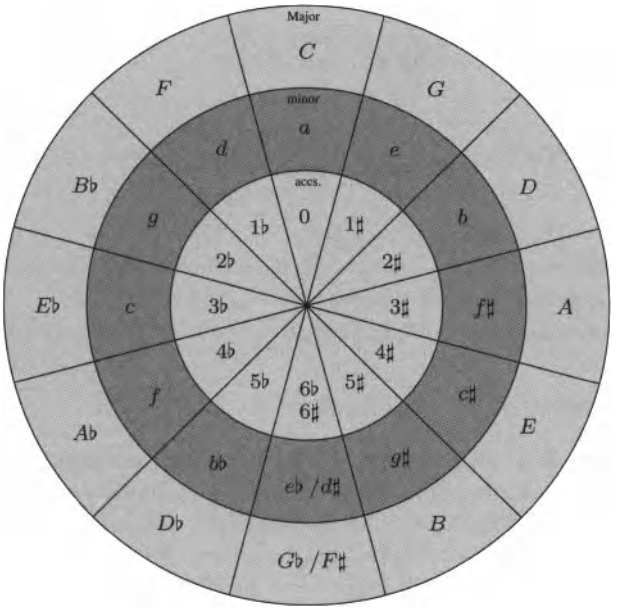

Figure 5.8 Circle of fifths for both major keys and minor keys, plus the number of accidentals of the key signature per key

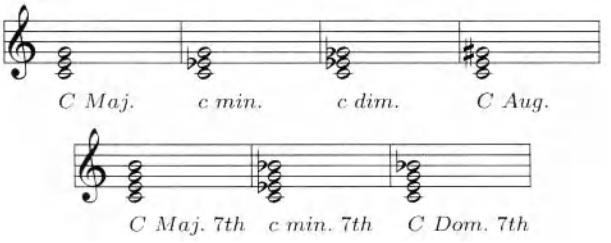

Figure 5.9 Common chords in musical score notation on a root note of C

built of third intervals — a few examples are displayed in Fig. 5.9 with respect to a root note of C. Note that the same naming convention is used to label both the key and some common chords (e.g., $C\ Major$), so one has to derive from the context which of the two is meant.

Chord-inherent pitches can be doubled in different octaves without changing the chord type. The chord's root note is the most commonly doubled pitch. If the lowest note of a sounding chord is not its root note it is called *inverted*. The first chord inversion of a $C\ Major$ triad would therefore have the pitch classes (bottom-up) $E-G-C$. Figure 5.10 shows the two possible inversions of a $D\ Major$ triad. The second inversion usually appears less frequent than the first.

Each chord can have one or more musical (harmonic) functions in a key and dependent on the musical context. The chord sharing the root note with the current key is referred to as the *tonic* and will most likely represent the tonal center. Other important harmonic

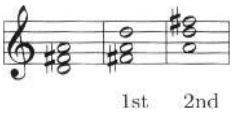

Figure 5.10 The two inversions of a *D Major* triad in musical score notation

functions are the so-called *dominant* with the key's fifth scale degree as the root note and the *subdominant* with the key's fourth scale degree as the root note. Dominant chords will usually induce an expectancy of a following tonic in the listener.

Approaches to automatic chord recognition from audio signals can be found in Sect. 5.6.

5.2.5 The Frequency of Musical Pitch

The most systematic model for relating musical pitch to frequency and vice versa is using the so-called equal temperament which also results in enharmonic equivalence (other temperaments will be mentioned in Sect. 5.2.5.2). The best example for such a transformation is the MIDI scale in which each semi-tone has a distance of 1 to its nearest neighbor. The equations for the transformation from frequency f to MIDI pitch $\mathfrak{p}$ and vice versa are

$$\mathfrak{p}(f) = 69 + 12 \cdot \log_2\left(\frac{f}{f_{A4}}\right), \tag{5.12}$$

$$f(\mathfrak{p}) = f_{A4} \cdot 2^{\frac{\mathfrak{p}-69}{12}}. \tag{5.13}$$

The reference frequency f_{A4} is the tuning frequency (see Sect. 5.2.5.1); using 12 times the logarithm to the base 2 ensures that each octave is divided into 12 parts of equal "length" and the constant 69 results in the pitch $A4$ having the index 69, a convention of the MIDI standard [3]. The MIDI pitch can be mapped easily to the pitch class index PC as introduced above by using a modulo operation:

$$\mathrm{PC}(\mathfrak{p}) = \mod(\mathfrak{p}, 12). \tag{5.14}$$

The unit cent $\Delta C(f_1, f_2)$ is a distance measure between two pitches or frequencies f_1 and f_2. It can be computed by

$$\begin{aligned}\Delta C(f_1, f_2) &= 100 \cdot (\mathfrak{p}(f_1) - \mathfrak{p}(f_2)) \\ &= 100 \cdot \left(\left(69 + 12 \cdot \log_2\left(\frac{f_1}{f_{A4}}\right)\right) - \left(69 + 12 \cdot \log_2\left(\frac{f_2}{f_{A4}}\right)\right)\right) \\ &= 1200 \cdot \log_2\left(\frac{f_1}{f_2}\right).\end{aligned} \tag{5.15}$$

A semi-tone interval has thus a distance of 100 cents and an octave has a distance of 1200 cents.

5.2.5.1 Tuning Frequency

The *tuning frequency* f_{A4} is the frequency of the *concert pitch* $A4$ (also: *standard pitch*) which is used for tuning one or more musical instruments. The tuning frequency is standardized internationally to 440 Hz [110], but the exact frequency used by musicians can

Table 5.4 Typical range of deviation of the tuning frequency from 440 Hz over three centuries

Year	*Lower Deviation*	*Upper Deviation*
1750	−50 Hz	+30 Hz
1850	−20 Hz	+20 Hz
1950	−5 Hz	+10 Hz

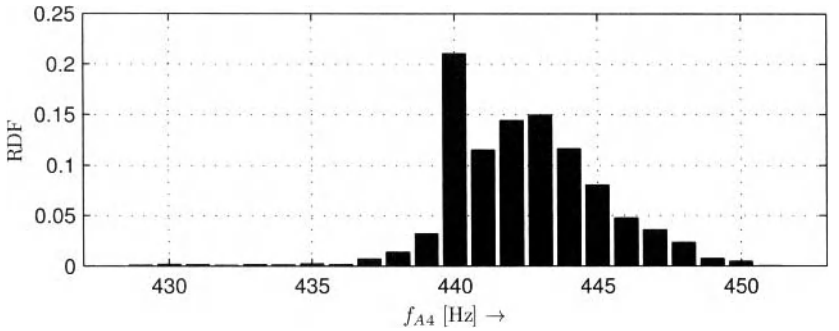

Figure 5.11 Distribution of tuning frequencies in Lerch's data set

vary due to various reasons such as the use of historic instruments or timbre preferences. Two performances of the same piece of music using different tuning frequencies will differ in their average pitch height.

The range of typical tuning frequencies decreased over the centuries. Table 5.4 shows this range for the past three centuries as deviation from 440 Hz [111].

Nowadays, while for many electronic music productions the "default" tuning frequency of 440 Hz is used, the tuning frequencies of orchestras still deviate from this standard tuning frequency. For example, the Chicago Symphony Orchestra and the New York Philharmonic tune at 442 Hz, while the Berliner Philharmoniker and the Wiener Philharmoniker have a tuning frequency of 443 Hz.[1] At least in the case of both European orchestras, the tuning frequency was higher in previous decades. The frequencies 442 and 443 Hz correspond to deviations of 7.85 and 11.76 cents from the standard tuning frequency, respectively.

There exist two studies analyzing the tuning frequency of a data set of recordings. Zhu et al. processed a database of 60 popular and 12 classical pieces[2] and found only three pieces of this database with a deviation of approximately 2–4 cents from the standard tuning frequency [112]. Lerch presented the results of a study with a large database of classical music, consisting of more than 3000 tracks and an overall playing time of approximately 291 hours [113]. A histogram of the extracted tuning frequencies as displayed in Fig. 5.11 shows a maximum at a tuning frequency of 440 Hz; the maximum itself consists of about 21% of the test database. The distribution has an arithmetic mean value of 442.38 Hz and a standard deviation of 2.75 Hz. The majority of the results (95%) is in the range from 439 to 448 Hz and only 50% of the results have a tuning frequency in the range of 440–443 Hz. The percentage of files below 439 Hz is about 3.3%.

[1] According to the orchestra's archivists, March and April 2006.

[2] The term *classical music* is in this context understood as "non-popular" music, as opposed to the epoch itself.

Table 5.5 Deviations of the Pythagorean, meantone, and two diatonic temperaments from the equally tempered scale in cents at a reference pitch of C (after Briner [111])

Pitch Class	*Equally*	*Pythagorean*	*Meantone*	*Diatonic Major*	*Diatonic Minor*
C	0	0	0	0	0
$C^{\#}$	0	–	–	–	–
D	0	+3.9	−6.9	+3.9	+3.9
$E^{\flat}$	0	–	–	–	+15.6
E	0	+7.8	−13.7	−13.7	–
F	0	−2.0	+3.4	−2.0	−2.0
$F^{\#}$	0	–	–	–	–
G	0	+2.0	−3.5	+2.0	+2.0
$A^{\flat}$	0	–	–	–	+13.7
A	0	+5.9	−10.2	−15.6	–
$B^{\flat}$	0	–	–	–	+17.6
B	0	+9.8	−17.1	−11.7	–

The tuning frequency is not necessarily static once the instruments have been tuned; it may change during a concert or a recording session. On the one hand the tuning frequency could be slowly decreasing as it sometimes happens at *a cappella* performances, on the other hand the tuning frequency may slightly increase, for example, due to a rising involvement of the musicians during the concert. The maximum range of this deviation can be assumed to be small in the case of professional musicians (about 3–5 cents).

Approaches to estimate the tuning frequency of a music signal can be found in Sect. 5.4.

5.2.5.2 Temperament

The temperament defines a system of frequency ratios for intervals.

In the equally tempered case assumed above the frequency ratio between the mid-frequencies of two pitches f_1 and f_2 spaced by N semi-tones (with N being a negative or positive integer) is always

$$\frac{f_1}{f_2} = 2^{N/12}. \tag{5.16}$$

Therefore, the distance between two pitches is constant and independent of tonal and harmonic context; it stays also constant if the enharmonic equivalents of the two pitches are used: the interval $B, F\sharp$ is the same as the intervals $B, G\flat$ or $C\flat, G\flat$. This, however, is only true for the equal temperament.

The Pythagorean, meantone and diatonic temperaments are examples of other temperaments in which the pitches are tuned depending on the key. Basically, the Pythagorean temperament is constructed from perfect fifths (frequency ratio 3 : 2), the meantone temperament is constructed with nearly perfect thirds, and the diatonic temperament intends to use as small frequency ratios as possible toward the tonic for every scale degree.

Table 5.5 shows the deviations in cents of different temperaments from the equal temperament.

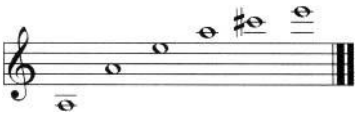

Figure 5.12 Six harmonics of the fundamental pitch $A3$ in musical score notation

5.2.5.3 Intonation

In contrast to temperament, the frequency of a certain pitch may vary over time depending on the musical context. This deviation from the frequency grid set by the temperament is called (expressive) intonation and is part of the musical performance.

Obviously, only musicians who are not forced to a pre-defined temperament can use expressive intonation. Examples are vocalists as well as players of string, brass, or woodwind instruments, while, for example, piano players have no means of changing the pitch frequency during a performance. It is generally assumed that musicians tend to produce pure frequency relationships such as $f_1/f_2 = 3/2$ for a fifth (seven semi-tones) because it sounds more "natural" [114]. Furthermore, if a specific note *leads* musically to the following, the frequency will in many cases be adjusted toward the following pitch. A good example are leading tones where the seventh scale degree leads to the first scale degree (in $C\ Major$: $B \rightarrow C$).

A special performance phenomenon related to expressive intonation is *vibrato*. Vibrato, a musical (performance) ornament, is a periodic frequency modulation of the pitch around its mean frequency. The extent and frequency of a vibrato is somewhat instrument dependent; typical frequencies are in the range of 5–9 Hz and the amplitude may be as large as two semi-tones [45, 115, 116].

5.3 Fundamental Frequency Detection

The basic assumption for all approaches to the estimation of the fundamental frequency (also referred to as *pitch detection* or *pitch tracking*) is that the signal is periodic or quasi-periodic. The periodic state of an acoustic tone can be represented in a Fourier series as introduced in Sect. 2.1.1, meaning that it is a superposition of weighted sinusoidals. The frequency of these sinusoidals is an integer multiple of the lowest — the *fundamental* — frequency. The different frequency components of a tone are called *harmonics* or *partials*, with the first harmonic being the *fundamental frequency*. Higher harmonics are also called *overtones*. The first six harmonics of the musical pitch $A3$ are displayed in Fig. 5.12 in traditional musical score notation.

This frequency structure can be found for most signals generated by acoustic or electronic instruments which are perceived as pitched sounds. However, there are certain deviations from this rule. For example, humans will hear the fundamental frequency of a harmonic series even without this frequency actually being present in the signal. Although an absent fundamental frequency occurs only rarely, it happens frequently that the first harmonic has lower energy than one or more of the higher harmonics.

The piano and string instruments are examples of acoustic signals in which the harmonics are not placed precisely at integer frequency multiples of the fundamental frequency but slightly off. A model of this deviation or inharmonicity is

$$f_k = k f_0 \sqrt{1 + \lambda(k^2 - 1)} \tag{5.17}$$

with k being the index of the harmonic (in this case better called partial as it is not entirely harmonic) and λ being the inharmonicity factor with typical values in the range $[10^{-3}; 10^{-4}]$ [117]. There are also instruments which are perceived as being pitched but show no clean harmonic pattern. Examples for this class of instruments are the xylophone, the vibraphone, and timpani. Nevertheless, the assumption of quasi-periodic states of a music signal has in many cases been proven to be valid and successful for *fundamental frequency detection*.

The common range of fundamental frequencies for musical instruments roughly starts between 20 and 50 Hz (e.g., on the double bass) and ends between 3–5 kHz (e.g., on the piccolo). Depending on the instrument, a number of at least three to seven harmonics should be considered to be important components of the sound (if components other than the fundamental frequency are of interest).

5.3.1 Detection Accuracy

Algorithms for fundamental frequency detection work either in the time domain by estimating the period length of the fundamental or in the frequency domain by finding the frequency of the fundamental. Both are discrete value domains and have thus a maximum accuracy determined by the distance of two time domain samples and two frequency domain bins, respectively. There are work-arounds for virtually enhancing the resolution such as frequency reassignment (see Sect. 2.2.3.1), but for now the effects of this discretization on the accuracy of fundamental frequency detection will be investigated.

5.3.1.1 Time Domain

As already mentioned above, the estimated period length of the fundamental frequency is quantized to samples in the time domain, resulting in the estimated period length being a multiple of the distance between two samples

$$T_{\mathrm{Q}} = j \cdot T_{\mathrm{S}} \tag{5.18}$$

with the integer multiplier j. This quantization leads to a certain amount of error depending on the sample rate and the period length. Figure 5.13 shows the minimum detection error in cent in dependency of the fundamental frequency for two different sample rates. The absolute worst-case error is small for low frequencies and increases with the frequency. Higher sample rates will result in smaller errors.

5.3.1.2 Frequency Domain

The trade-off between time resolution and frequency resolution is one of the big issues in STFT-based frequency analysis. While long analysis blocks increase the frequency resolution, they require a periodic and stationary signal during the analysis block in order to be useful. As can be easily seen from Eq. (2.44), the frequency resolution will stay constant if the ratio of sample rate f_{S} and block length K stays constant:

$$f_{\mathrm{Q}} = k \cdot \frac{f_{\mathrm{S}}}{K}. \tag{5.19}$$

Table 5.6 displays the frequency resolution for different STFT sizes.

Figure 5.14 visualizes the error in cents for an analysis block length of 2048 samples at the sample rates 44.1 and 96 kHz. As the error is measured in cents (logarithmic scale) it decreases with increasing frequency (linear scale).

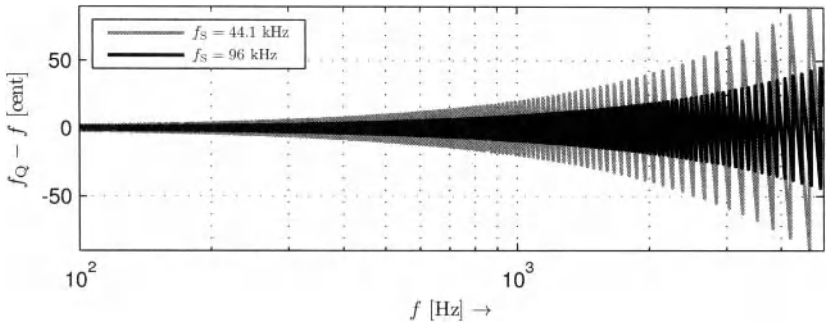

Figure 5.13 Detection error in cents resulting from quantization of the period length to a length in samples for two different sample rates

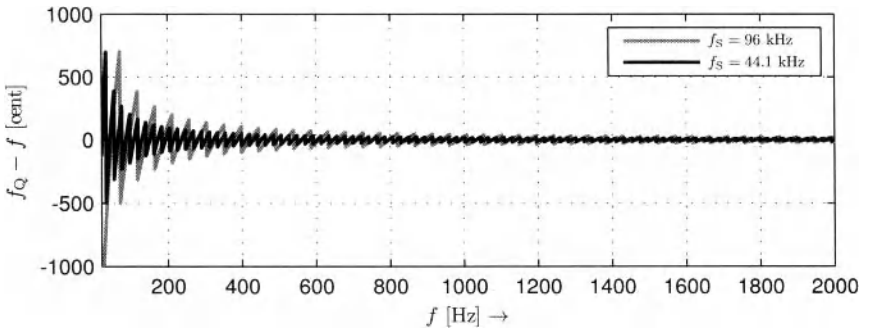

Figure 5.14 Detection error in cents resulting from quantization of the fundamental frequency to the STFT bin at an analysis block length of 2048 samples for two different sample rates

Table 5.6 Frequency resolution of the STFT for different block lengths at a sample rate of 48 kHz with bin index and frequency of the first bin k_{ST} with a distance to the following bin smaller than half a semi-tone

$\mathcal{K}$	Δf [Hz]	k_{ST}	$f(k_{ST})$ [Hz]
256	187.5	35	6562.5
512	93.75	35	3281.25
1024	46.875	35	1640.625
2048	23.4375	35	820.3125
4096	11.7188	35	410.1563
8192	5.8594	35	205.0781
16384	2.9297	35	102.5391

5.3.2 Pre-Processing

As with nearly all signal processing algorithms, the results of fundamental frequency detection might be improved by applying appropriate pre-processing steps such as down-mixing, filtering, and sample rate conversion. Furthermore, it might be helpful in certain cases to remove noisy and non-tonal components before the actual detection.

5.3.2.1 Filtering and Down-Sampling

Since the range of fundamental frequencies is quite restricted compared to the sample rates used nowadays in audio signal processing, higher frequency components are frequently removed by both low-pass filtering and down-sampling the input signal. The cut-off frequency of the low-pass filter depends on the fundamental frequency range of the input signal as well as on the usefulness of higher harmonics for the subsequent detection algorithm. The target frequency of the resampling process might also depend on the required resolution for the period length estimation.

In addition, a high-pass filter removing components near DC may be helpful to clean up the input signal — typical cut-off frequencies would be in the range of 30–300 Hz.

5.3.2.2 Identification of Tonal Components

The differentiation of tonal and non-tonal (sinusoidal and noisy) signal components is a crucial pre-processing step for many audio signal processing systems and remained an active research topic through the last decades. The range of applications which can benefit from a reliable tonalness detector is wide: psycho-acoustic models in perceptual audio encoders can be optimized with a more accurate estimation of the signal-to-mask ratio, source separation algorithms may be improved by avoiding noise-like components, the quality of analysis/synthesis systems such as phase vocoders and audio restoration algorithms may increase by treating tonal and noisy parts differently, and the accuracy of pitch-based analysis systems such as key detection and chord recognition and ultimately music transcription systems can be enhanced.

As the variety and number of applications benefiting from the detection of tonalness suggest, there has been a plethora of publications dealing with the identification of tonal components.

Research on this topic began in the 1970s when spectral processing of digital audio became more and more common. In contrast to non-spectral approaches targeting a single monophonic source signal distorted by noise (see, e.g., [118]), the analysis of individual spectral bins allows the processing of multi-voiced input signals.

In the context of speech separation, Parsons identified peaks by finding local maxima in the magnitude spectrum — a first processing step that can be found in practically every publication dealing with the detection of tonal bins — and used the peak's symmetry, its proximity to the next peak as well as the continuity of the frequency bin's phase for detecting "peak overlaps," i.e., bins with supposedly two or more influencing sinusoidals [119]. Terhardt extended the concept of detecting the local maximum by expecting more distant bins (specifically the bins with a distance of 2 and 3) to be a certain level lower than the maximum itself [120]. Serra proposed a measure of "peakiness" of local maxima by comparing the bin magnitude with the surrounding local minima; he also discarded peaks outside of a pre-defined frequency and magnitude range [121].

An amplitude-based measure computing the correlation function between the magnitude spectrum and the shifted spectrum of the used window function has been presented by Peeters and Rodet [122] as well as Lagrange [123]. They also utilized a phase-derived

measure comparing the bin frequency of a peak with its reassigned (instantaneous) frequency.

In addition to a local maximum feature similar to Terhardt's, Every proposed a threshold-based feature computed from the low-pass filtered magnitude spectrum, discarding peaks below the threshold [124].

Röbel et al. presented a set of features to classify spectral peaks into being sinusoidal or non-sinusoidal [125]. These features included the deviation of the bin frequency and its reassigned frequency, the peak's energy location according to its group delay, as well as the bandwidth of a spectral peak.

All of the publications presented above make a binary decision for a spectral bin being tonal or not; Kulesza and Czyzewski proposed an algorithm which aims at estimating the likelihood of a bin's tonalness [126]. They refer to this as a scoring classifier. This non-binary decision makes the algorithm probably most similar to the one outlined below; they use a so-called peakiness feature similar to Serra's, a frequency stability criterion for detected peaks, and a phase-based frequency coherence over subsequent blocks of the STFT. Their approach combines several features and uses a combination of heuristics and both binary and non-binary features to compute the resulting likelihood.

A feature-based approach to tonalness detection is a systematic and extensible way of computing the likelihood of an individual spectral bin being tonal. One example of such an approach is outlined in the following. It is based on the following assumptions for the input signal:

- it is a time-variant mixture of tonal and non-tonal signals,
- it has an undefined number of (tonal) voices (i.e., it is polyphonic), and
- the spectral envelope of both tonal and non-tonal components is unknown.

Furthermore, an individual tonal component is assumed to be

- salient, i.e., it is not masked by nearby components and has a certain intensity,
- deterministic, i.e., its phase cannot change erratically between the points of observation, and
- stationary for at least a minimum length of time.

These expected properties should be described by a set of features in the spectral domain. Each feature by itself should be simple to compute as well as simple to understand; it focuses on one individual property or aspect of a tonal component. Each feature $v_j(k,n)$ is the input of a Gaussian function $\varphi(x)$. Its output will be referred to as the *specific tonalness* $\Lambda_j(k,n) \in [0;1]$ which is a measure of likelihood of the bin k in frame n being a tonal component with respect to feature j

$$\Lambda_j(k,n) = \varphi\left(v_j(k,n)\right) = \exp\left(-\epsilon_j \cdot v_j(k,n)^2\right) \tag{5.20}$$

with ϵ_i being the normalization constant. The specific tonalness $\Lambda_j(k,n)$ is weighted and then combined to result in the *overall tonalness*.

Since the feature output $v_j(k,n)$ is in turn the input of the Gaussian function presented in Eq. (5.20), the features have to result in 0 output for tonalness and maximum output for non-tonalness. Examples of possible features are:

- *Local maximum*: Declaring the local maxima to be candidates for being tonal is a rather self-evident step in spectral analysis. In the presented variation the M surrounding bins are inspected for their magnitude being lower than the magnitude at the bin of interest:

$$v_1(k,n) = \sum_{m=1}^{M} \left(1 - \frac{m-1}{M}\right) \cdot \big(\Delta(k,m,n) + \Delta(k,-m,n)\big) \tag{5.21}$$

with

$$\Delta(k,m,n) = \begin{cases} 1, & \text{if } \big(|X(k,n)| - |X(k+m,n)|\big) \leq 0 \\ 0, & \text{if } \big(|X(k,n)| - |X(k+m,n)|\big) > 0. \end{cases} \tag{5.22}$$

The weighting increases the influence of nearby frequency bins on the likelihood.

- *Peakiness*: While the *local maximum* feature only takes into account whether the bin of interest has a higher magnitude than the neighboring bins, the *peakiness* evaluates their magnitude differences by relating the magnitude at the center bin with the mean of the neighboring magnitudes:

$$v_2(k,n) = \frac{|X(k-1,n)| + |X(k+1,n)|}{2 \cdot |X(k,n)|}. \tag{5.23}$$

The more pronounced a peak is, the lower will the feature value be and the higher will its tonalness be. The result $v_2(k,n)$ also depends on the used STFT windowing function and the spread of its main lobe. Because of these windowing effects, it could also be of advantage not to use the direct neighbors but bins with a bin distance of two or more for the averaging function, or to use the closest local minima as proposed by Serra [121].

- *Thresholding*: Since the tonal components are expected to be salient and to have more energy than noisy components, a magnitude threshold can be applied to increase the likelihood of bins with magnitudes above the threshold and decrease the likelihood of remaining bins correspondingly. This can be done by computing the ratio of a threshold $G(k,n)$ and the spectral magnitude $|X(k,n)|$:

$$v_3(k,n) = \frac{G(k,n)}{|X(k,n)|}. \tag{5.24}$$

The threshold can be determined by taking the maximum of an absolute threshold, a threshold relative to the highest magnitude in the current frame, and an adaptive threshold computed from the smoothed magnitude spectrum $X_{\mathrm{LP}}(k,n)$:

$$G(k,n) = \max\big(\lambda_1, \lambda_2 \cdot \max_{\forall k}(|X(k,n)|), \lambda_3 \cdot X_{\mathrm{LP}}(k,n)\big) \tag{5.25}$$

with λ_j being user-defined weighting parameters.

The smoothed magnitude spectrum X_{LP} may be computed with a single-pole filter:

$$X_{\mathrm{LP}}(k,n) = \alpha \cdot X_{\mathrm{LP}}(k-1,n) + (1-\alpha) \cdot |X(k,n)| \tag{5.26}$$

which is applied over the frequency in both the forward and the backward direction to ensure zero-phase response (see Sect. 2.2.1.2).

This thresholding process can be interpreted in different ways. On the one hand, one could see it as a pre-whitening process as used, for example, in pitch tracking systems with the goal of removing the spectral envelope from the signal [28]; on the other hand, it can be interpreted as a rudimentary model of a perceptual masking threshold applied to detect unmasked bins.

- *Frequency Coherence*: While the features presented above were based on the magnitude spectrum, the phase spectrum can also provide information on the tonalness of a frequency bin. More specifically, the instantaneous (or reassigned) frequency $f_I(k,n)$ (see Sect. 2.2.3.1) can be derived from the phase difference of overlapping spectra [17]. The instantaneous frequency has to be close to the bin frequency $f(k)$ in case of the main lobe of a stationary sinusoidal, otherwise, i.e., in the case of a noisy signal or a side lobe, the instantaneous frequency will probably deviate from the bin frequency. Furthermore, the instantaneous frequency should be comparably constant over consecutive blocks so that results can be averaged over two or more blocks. The final feature is then the difference of bin frequency and the (weighted) average of the instantaneous frequencies at this bin:

$$v_4(k,n) = f(k) - \frac{1}{\sum\limits_{\forall n_\mathcal{O}} b_{n_\mathcal{O}}} \sum_{n_\mathcal{O}=0}^{\mathcal{O}-1} b_{n_\mathcal{O}} \cdot f_I(k, n - n_\mathcal{O}). \tag{5.27}$$

The *overall tonalness* $\Lambda(k,n)$ is the weighted combination of each specific tonalness. It can be computed by the weighted arithmetic mean of the specific tonalness, an approach that shows similarities to a simplified *Radial Basis Function (RBF)* network [127],

$$\Lambda_A(k,n) = \frac{1}{\lambda_s} \sum_{j=1}^{4} \lambda_j \cdot \Lambda_j(k,n)\ , \qquad \lambda_s = \sum_{\forall j} \lambda_j, \tag{5.28}$$

or alternatively, when understood as a conditional probability, computed by the multiplication of the specific tonalness

$$\Lambda_G(k,n) = \prod_{j=1}^{4} \Lambda_j(k,n)^{\lambda_j/\lambda_s}\ , \qquad \lambda_s = \sum_{\forall j} \lambda_j. \tag{5.29}$$

5.3.3 Monophonic Input Signals

A monophonic signal as opposed to a polyphonic signal[3] is single-voiced. There will never be more than one fundamental frequency present at a time. The problem of detecting the fundamental frequency in monophonic signals is basically limited to detecting the longest periodicity period as the frequencies of the harmonics will be integer multiples of the fundamental frequency. Since many algorithms for monophonic input signals make heavy use of this property, they are of no or only limited use in the context of polyphonic input signals which are mixtures of multiple voices with possibly different and time-variant timbre.

[3]The term *polyphonic* will be used for signals with multiple voices. It will not be used in the more music-theory-related restricted definition of quasi-independent voices as opposed to homophonic, where the multiple voices move together in harmony.

5.3.3.1 Zero Crossing Rate

The *zero crossing rate* has already been introduced in Sect. 3.4.3. There are two ways to estimate the fundamental period length with zero crossings; the first is to relate the number of zero crossings in an analysis block to its length, an approach that will only produce acceptable results for very large block sizes:

$$T_0(n) = \frac{2 \cdot \big(i_e(n) - i_s(n)\big)}{f_S \cdot \sum_{i=i_s(n)}^{i_e(n)} |\mathrm{sign}\,[x(i)] - \mathrm{sign}\,[x(i-1)]|}, \tag{5.30}$$

and the second is to measure the interval $\Delta t_{ZC}(j)$ between neighboring zero crossings. This interval relates directly to the fundamental period length. If $\mathcal{Z}$ zero crossings have been detected in the analysis block, the fundamental period length can be estimated with

$$T_0(n) = \frac{2}{\mathcal{Z}-1} \sum_{z=0}^{\mathcal{Z}-2} \Delta t_{ZC}(z). \tag{5.31}$$

In case of large block lengths the time intervals can also be sorted into a histogram. The estimated period would then be the location of the histogram maximum multiplied by a factor of 2.

A related approach is to investigate the distance not only between zero crossing but between local extrema such as the maximum and minimum to increase robustness as explained by Rabiner and Schafer [128].

5.3.3.2 Autocorrelation Function

The use of the normalized ACF (see Sect.2.2.6) is very common in fundamental period length estimation. The lag of the maximum value is a direct estimation of the fundamental period length. Certain restrictions to maximum detection as exemplified in Sect. 3.4.1.4 can be applied to increase the algorithm's reliability.

Pre-Processing: Center Clipped Autocorrelation Function

The robustness of the detection can sometimes be improved by using so-called *center clipping* [128]. The non-linear function χ as shown in Fig. 5.15 (left) is applied to the input signal $x(i)$

$$x_c(i) = \chi\big(x(i)\big). \tag{5.32}$$

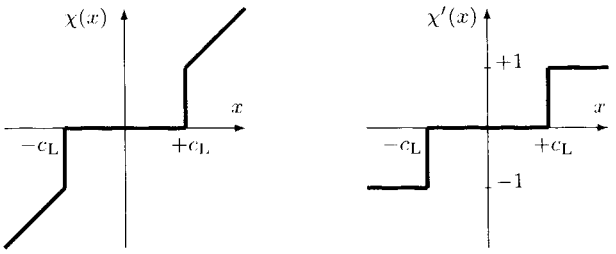

Figure 5.15 Non-linear pre-processing for ACF-based pitch period estimation: standard center clipping (left) and 3-level center clipping (right)

The idea of center clipping is to yield a result with more easily identifiable peaks. Typical thresholds c_L are chosen between 30% and 60% of the (instantaneous) maximum amplitude.

An alternative 3-level center clipping function χ' is shown in Fig. 5.15 (right). This function allows a very efficient computation of the ACF because the input signal then only has the three possible amplitudes $-1, 0, +1$. This performance optimization is usually not necessary anymore on modern hardware.

ACF Pre-Processing: Pre-Whitening

If the analyzed signal can be assumed to originate from a pulse-like excitation signal filtered with a transfer function, a common assumption in speech signal processing, a reasonable approach is to reverse the filtering process by estimating the smoothed spectral envelope of the current analysis block and apply the inverse filter to the signal — *pre-whitening*. The goal of the inverse filtering process is to convert the signal back to the initial pulse train in order to improve the results of the following correlation analysis.

There exist numerous ways to estimate the spectral envelope, including a low-order linear predictive filter (see Sect. 2.2.7), a smoothed power spectrum, or cepstrum-based iterative methods [129].

5.3.3.3 Average Magnitude Difference Function

The *Average Magnitude Difference Function (AMDF)* is similar to the ACF but avoids the use of multiplications and is therefore computationally efficient. The AMDF is computed by [130]

$$\mathrm{AMDF}_{xx}(\eta, n) = \frac{1}{i_e(n) - i_s(n) + 1} \sum_{i=i_s(n)}^{i_e(n)-\eta} |x(i) - x(i+\eta)|. \tag{5.33}$$

The estimated fundamental period length is then chosen to be the lag of the overall minimum if its value is also smaller than a certain (signal adaptive) threshold. The popular pitch tracking algorithm YIN utilizes the AMDF [131].

5.3.3.4 AMDF-Weighted Autocorrelation Function

The AMDF can also be used to weight the ACF. There are indications that this weighting leads to quite robust results [132]:

$$r'_{xx}(\eta, n) = \frac{r_{xx}(\eta, n)}{\mathrm{AMDF}_{xx}(\eta, n) + 1}. \tag{5.34}$$

5.3.3.5 Harmonic Product Spectrum and Harmonic Sum Spectrum

The *Harmonic Product Spectrum (HPS)* is an efficient method for finding the periodic harmonic pattern of an acoustic tone in its stationary state [133, 134]. It is defined by

$$X_{\mathrm{HPS}}(k, n) = \prod_{j=1}^{\mathcal{O}} |X(j \cdot k, n)|^2. \tag{5.35}$$

The parameter $\mathcal{O}$ is the order of the HPS.

The idea of the HPS is that the compression of the frequency axis by integer factors j causes higher harmonics at multiples of the fundamental frequency to coincide at the bin

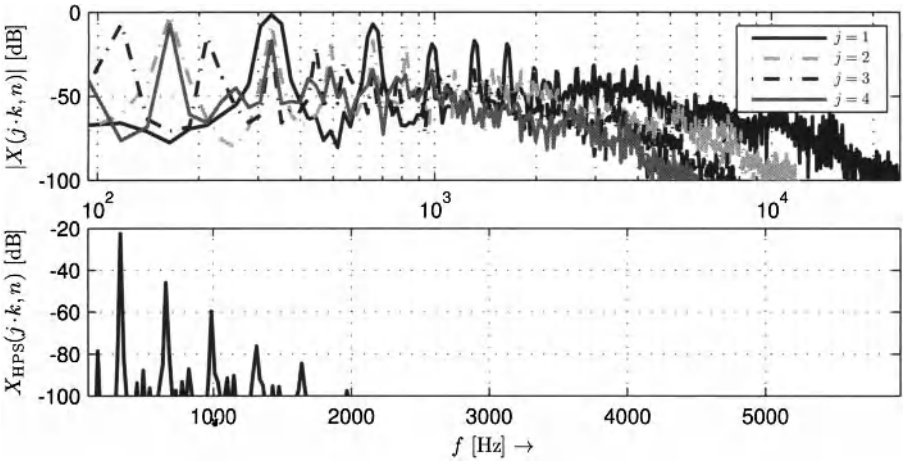

Figure 5.16 Compressed spectra with $j = 1, 2, 3, 4$ (top) and resulting HPS (bottom)

of the fundamental frequency. Thus, the first $\mathcal{O}$ harmonics will be mapped to their fundamental frequency. Since the harmonics can be assumed to have significantly higher power than any other signal components, the resulting (harmonic product) spectrum $X_{\mathrm{HPS}}(k, n)$ should have a clearly identifiable peak at the fundamental frequency. Figure 5.16 exemplifies this: the resulting peak in the HPS has a significantly higher distance to the second highest peak than in the original spectrum.

Alternative implementations of the HPS use the magnitude spectrum.

Typical Problems

If the fundamental frequency of the input signal is not located exactly on a frequency bin but between two bins, then the maxima of higher order harmonics will not be taken into account for the decimated spectra. The likelihood of missing the locations of the maxima increases with j. Two possible work-arounds can be used, but both will result in less efficient and more complicated implementations:

- increase the frequency resolution by using longer STFT block sizes or by interpolating (up-sampling) the spectrum, or
- take the maximum within a bin range for the multiplication. The bin range will have to increase by ± 1 bin with every increment of j.

If one of the harmonics is zero or near zero, the detection of the fundamental frequency will probably fail as the HPS at the fundamental frequency bin will be scaled with (near) zero. One approach to avoid this is to compute the *Harmonic Sum Spectrum (HSS)* with a definition similar to the HPS [134]:

$$X_{\mathrm{HSS}}(k, n) = \sum_{j=1}^{\mathcal{O}} |X(j \cdot k, n)|^2. \tag{5.36}$$

While the HSS is more robust against missing harmonics, the resulting maximum is usually not as pronounced as in the HPS.

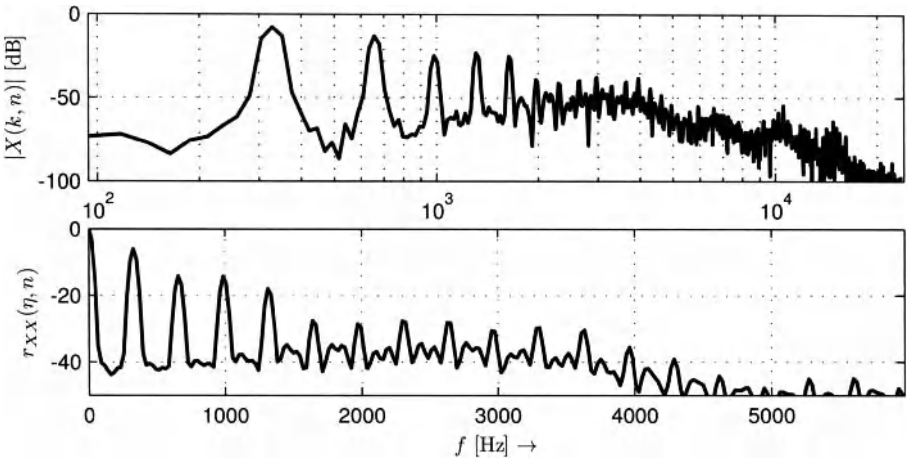

Figure 5.17 Magnitude spectrum (top) and ACF of this spectrum (bottom)

5.3.3.6 Autocorrelation Function of the Magnitude Spectrum

The usage of the ACF is a very intuitive approach to finding periodicities. Since the harmonics are equally spaced in the magnitude spectrum and the distance of neighboring harmonics equals the fundamental frequency, finding this periodicity is equivalent to finding the fundamental frequency. The lag of the maximum of the ACF is an estimate of the fundamental frequency in spectral bins. Figure 5.17 shows the result of the ACF of a magnitude spectrum.

5.3.3.7 Cepstral Pitch Detection

Cepstral pitch detection is based on the assumption that the analysis signal $x(i)$ is the result of a convolution of an excitation signal $e(i)$ with a transfer function $h(i)$

$$x(i) = e(i) * h(i). \tag{5.37}$$

This is a common assumption in speech signal processing; the excitation signal originates from the air streaming through the *glottis*. This excitation signal $e(i)$ consists of quasi-periodic pulses in the case of voiced (tonal) sounds [128]. The vocal tract and the nasal tract act as tubes that shape the frequency spectrum with the transfer function $h(i)$.

Equation (5.37) can be rephrased in the frequency domain (see Sect. B.1.3) as

$$X(\mathrm{j}\omega) = E(\mathrm{j}\omega) \cdot H(\mathrm{j}\omega). \tag{5.38}$$

If a (complex) logarithm is applied to this equation, the result is

$$\begin{aligned} \log\big(X(\mathrm{j}\omega)\big) &= \log\big(E(\mathrm{j}\omega) \cdot H(\mathrm{j}\omega)\big) \\ &= \log\big(E(\mathrm{j}\omega)\big) + \log\big(H(\mathrm{j}\omega)\big). \end{aligned} \tag{5.39}$$

Applying the logarithm allowed us to replace the multiplication with an addition. We define the *cepstrum* $c_x(i)$ to be the inverted logarithmic spectrum

$$\begin{aligned} c_x(i) &= \mathfrak{F}^{-1}\left\{\log\left(X(\mathrm{j}\omega)\right)\right\} \\ &= \mathfrak{F}^{-1}\left\{\log\left(E(\mathrm{j}\omega)\right) + \log\left(H(\mathrm{j}\omega)\right)\right\} \\ &= \mathfrak{F}^{-1}\left\{\log\left(E(\mathrm{j}\omega)\right)\right\} + \mathfrak{F}^{-1}\left\{\log\left(H(\mathrm{j}\omega)\right)\right\}. \end{aligned} \tag{5.40}$$

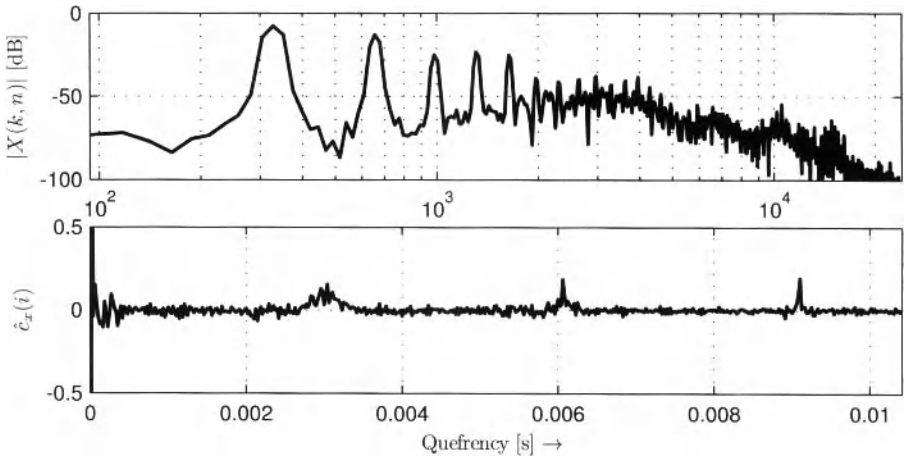

Figure 5.18 Magnitude spectrum (top) and cepstrum of this magnitude spectrum (bottom)

The inverse-transformed spectrum thus consists of signals $e(i)$ and $h(i)$ being *added* (although logarithmically) instead of being convolved. In order to emphasize the difference between the original time domain and the cepstral domain, the term *quefrency* is frequently used as axis label.

The cepstrum can be approximated by only using the magnitude spectrum

$$\hat{c}_x(i_\mathrm{s}(n)\ldots i_\mathrm{e}(n)) = \sum_{k=0}^{K/2-1} \log\left(|X(k,n)|\right) e^{\mathrm{j}ki\Delta\Omega} \tag{5.41}$$

and avoiding the use of the complex logarithm.

The cepstrum has two properties that are of particular interest in the context of pitch detection [133, 135]. First, a pulse-like excitation signal will also lead to a pulse in the cepstrum, and, second, the cepstrum will decay rapidly for large i. Therefore, the detection of a peak (or more accurately a pulse train) in the cepstrum should give the period length of the fundamental frequency. Figure 5.18 shows the cepstrum of an exemplary magnitude spectrum.

5.3.3.8 Auditory Motivated Pitch Tracking

A rather important class of pitch detection algorithms use models of human pitch perception to determine the pitch of a signal. Meddis and O'Mard describe the processing stages of such algorithms as [136]:

1. band-pass filtering,
2. HWR [see Eq. (3.33)] and band processing,
3. within-band periodicity extraction, and
4. across-band aggregation of periodicity estimates.

The filterbank used for band-pass filtering is frequently a gammatone filterbank as the "standard" filterbank for auditory processing. Gammtone filters have been introduced in Sect. 2.2.5.1.

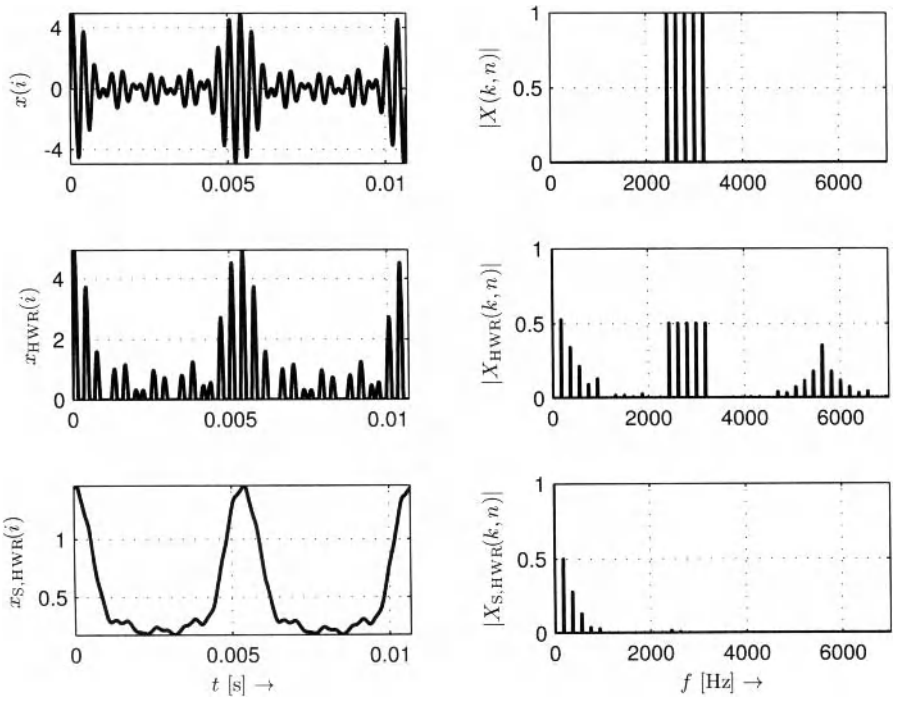

Figure 5.19 Time domain (left) and frequency domain (right) for a signal consisting of the 13th to 17th partial of a sound with a fundamental frequency of 187.5 Hz (top), the corresponding signal subjected to HWR (mid), and the low-pass filtered signal subjected to HWR (bottom)

Klapuri pointed out the importance of HWR on the filter channel outputs for fundamental pitch estimation [137]. Figure 5.19 shows this effect for one band containing the 13th–17th harmonics of a periodic sound: new frequency components are being generated at the distance of frequency components in the signal and thus around the fundamental frequency. Further band processing might include gain compression by, for instance, scaling the variance to unity [137].

A periodicity analysis of the filterbank outputs $z_c(i)$ may then be used for the detection of the fundamental period length by, for example, applying an ACF to each channel

$$r_{zz}(c, n, \eta) = \sum_{\eta=0}^{\mathcal{K}-1} z_c(i) \cdot z_c(i + \eta) \tag{5.42}$$

and summing the resulting ACFs

$$r_{\mathrm{A}}(n, \eta) = \sum_{c=0}^{\mathcal{C}-1} r_{zz}(c, n, \eta). \tag{5.43}$$

5.3.4 Polyphonic Input Signals

Most of the algorithms outlined above will not work well in the case of polyphonic signals with multiple simultaneous fundamental frequencies. The number of simultaneous pitches

in polyphonic signals depends on genre, epoch, and musical context. In most cases the number of independent voices will be between one and eight.

One of the first *multi-pitch detection* systems was presented by Chafe et al. in 1985 [138]. They pick spectral magnitude peaks in a multi-resolution FT and derive candidates for fundamental frequencies by grouping the peaks with respect to their frequency ratio. The detected candidates are then tracked in the spectrogram to discard spurious detections. Nowadays, multi-pitch detection is a lively research field with numerous methods and approaches of which only a few basic ones will be described below.

5.3.4.1 Iterative Subtraction

The principle of iterative subtraction is to apply a fundamental frequency detection algorithm for *monophonic* input signals to the signal to extract the predominant fundamental frequency, then find a way to subtract this and related (mostly harmonic) frequency components from the original signal and repeat the process on the residual until the criterion for termination has been reached.

An early reference to such an algorithm in the spectral domain has been published by Parsons [119] who — inspired by the work of Schroeder [133] — constructed a histogram of spectral peaks and their integer submultiples, chose the largest peak for the first fundamental frequency estimate, and removed this estimate and its multiples from the histogram to detect the second fundamental frequency. Klapuri et al. published two adaptations on the iterative subtraction procedure more recently. They use an auditory-motivated monophonic fundamental frequency detection algorithm to find the predominant, most salient fundamental frequency and estimate the spectrum of the voice with this frequency to subtract it from the original spectrum for the detection of additional voices [139, 140].

Cheveigné proposed a system for the tracking of multiple pitches in the time domain [141, 142]. First, the squared AMDF (compare Sect. 5.3.3.3) is computed with

$$\mathrm{ASMDF}_{xx}(\eta, n) = \frac{1}{i_\mathrm{e}(n) - i_\mathrm{s}(n) + 1} \sum_{i=i_\mathrm{s}(n)}^{i_\mathrm{e}(n)} \big(x(i) - x(i+\eta)\big)^2. \tag{5.44}$$

Then, the most salient period length is found at the lag of the ASMDF minimum:

$$\eta_\mathrm{min} = \eta \,\big|_{\min(\mathrm{ASMDF}_{xx}(\eta,n))}\,. \tag{5.45}$$

In order to remove the detected frequency and its harmonics from the signal, a (FIR) comb cancellation filter is applied to the signal with a delay corresponding to the lag of the detected minimum. The comb filter has the impulse response

$$h(i) = \delta(i) - \delta(i - \eta_\mathrm{min}) \tag{5.46}$$

and not only attenuates the detected fundamental frequency but also its harmonics at integer multiples. The process can then be repeated for the detection of a second fundamental frequency. It is also possible to implement this approach non-iteratively by using an exhaustive search. In this case, two cascaded cancellation filters can be applied to the signal in all possible combinations of η_1 and η_2. The most likely pair of fundamental period lengths is the combination of η_1 and η_2 which minimizes the overall output power of the output signals.

Meddis and Hewitt use an auditory approach similar to the one described in Sect. 5.3.3.8 for detecting one fundamental frequency and then use all remaining filter channels, i.e., filter channels not showing a peak at the detected frequency, to detect more fundamental frequencies [143]. The iteration process is terminated if more than 80% of the channels have been removed.

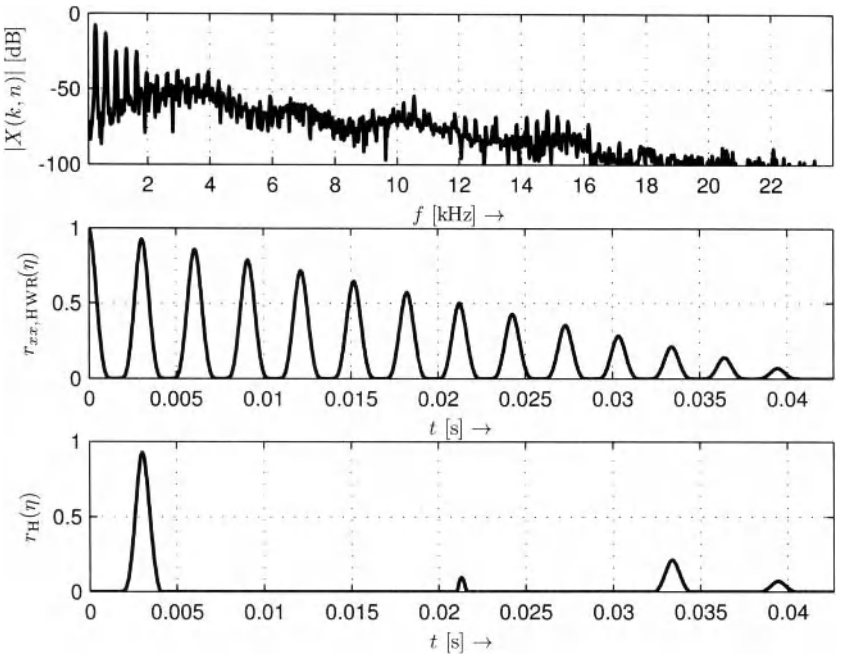

Figure 5.20 Magnitude spectrum (top), ACF of the signal subjected to HWR and harmonic ACF processing according to Karjalainen and Tolonen

5.3.4.2 Karjalainen and Tolonen

Another auditory-inspired multi-pitch detection algorithm focusing on computational efficiency has been published by Karjalainen and Tolonen [28, 144]. In a pre-processing step, they apply pre-whitening to flatten the spectral envelope. The spectral envelope is estimated by frequency-warped linear prediction. The next processing step splits the signal into a low-pass band and a high-pass band with a cut-off frequency of 1 kHz. The filter outputs are then subjected to HWR and smoothed with a low-pass filter. The periodicity within each band is estimated with an ACF. In an attempt to model auditory loudness scaling, the generalized ACF as introduced in Eq. (2.74) is used with an exponent of $\beta = 2/3$. The two resulting functions are then added to result in an overall summary ACF.

This summary ACF is then harmonically processed with an idea similar to HPS. The harmonic post-processing is an iterative process in which an interpolated, time-scaled and half-wave rectified version is subtracted from the half-wave rectified ACF $r(\eta, n) = r(j, \eta, n)$:

$$r(j,\eta,n) = \mathrm{HWR}\left(\mathrm{HWR}\left(r(j-1,\eta,n)\right) - \mathrm{HWR}\left(r\left(j,{}^{\eta}/_{j},n\right)\right)\right) \tag{5.47}$$

with $r\left(j,{}^{\eta}/_{j},n\right)$ generated with linear interpolation.

The repeated application of HWR in combination with scaling aims at discarding frequencies other the fundamental frequencies. Figure 5.20 shows the result of a "normal" ACF with HWR and an harmonic ACF of the same block.

5.3.4.3 Klapuri

Klapuri makes use of an auditory approach by computing the normalized filterbank outputs after HWR [145]. He then computes the STFT of each filter band, sums their magnitudes

$$Z(k,n) = \sum_{c=0}^{C-1} |Z_c(k,n)|, \tag{5.48}$$

and weights the resulting overall spectrum $Z(k,n)$ with a low-pass filter transfer function. To identify possible fundamental frequency candidates, a set of delta pulse templates is used representing every detectable fundamental frequency with its harmonics. Similar to the calculation of the HSS, these templates are multiplied with the spectrum, and the result can be used as an estimate for the salience of each fundamental frequency. The components of the most salient frequency are then being removed from the spectrum to find the next fundamental frequency in an iterative process.

5.3.4.4 Other Methods

Probabilistic approaches aim at modeling the pitch tracking problem by means of a statistical framework. To give only one example, Kameoka et al. model a tone in the frequency domain as a superposition of weighted Gaussian distributions at integer multiples of the fundamental frequency and its power envelope function in the time domain by overlapping Gaussian distributions [146]. The spectrogram is then composed into clusters which model individual notes.

Non-negative matrix factorization, introduced by Lee and Seung [147], has attracted noteworthy attention in the context of multi-pitch detection during the last decade. It decomposes a time-frequency representation into a matrix containing the spectra of the individual sounds and another matrix containing the information on when each of the individual spectra is active. Smaragdis and Brown applied the technique to the magnitude spectrogram in order to detect pitches with promising results [148].

5.4 Tuning Frequency Estimation

The computation of the tuning frequency (see Sect. 5.2.5.1) is a prerequisite for every mapping from frequency to musical pitch given in Eq. (5.12). Examples of applications utilizing this mapping are key detection, chord recognition, automatic transcription and melody finding. Many of the published algorithms are based on the assumption that a tuning frequency of 440 Hz has been used for the recording. This assumption works reasonably well in many cases, but in general the tuning frequency should be detected from the audio in order to improve detection accuracy [113]. This is particularly true when the tuning frequency can be expected to change over time.

To estimate the tuning frequency from an audio signal, Scheirer used a set of narrow band-pass filters with their mid-frequencies at particular bands that had been handpicked to match pitches from the previously analyzed score [149]. These filters are swept over a small frequency range such as a semi-tone. The estimated tuning frequency is then determined by the mid-frequency of the maximum of the sum of the energy of all filterbank outputs.

Dixon proposed to use a peak detection algorithm in the frequency domain and to calculate the instantaneous frequency of the detected peaks [150]. The equally tempered reference frequencies are then modified iteratively until the distance between detected and

reference frequencies is minimized. The amount of adaptation is the low-pass filtered geometric mean of previous and current reference frequency estimate.

Zhu et al. computed a CQT with the frequency spacing of 10 cents over a range of 7 octaves [112]. The detected peaks in the CQT spectrum are grouped based on the modulus distance against the concert pitch, resulting in a 10-dimensional histogram spanning 100 cents. If the maximum cumulative energy of the histogram is below a certain energy threshold, it is discarded. For the results of all processing blocks, a 10-dimensional average tuning pitch histogram is computed and the overall tuning frequency is chosen corresponding to the position of the histogram maximum.

In the context of single-voiced input signals, Ryynänen added the modulus distance of detected fundamental frequencies to a 10-dimensional histogram that is low-pass filtered over time [151]. Then, a "histogram mass center" is computed and the tuning frequency is adjusted according to this mass center.

Dressler and Streich modeled the tuning frequency deviation in cents with a circular model

$$z(n) = r_n \cdot \exp\left(\mathrm{j}\frac{2\pi}{100}\Delta C(f_n, 440\,\mathrm{Hz})\right), \tag{5.49}$$

$$\mu_z = \frac{1}{\mathcal{N}}\sum_{n=0}^{\mathcal{N}-1} z(n), \tag{5.50}$$

$$\hat{f}_{A4} = 2^{\frac{\arg(\mu_z)}{2\pi\cdot 12}} \cdot 440\ [\mathrm{Hz}], \tag{5.51}$$

and used different measures for r_n: the magnitude of salient spectral peaks, the magnitude of the estimated fundamental frequencies of the melody, and the magnitude of the average melody pitch, disregarding deviations such as vibrato [152]. As an alternative to computing the arithmetic mean, they propose constant adaption by low-pass filtering r_n with a single-pole filter.

Lerch proposed using a bank of steep resonance filters for detecting the tuning frequency, allowing both real-time processing and adaptation to a time-variant tuning frequency [113, 153]. In the range of two octaves, there are 24 groups of filters in (equally tempered) semi-tone distance, with each group consisting of 3 filters. The mid-frequencies of each group are narrowly spaced, and the mid-frequency of the centered filter is selected based on the most recent tuning frequency estimate. The filter output energy over a time window is then grouped based on the modulus distance against the concert pitch, resulting in a three-dimensional vector E. The symmetry of the distribution of the three accumulated energies gives an estimate on the deviation from the current tuning frequency compared to the assumption. If the distribution is symmetric, i.e., $E(0)$ equals $E(2)$, the assumption was correct. In the other case, all filter mid-frequencies are adjusted with the objective to symmetrize the energy distribution for the following processing blocks. The adaption rule used is the RPROP algorithm which allows fast and robust adaptation without the requirement of a signal-dependent adaption step size [154]. More specifically, the adaption rule for the adjustment of the estimated tuning frequency $\hat{f}_{A4}$ of the next processing block $n+1$ is

$$\hat{f}_{A4}(n+1) = \left(1 + \mathrm{sign}\left(E(2) - E(0)\right) \cdot \eta\right) \cdot \hat{f}_{A4}(n) \tag{5.52}$$

with η being scaled up if the direction of the previous adaptation was the same and scaled down otherwise. The advantage of the scaling is an increasing step size as long as the sign does not change and a decreasing step size otherwise. Figure 5.21 shows the adaptation

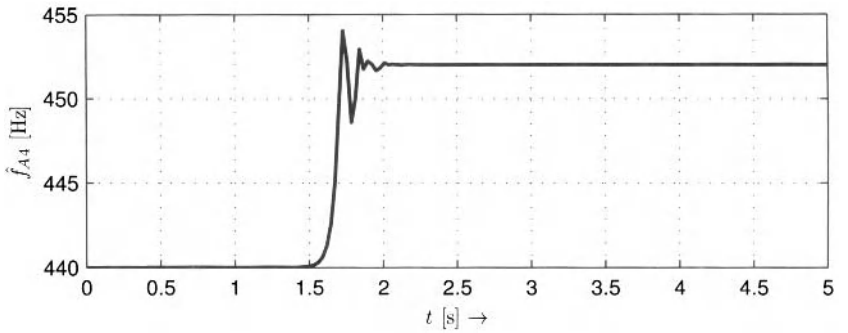

Figure 5.21 Adaptation of the tuning frequency estimate from an initial setting of 440 Hz to the target frequency of 452 Hz with the RPROP algorithm

from the initial tuning frequency of 440 Hz to the real frequency of 452 Hz. Adaptation can be tuned either for speed or for accuracy.

5.5 Key Detection

The musical key is an important tonal property of a piece of tonal music as it signifies the tonal center and restricts the pitch classes used within this key context. In classical music, the key is one descriptor used to identify a specific piece of music, complementing name, genre, and opus. In modern software applications for DJs the key is used to display the tonal compatibility between two tracks, i.e., to visualize how much "tonal overlap" they have to aid so-called harmonic mixing.

Since the key can be seen as a set of "allowed" pitch classes, the usual approach is to make use of an octave-independent representation of pitch for its automatic detection. The most common representation is the so-called pitch chroma.

5.5.1 Pitch Chroma

The *pitch chroma* (sometimes also referred to as *pitch chromagram* or *pitch class profile*) is a histogram-like 12-dimensional vector with each dimension representing one pitch class (C, $C\sharp$, D, ..., B; compare Sect. 5.2.1). It can be seen as a *pitch class distribution* for which the value of each dimension may represent both the number of occurrences of the specific pitch class in a time frame and its energy or velocity throughout the analysis block.

There are several advantages of using pitch chroma-based analysis. It is less dependent on timbre fluctuations and noise than many other feature. The pitch chroma is also robust against loudness fluctuations as well as against octave errors — a typical problem of pitch detection algorithms — with the self-evident disadvantage that the octave information is lost. It is, for example, not possible to distinguish between a note repetition an an octave interval with the pitch chroma representation.

While a representation of pitch similar to the pitch chroma has been frequently used in the past (see, e.g., Krumhansl's tonal distributions [155]), Bartsch and Wakefield were probably the first to propose its use in the context of audio signal processing [156]. Nowadays, this representation can be frequently found in publications in the context of ACA [157].

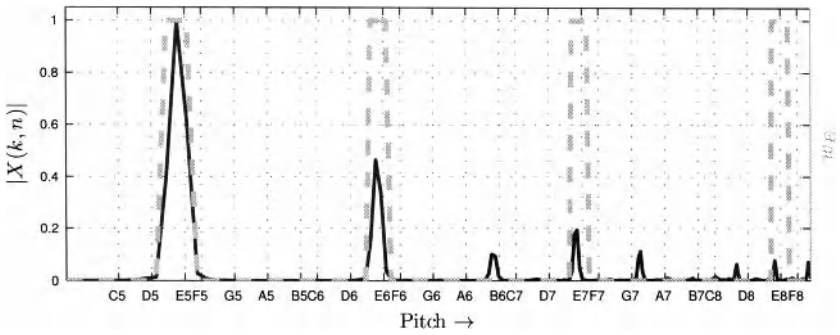

Figure 5.22 Window function for pitch class E for pitch chroma computation (octaves 5 . . . 8)

The exact algorithmic description of the pitch chroma computation varies from publication to publication; in all cases

- a frequency representation of the audio signal block is grouped into semi-tone bands,
- a measure of salience is computed in each band, and finally
- the sum of all bands (over all octaves) corresponding to a specific pitch class is calculated.

The simplest way to extract the pitch chroma sums the STFT magnitudes in each semi-tone band with the boundary indices $k_\mathrm{l}, k_\mathrm{u}$, and the result in every octave o is added to the corresponding pitch chroma entry with pitch class index j:

$$\nu(j,n) = \sum_{o=o_\mathrm{l}}^{o_\mathrm{u}} \left(\frac{1}{k_\mathrm{u}(o,j) - k_\mathrm{l}(o,j) + 1} \sum_{k=k_\mathrm{l}(o,j)}^{k_\mathrm{u}(o,j)} |X(k,n)| \right), \tag{5.53}$$

$$\boldsymbol{\nu}(n) = [\nu(0,n),\, \nu(1,n),\, \nu(2,n),\, \ldots,\, \nu(10,n),\, \nu(11,n)]^\mathrm{T}. \tag{5.54}$$

The indices $k_\mathrm{l}, k_\mathrm{u}$ are located at a distance of 50 cents from the mid-frequency of each equally tempered pitch.

Figure 5.22 shows the semi-tone bands for the pitch class E (pitch class index 4) in the octaves 5–8. Note that the window bandwidth is constant on the pitch axis; on the linear frequency axis it would increase with increasing frequency.

Frequently the pitch chroma is normed so that the sum of all possible pitch classes equals 1:

$$\boldsymbol{\nu}_\mathrm{N}(n) = \boldsymbol{\nu}(n) \cdot \frac{1}{\sum_{j=0}^{11} \nu(j,n)}. \tag{5.55}$$

Occasionally the pitch chroma is interpreted as a vector and is normed to a length of 1 instead:

$$\boldsymbol{\nu}_\mathrm{N}(n) = \boldsymbol{\nu}(n) \cdot \sqrt{\frac{1}{\sum_{j=0}^{11} \nu(j,n)^2}}. \tag{5.56}$$

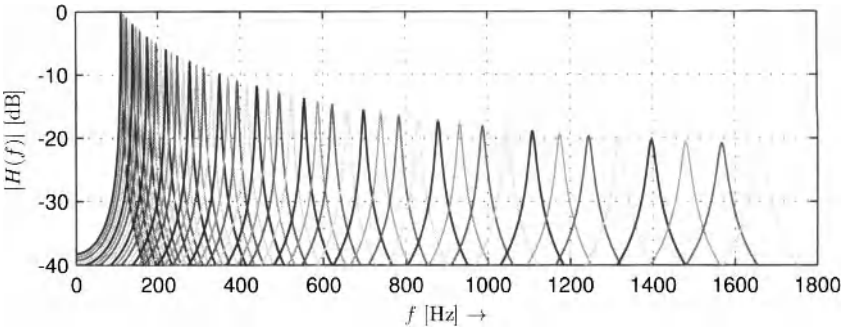

Figure 5.23 Frequency response of the resonance filterbank with one filter per semi-tone

5.5.1.1 Common Variants

The two main stages in the pitch chroma computation allowing alternative implementations are the type of the frequency transform and the selection of spectral content to sum.

When using an STFT as frequency transform, different windows with triangular, trapezoid, or sinusoidal shapes can be applied to each semi-tone band to weight bins in the center of the band higher than bins at the boundaries, as opposed to the rectangular window implicitly used in Eq. (5.53) and plotted in Fig. 5.22. Furthermore, the pitch chroma can be computed from a peak-picked or tonalness-weighted magnitude spectrum since only the tonal components are of interest for this feature.

A modification of the STFT has been used by Cremer and Derboven [158] for the computation of the pitch chroma: they utilize a so-called frequency-warped STFT as introduced by Oppenheim et al. which uses a chain of first-order all-pass filters to achieve non-equidistant spacing of the frequency bins [159]. The CQT as introduced in Sect. 2.2.4 can also be used for pitch chroma computation [160]. In this case the number of bands per semi-tone will equal 1 or be constant. It is also possible to use a filterbank with one filter for each semi-tone. Lerch used a bank of resonance filters as shown in Fig. 5.23 [153].

The analysis window length for the computation of one pitch chroma is another parameter allowing variations between different algorithms. While in the simplest case the STFT length equals the extraction window length [161, 162], other algorithms combine a fixed number of short analysis blocks [163] or adapt the extraction window length to the period between two neighboring beats [164] or to a measure of harmonic change in the audio [165]. The extraction window length obviously also depends on the task at hand; key detection requires comparably long windows as opposed to chord recognition which usually requires window lengths between a beat and a bar length.

5.5.1.2 Properties

In most audio applications, the pitch chroma is used as if only fundamental frequencies (and no overtones) have been mapped to it. In reality, all frequency content in the predefined frequency range of interest is mapped to the pitch chroma, regardless of it being a fundamental frequency or a higher harmonic. This leads to two possible problems:

- High non-power-of-two harmonics lead to distortions in the pitch chroma by adding undesired components. Figure 5.24 displays a series of harmonics with the fundamental pitch $A3$ and the resulting pitch chroma which shows spurious components at

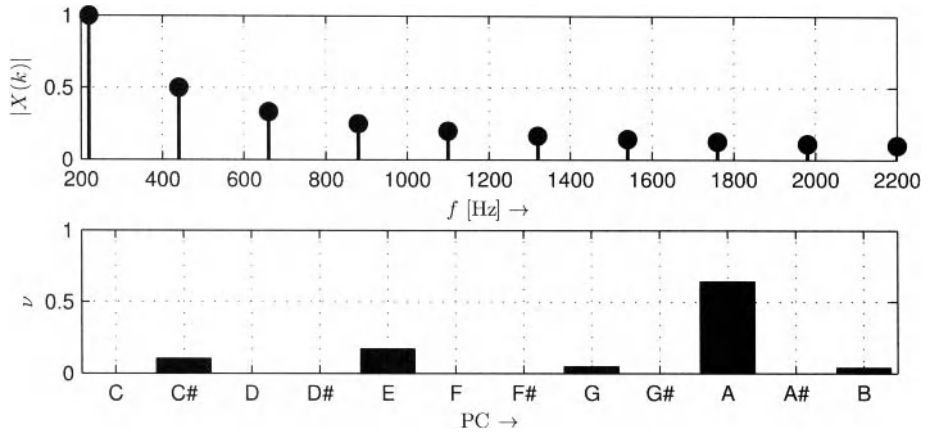

Figure 5.24 Pitch chroma of the pitch $A3$ with harmonics

Table 5.7 Deviation (in cents) of seven harmonics from the nearest equally tempered semi-tone mid-frequency

Harmonic	$\lvert\Delta C(f, f_T)\rvert$
$f = f_0$	0
$f = 2 \cdot f_0$	0
$f = 3 \cdot f_0$	1.955
$f = 4 \cdot f_0$	0
$f = 5 \cdot f_0$	13.6863
$f = 6 \cdot f_0$	1.955
$f = 7 \cdot f_0$	31.1741
$\mu_{\lvert\Delta C\rvert}$	6.9672

E and $C\sharp$ and to a lesser degree at G and B. It is possible to partly compensate for that effect by using an amplitude model for the harmonics and modifying the pitch chroma accordingly [160]. The effect can also be attenuated by applying a weighting function to de-emphasize higher frequencies (see e.g. [166]).

- High harmonics may even be able to distort the pitch chroma in a different way: since the harmonics will deviate from the equal temperament — the temperament the pitch chroma computation is based on — they may map to pitches not really part of the tonal context. This effect is, however, of limited influence since the deviation of the lower harmonics is relatively small as shown in Table 5.7.

The only possibility to avoid these artifacts is to use a multi-pitch detection system as a pre-processing step which ensures that only fundamental frequencies are mapped to the pitch chroma (compare [163, 167, 168]).

The pitch chroma entries may be ordered differently to reflect tonal relationships between the pitch classes. One way to do so is to replace the standard modulo operation for

Table 5.8 Pitch class order in the original and the rearranged pitch chroma

PC	C	$C\sharp$	D	$D\sharp$	E	F	$F\sharp$	G	$G\sharp$	A	$A\sharp$	B
PC_S	C	G	D	A	E	B	$F\sharp$	$C\sharp$	$G\sharp$	$D\sharp$	$A\sharp$	F

computing the pitch chroma index as given in Eq. (5.14) by a shifted version:

$$\mathrm{PC_S}(\mathfrak{p}) = \mod(7 \cdot \mathfrak{p}, 12). \tag{5.57}$$

This results in related keys with a distance of seven semi-tones to be close to each other. Table 5.8 shows the pitch class order of the rearranged chroma. Certain similarity measures such as the CCF will behave more gracefully or musically meaningful when using this resorted pitch chroma [112, 157].

5.5.1.3 Pitch Chroma Features

Similar to extracting instantaneous features from a spectrum, instantaneous features can be extracted from the 12-dimensional pitch chroma as well. The computation of statistical features such as the standard deviation is only of limited use due to the small number of elements. It should also be kept in mind that the pitch chroma is neither a series of observations nor a position in a space with 12 unrelated dimensions; it is a *distribution*.

There is no established set of features to be extracted from the pitch chroma, although the set of three features presented by Tzanetakis et al. proved to be simple yet effective [157]. They use both the index and the amplitude of the pitch chroma maximum as well as interval between its two highest bins. For the latter feature, the pitch chroma is rearranged in order to place related keys as neighbors (see above).

There exists a multitude of other features to extract from a pitch chroma that might be useful in certain applications. Possibilities include a pitch chroma crest factor or the centroid of the resorted pitch chroma.

5.5.2 Key Recognition

Using the assumptions that (a) the occurrence of key inherent pitch classes is more likely than the occurrence of non-key pitch classes and that (b) the number and energy of occurrences is an indication of the key, the two basic processing steps for automatic *key detection* are

- the extraction of a pitch chroma to estimate a pitch class distribution, and
- the computation of the likelihood of the extracted pitch chroma being a specific key and selecting the most likely one.

The various approaches of extracting the pitch chroma have been discussed in Sect. 5.5.1. The time frame over which the pitch chroma is extracted can vary from the whole file to a common texture window length. It is also possible to extract the pitch chroma only at the beginning and end of the audio file because the key is usually less ambivalent in these regions [153].

5.5.2.1 Key Profiles

The likelihood of a specific key is usually computed by comparing the *extracted* pitch chroma with a *template* pitch chroma, in the following referred to as key profile. The template minimizing the distance to the extracted pitch chroma will be the most likely key.

Various templates can be used (see also the summary in Table 5.9 for $C\ Major$):

- *Orthogonal* ν_o: The orthogonal template assumes that the root note is the most salient component of the pitch chroma and that all other keys have the same distance (and are thus unrelated). The template allows no distinction between major mode and minor mode.

- *Smoothed Orthogonal* ν_s: The smoothed orthogonal template is, as the name says, a low-pass filtered version of the simple orthogonal template. It results in an increasing distance with increasing index distance with a distance maximum for the key at the tritone.

- *Diatonic* ν_d: The diatonic template is 1 for every key inherent pitch class and 0 otherwise. The distance to other keys increases linearly like an "unwrapped" circle of fifths. It allows no distinction between major mode and minor mode.

- *Circle of Fifths* ν_5: The distance of two keys can be modeled by its distance in the circle of fifths with respect to its radius r. The coordinates of each key are given by the angle of the key in the circle of fifths. Each key has the same distance r to the point of origin. The model can be expanded into a third dimension to also include a distance between major mode and minor mode.

- *Probe Tone Ratings* ν_p: Krumhansl's probe tone ratings are not directly a key description but are the result of a listening test evaluating how a pitch fits into a given tonal context [155]. However, Krumhansl showed in experiments that these probe tone ratings correlate well with the number of occurrences of the pitch classes in real pieces of music.

- *Extracted Key Profiles* ν_t: Key profile vectors can also be derived from real-world data. The template given in Table 5.9 has been published by Temperley and has been extracted from symbolic data [169].

Table 5.9 Various key profile templates, normalized to a sum of 1

	ν_o	ν_s	ν_d	ν_5	ν_p	ν_t
$\nu(0)$	1	0.44721	0.37796	$r \cdot e^{j2\pi \frac{0}{12}}$	0.49483	0.49355
$\nu(1)$	0	0.44721	0	$r \cdot e^{j2\pi \frac{-5}{12}}$	0.17377	0.039589
$\nu(2)$	0	0.44721	0.37796	$r \cdot e^{j2\pi \frac{2}{12}}$	0.27118	0.32199
$\nu(3)$	0	0	0	$r \cdot e^{j2\pi \frac{-3}{12}}$	0.18157	0.054105
$\nu(4)$	0	0	0.37796	$r \cdot e^{j2\pi \frac{4}{12}}$	0.34131	0.44208
$\nu(5)$	0	0	0.37796	$r \cdot e^{j2\pi \frac{-1}{12}}$	0.31872	0.30352
$\nu(6)$	0	0	0	$r \cdot e^{j2\pi \frac{6}{12}}$	0.19637	0.063343
$\nu(7)$	0	0	0.37796	$r \cdot e^{j2\pi \frac{1}{12}}$	0.40443	0.47177
$\nu(8)$	0	0	0	$r \cdot e^{j2\pi \frac{-4}{12}}$	0.18624	0.068621
$\nu(9)$	0	0	0.37796	$r \cdot e^{j2\pi \frac{3}{12}}$	0.28521	0.24149
$\nu(10)$	0	0.44721	0	$r \cdot e^{j2\pi \frac{-2}{12}}$	0.17845	0.03761
$\nu(11)$	0	0.44721	0.37796	$r \cdot e^{j2\pi \frac{5}{12}}$	0.22443	0.26393

The top left plot of Fig. 5.25 shows the listed key profiles in a bar graph. In this specific plot they are normed to a vector length of 1 with Eq. (5.56) as opposed to a sum of 1 as the entries in Table 5.9.

5.5.2.2 Similarity Measure between Template and Extracted Vector

The most likely key can be estimated by finding the minimum distance between the extracted pitch chroma $\boldsymbol{\nu}_\mathrm{e}$ and the key profile template. Under the reasonable assumption that the template key profiles for different keys are identical but shifted versions of the $C\ Major$ profile, only 1 template needs to be stored per mode. Detecting modes other than major mode and minor mode is not of relevance in practice; thus, only 2 template vectors have to be stored to generate an overall set of 24 shifted templates.

The index m of the most likely key can then be computed by finding the minimum distance d between the extracted pitch chroma $\boldsymbol{\nu}_\mathrm{e}$ and the set of 24 template key profiles $\boldsymbol{\nu}_\mathrm{t}$:

$$m = \min_{0 \le s \le 24} \left(d(s) \right). \tag{5.58}$$

Typical distance measures are

- *Euclidean Distance*:

$$d_\mathrm{E}(s) = \sqrt{\sum_{j=0}^{11} \left(\nu_\mathrm{e}(j) - \nu_{\mathrm{t,s}}(j) \right)^2}. \tag{5.59}$$

- *Manhattan Distance*:

$$d_\mathrm{M}(s) = \sum_{j=0}^{11} \left| \nu_\mathrm{e}(j) - \nu_{\mathrm{t,s}}(j) \right|. \tag{5.60}$$

- *Cosine Distance*:

$$d_\mathrm{C}(s) = 1 - \left(\frac{\sum_{j=0}^{11} \nu_\mathrm{e}(j) \cdot \nu_{\mathrm{t,s}}(j)}{\sqrt{\sum_{j=0}^{11} \nu_\mathrm{e}(j)^2} \sqrt{\cdot \sum_{j=0}^{11} \nu_{\mathrm{t,s}}(j)^2}} \right). \tag{5.61}$$

- *Kullback-Leibler Divergence*:

$$d_\mathrm{KL}(s) = \sum_{j=0}^{11} \nu_\mathrm{e}(j) \cdot \log \left(\frac{\nu_\mathrm{e}(j)}{\nu_{\mathrm{t,s}}(j)} \right). \tag{5.62}$$

Figure 5.25 plots the inter-key distances between the shifted key profile templates given above with respect to $C\ Major$; the distances are shown both in a line plot (bottom left) and within the circle of fifths (right). The Euclidean distance has been used to compute these diagrams; therefore it is only logical that the input key profiles have been normed to a vector length of 1 (as opposed to a sum of 1). The distances basically behave as anticipated: the orthogonal template has the same high distance to all other keys while for the smoothed orthogonal template the distance increases to a maximum at the tritone — this would be musically reasonable were it not for the fact that the keys with root notes

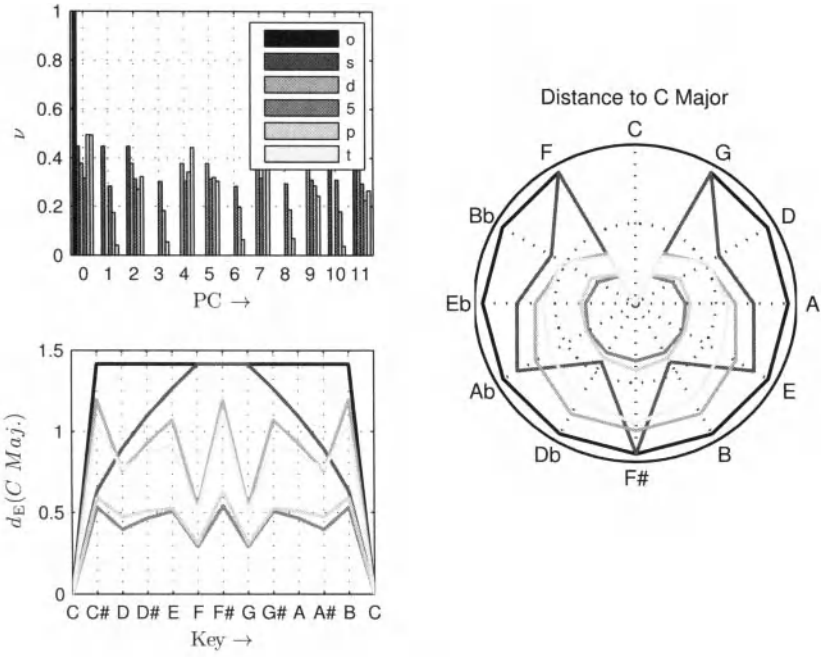

Figure 5.25 Key profile vectors (top left) and the resulting inter-key distances (Major) to $C\ Major$ (bottom left and right)

surrounding the tritone are the most closely related but have the same maximum distance. The remaining profiles basically show all the tendency of greater distances for keys more distant according to the circle of fifths. This is demonstrated by the cardioid shape of these distances in the polar diagram. It should be pointed out that the graph only visualizes the distances for keys in major mode and that none of the templates but the probe tone ratings and the extracted key profiles are able to separate between major mode and minor mode keys (although the model based on the circle of fifths can be adapted to a 3D model).

Theoretically, these inter-key distance measures are systematically wrong as the pitch chroma is a distribution and no vector. However, vector distances have been used and appear to work reasonably well in the absence of more fitting distance measures.

It is possible to smooth the pitch chroma to avoid maximum distances for nearby pitches [170] which in the case of tonality perception is probably most effective if the pitch chroma is reordered in a way that related keys have similar pitch chroma indices as described in Sect. 5.5.1.2.

There are other more complex mathematical models for estimating a distance between two keys or pitch class distributions; one example is Chew's *spiral array model* [171, 172]. Another model targeted specifically at automatic tonalness detection is the multi-dimensional *tonal centroid* proposed by Harte et al. [165] for which the pitch chroma is converted into a six-dimensional representation called tonal centroid based on the so-called harmonic network or *Tonnetz*. If enharmonic equivalence can be assumed, the Tonnetz can

be transformed into three two-dimensional planes representing the circle of fifths, major thirds, and minor thirds. Another model is Gatzsche's *circular pitch space* [173].

5.5.2.3 Typical Key Detection Errors

The most frequent error of automatic key detection systems is — unsurprisingly — the confusion with closely related keys, namely the keys with a dominant and a subdominant relationship as well as the parallel key. Another typical error surfacing mainly with popular music is the confusion of major modes with minor modes and vice versa while correctly estimating the root note.

When detecting the key of pieces with modulations, i.e., of pieces with a changing key, analyzing only small sections at the start and end of the piece of music will improve results as it is more than common that the piece will start and end in the same (main) key [153, 166, 174].

5.6 Chord Recognition

Similar to key detection systems, the automatic recognition of chords utilizes the pitch chroma, with the difference that the pitch chroma is extracted from a shorter segment in the piece of music as the focus is on the local tonal context. Most modern systems extract one pitch chroma for the period between each pair of neighboring beats.

The pitch chroma is commonly mapped to a chord probability vector; the chord estimation itself is often based on heuristic or statistical models for the progression of chords over time. The transformation of the pitch chroma into the chord space is in the simplest case done with a linear transformation by the chord transformation matrix $\mathbf{\Gamma}$. Since the pitch chroma has the dimensions 12×1, the transformation matrix would be of dimension $\mathcal{T} \times 12$ with $\mathcal{T}$ being the number of chord templates. For example, the number of chord templates will be $\mathcal{T} = 24$ if only all major and minor triads are allowed. The transformation can be formulated as

$$\boldsymbol{\psi}(n) = \mathbf{\Gamma} \cdot \boldsymbol{\nu}(n) \tag{5.63}$$

and can also be interpreted as the correlation between pitch chroma and each template (each row).

The resulting chord vector $\boldsymbol{\psi}(n)$ then has the dimension $\mathcal{T} \times 1$ and is a measure of the salience or likelihood of a specific chord given the pitch chroma $\boldsymbol{\nu}(n)$.

Each row of the transformation matrix represents one chord template; the simplest template would be to weight each pitch which is part of the chord by 1 and the remaining pitches with 0. Each row could be normalized to the sum of all entries in this row in order to compute the arithmetic mean. In this case, the template for a $C\ Major$ triad would be $[1/3, 0, 0, 0, 1/3, 0, 0, 1/3, 0, 0, 0]$.

Other transformations than a linear transformation are possible; the matrix multiplication (which can be seen as the calculation of the arithmetic mean of the result of multiplying each chord template with the pitch chroma):

$$\psi(0, n) = \sum_{j=0}^{11} \Gamma(0, j) \cdot \nu(j, n) \tag{5.64}$$

could, for example, be replaced by computing the geometric mean

$$\psi_{\mathrm{G}}(0, n) = \prod_{j=0}^{11} \nu(j, n)^{\Gamma(0,j)}. \tag{5.65}$$

It is also possible to use each row of the matrix directly as a template and compute a distance measure between extracted pitch chroma and the template.

Simply calculating the instantaneous probability of a chord while neglecting the likelihood of certain chord progressions would mean to ignore typical and well-known musical "standards" and to dismiss valuable information allowing us to improve the algorithm's reliability and robustness. Thus, nearly every system for automatic chord detection utilizes a model for chord progressions. This model is either analytically derived or trained from a set of data. There are three basic properties that determine the musicological validity of the model:[4]

- *Key Dependence*: a specific chord will have different preferred progressions dependent on the tonal context. Let us consider a typical cadence progression with the chords on the scale degrees I $\rightarrow$ IV $\rightarrow$ V $\rightarrow$ I. In the key $A\ Major$ ($A \rightarrow D \rightarrow E \rightarrow A$) this would imply that the dominant's ($E\ Major$) preferred transition would usually be to the tonic ($A\ Major$). However, were we in the key $B\ Major$, $E\ Major$'s harmonic function would be the subdominant with a relatively high likelihood of the following chord being $F\sharp\ Major$ (which would not even be part of the key $A\ Major$). Thus, chord progression models (theoretically) have to use the key of the piece of music.

- *Model Order*: musical context spans usually a larger area than just the neighboring chords. Therefore, the likelihood of a specific chord may depend not only on the directly preceding chord but on several preceding chords and possibly on some following chords.

- *Style Dependence*: different musical styles are based on different rule sets and different musical expectations. The transition probabilities between different chords will depend on the style of the piece of music being analyzed.

A typical approach to model the transition probability between chords is to use a *Hidden Markov Model (HMM)*. The states of the HMM represent the possible chords through the (possibly transformed) pitch chromas. The observation vectors can be either set a priori by applying "musical" knowledge or can be trained from a training data set. Examples for a priori settings are the simple chord template mentioned above [164] or the same templates weighted to take into account the influence of harmonics [161].

A very simple (and key independent) model for the chord progression likelihood is to use a circle of fifths [164] with the model errors mentioned above. A related approach is to utilize the correlation between Krumhansl's key profiles as chord distances [161].

The training data set can either be annotated audio [164], MIDI-synthesized audio [175], or a symbolic score format [161].

An example for a system acknowledging the key dependence has been published by Lee and Slaney by using one HMM for each of the 24 keys [176].

[4]It should be noted that these properties do not necessarily directly reflect on the algorithm's accuracy.

CHAPTER 6

TEMPORAL ANALYSIS

The temporal aspects of music signals such as the tempo and the rhythm are important musical properties. A fundamental building block of these aspects is the onset: the beginning of a musical sound event such as a tone or the stroke on a percussive instrument. The start time of an event is usually considered to be more important than the time of the end of that event, as listeners apparently perceive musical events more in terms of onset-to-onset intervals [177].

6.1 Human Perception of Temporal Events

During the process of human perception, the audio stream will be segmented into a series of events; speaking of segmentation is a simplification because musical meaning and even rhythm can be conveyed by audio streams with no such clear division into distinct events [178]. However, this simplification can be assumed to be sufficiently valid in the context of western music for the majority of possible input signals — other incarnations of temporal information will be ignored for the sake of simplicity.

6.1.1 Onsets

As stated above, an *onset* is the start of a (musical) sound event. The term *onset* is frequently used as a synonym to onset time, but it should be more correct to state that its time

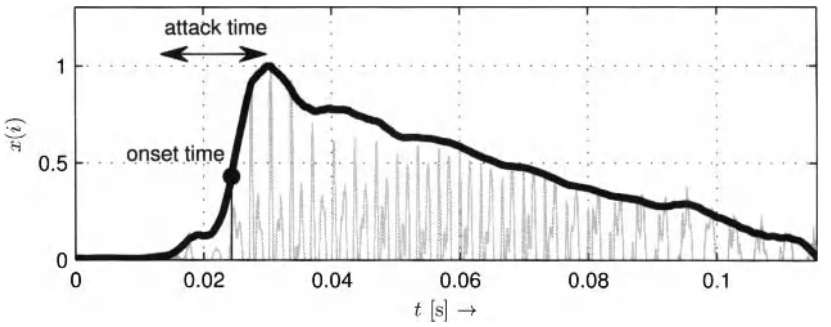

Figure 6.1 Visualization of an envelope and attack time and one possible location for an approximation of the perceptual onset time

position (i.e., the onset time) is one — most definitely the main — property of the onset while it can have other properties such as its strength.

In reality, the start of a musical sound usually is not an exact point in time, but a time span, the *attack time* or *rise time*. It is the time from the first instrument-induced oscillation until a maximum amplitude is reached. An example of an attack phase is shown in Fig. 6.1.

Sometimes the attack time is differentiated from the *initial transient time* which ends when the note reaches its quasi-periodic state. Obviously this differentiation works only for tonal events. The attack time can vary significantly between different musical instruments or groups of instruments. It ranges from about 5 ms for some percussive instruments to up to 200 ms for woodwind instruments (flute) under certain conditions [46].

The exact usage of the terms *onset*, *attack*, and *transient* is sometimes inconsistent and confusing. To give an example of a different naming convention than the one used here, Bello et al. propose to use the terms *attack* for our attack time, *transient* as a description of the initial phase of a musical event in which "the signal evolves quickly in some nontrivial or relatively unpredictable way" (the period covered by our attack time), and *onset* as a single instant chosen to mark the temporally extended transient (our onset time) [179].

Repp pointed out that three definitions of onset times can generally be distinguished [180]:

1. *Note Onset Time (NOT)*: the time when the instrument is triggered to make a sound. In the MIDI domain, the NOT is exactly the time of the Note-On command. Depending on the instrument or sample used for sound generation, this is not necessarily the time when the signal becomes detectable or audible.

2. *Acoustic Onset Time (AOT)*: the first time when a signal or an acoustic event is theoretically measurable. Sometimes the AOT is called *physical onset time*.

3. *Perceptual Onset Time (POT)*: the first time when the event can be perceived by the listener. The POT might also be distinguished from the *Perceptual Attack Time (PAT)* which is the instant of time that is relevant for the perception of rhythmic patterns [181]. While the PAT might occur later than the POT, they will be equal in many cases. For the sake of simplicity, there will be no distinction of POT and PAT in the following.

The POT can never occur before the AOT, which in turn never occurs before the NOT. Due to the "perceptual" definition of the POT, the exact location cannot be determined acoustically but has to be measured in a listening test. Both Gordon and Zwicker found strong location drifts of the PAT depending on the waveform properties during the rise time [47, 181]. There are indications of the POT to be correlated with the envelope slope [181].

Given the three definitions above, the following question arises: which of the three onset times is on the one hand detectable in the signal and on the other hand of the utmost interest in automatic onset detection and any rhythm-related task? Due to the symbolic nature of the NOT, it simply cannot be detected from the audio signal. The choice between AOT and POT might be application-dependent; assuming that musicians adapt their timing to their sound perception and that most ACA systems try to analyze the *perceptible* audio content, the POT is most likely the point in time desired as result. This reflection, however, is rather academic since in reality the accuracy of automatic onset detection systems is usually too poor to differentiate between the different onset times for all but a small class of signal types. The algorithm designer will probably strive to improve the detection performance of an onset detection system as opposed to its accuracy.

The human ability to locate onset times and to distinguish closely spaced onsets is of specific interest when estimating the required time accuracy of an onset detection system since most systems aim to be at least as accurate as the human perception.

Hirsh found that temporal discrimination of two onsets is possible for humans if the onset time difference is as little as 2 ms [182]. However, in order to determine the order of the stimuli, their distance had to be about 20 ms. The measurements were done with synthetic signals with short rise times.

Gordon reported a standard deviation of 12 ms for the accuracy of onset times specified by test listeners, using 16 real-world monophonic sounds of different instruments played in an infinitely long loop pattern with *Inter-Onset Intervals (IOIs)* of 600 ms [181]. Friberg and Sundberg undertook a similar experiment using tone stimuli [183]. For IOIs smaller than 240 ms, they reported a just noticeable difference of about 10 ms, and increasing values for larger IOIs.

Repp reported for the manual annotation of onset times by one listener in the context of piano recordings a mean absolute measurement error of about 4.3 ms and a maximum error of about 35 ms [184]. In a recent investigation, Leveau et al. had three test subjects annotating the onset times in audio files of various genres and instrumentations [185]. The results showed a mean absolute measurement error over all test data of about 10 ms; for one piece of classical music, the mean absolute measurement error nearly reached 30 ms.

Rasch evaluated the onset time differences between instruments in three ensemble performances [186]. He found synchronization deviations in a range between 30 and 50 ms between the (string and woodwind) instruments, while the mean onset time differences were in the range of ± 6 ms. However, since the measurement accuracy has not been evaluated in this case, it is unknown how much of the actual time differences can be attributed to the performance itself.

For piano duet performance, Shaffer reported standard deviations within the voices between 14 and 38 ms [187].

It may be concluded that the accuracy of human onset perception depends on the test data and that deviations evoked by motoric abilities seem to be in the same range. The presented results imply that an automatic onset detection system aiming at human detection accuracy (or being evaluated with test data annotated by humans) will have a minimum mean absolute error in the range of 5-10 ms; the error can be expected to be as high as 10 times more for specific instruments and pitches with long rise times. A real-world aspect

with negative impact on the onset detection "accuracy" is the occurrence of several quasi-simultaneous onsets in polyphonic music. In this case the deviation between the individual voices will virtually decrease the system's accuracy, although it may be argued that in this case there is no single reference onset.

6.1.2 Tempo and Meter

The *tempo* is the rate at which perceived pulses with equal duration units occur at a moderate and natural rate [188]. This perceived tempo is called the *tactus* [189] and is sometimes simply referred to as the *foot tapping rate* [190]. A typical natural rate would be located around 100 *Beats per Minute (BPM)* [191].

For segments of music with constant tempo, the tempo $\mathfrak{T}$ in BPM can be computed using the length of the segment Δt_s in seconds and the number of beats $\mathcal{B}$ in the segment:

$$\mathfrak{T} = \frac{\mathcal{B} \cdot 60\,\mathrm{s}}{\Delta t_\mathrm{s}} \ [\mathrm{BPM}]. \tag{6.1}$$

In the case of a dynamic tempo, the local tempo can be extracted by identifying the event time of every beat t_b and computing the distance between two neighboring beats with indices j and $j+1$:

$$\mathfrak{T}_\mathrm{local}(j) = \frac{60\,\mathrm{s}}{t_\mathrm{b}(j+1) - t_\mathrm{b}(j)} \ [\mathrm{BPM}]. \tag{6.2}$$

Deriving an *overall* tempo becomes increasingly difficult when the tempo is not constant; in this case the mean tempo given in Eq. (6.1) does not necessarily match the *perceived tempo* a listener would indicate. This led Gabrielsson to distinguishing between the mean tempo and the *main tempo*, the latter being a measure ignoring slow beginnings or final *ritardandi* [192]. Repp found good correlation of the perceived tempo with the mean value of a logarithmic IOI distribution [193]. Goebl proposed a *mode tempo* which is computed by sweeping a window over the histogram of *Inter-Beat Intervals (IBIs)* and selecting the maximum position as mode tempo [194].

McKinney and Moelants complicated matters further by arguing that a single tempo does not sufficiently describe the (listener) group response when presented with a piece of music. They propose a representation of the overall tempo with two BPM values instead of a single one [195].

The *meter* is a regular alternation of strong and weak musical elements which are grouped with a length of normally three to seven beats or a length of around 5 s.

6.1.3 Rhythm

The perception of *rhythm* can — similar to the meter — be described by its grouping properties. The grouping properties allow a hierarchical segmentation into smaller subsequences forming different grouping levels. The length of the groups can range from the length of a few notes up to whole parts of the work defining musical form [189].[1] Groups of a length between one beat and the length of the meter are most commonly referred to as rhythm. The rhythm is then defined by its accents and time intervals; if the durations of subsequent intervals relate to simple integer ratios, then the group usually has a closer binding than otherwise [196].

[1] However, we will use the term *rhythm* only for groups with a length of up to several beats.

The various hierarchical levels of temporal grouping are an important property of many (western) pieces of music. Humans perceive pulses at different levels and with different tempi; at all levels the grouping of strong and weaker events occurs. The basic building block on the lowest (and shortest level is commonly referred to as tatum [197], although other terms such as *atomic beat* have been used [198]. The tatum specifies the lowest period length or the period of the regular pulse train with the highest frequency represented in the music. Every rhythm is built of the tatums which can be interpreted as a rhythmic grid or a time quantization. The length of the highest level grouping depends on the definition of grouping and could go up to the level defining musical form such as the length of musical phrases or even longer structures which form groups.

6.1.4 Timing

The *timing* of individual notes or temporal events in a music performance does not necessarily exactly reflect the structural properties of the rhythm or meter but shows systematic temporal deviations from the underlying rhythmic structure [45]. A detailed overview of expressive timing will be given in Chap. 10.

6.2 Representation of Temporal Events in Music

The representation of musical (temporal) events is closely related to the perception of such events for both terms and the musical score.

6.2.1 Tempo and Time Signature

The *overall tempo* of a piece of music is usually chosen by the performing artists even if the composer indicates a preferred tempo. Tempo instructions for the performers became more and more explicit over the centuries. While many pieces from the Baroque period do not contain instructions due to the composer's assumption that the tempo was specified by performance conventions, it became more and more common in later epochs to indicate the tempo with Italian terms such as *Largo* (very slow), *Adagio* (slow), *Andante* (walking pace), *Moderato* (moderately), *Allegro* (fast), and *Presto* (very fast). During the last century it became more common to make specific tempo indications in BPM.

The *local tempo* varies throughout a piece of music for nearly all genres. The possibilities to include instructions for such variations in the score are limited besides adding tempo indicators; examples of tempo instructions for sliding tempo changes are *ritardando* (slowing down) and *accelerando* (speeding up).

The *bar* (also called a *measure*) is the score equivalent of the (perceptual) meter. A score marks the beginning of each bar by a vertical line. The first beat of a bar usually has the highest (perceptual) weight and is referred to as *downbeat*.

The *time signature* is a way to convey information on the properties of a bar, namely the number of beats grouped together in one bar as well as the note value constituting one beat. The time signatures in Fig. 6.2 group four, three, two, and two beats, respectively. The fourth example differs from the first three in grouping half notes instead of quarter notes. The denominator of the time signature thus indicates the note value of one beat while its numerator indicates the number of beats per bar.[2]

[2]There are exceptions from this rule such as a time signature 6/8 with a beat length of three eighth notes.

Figure 6.2 Frequently used time signatures

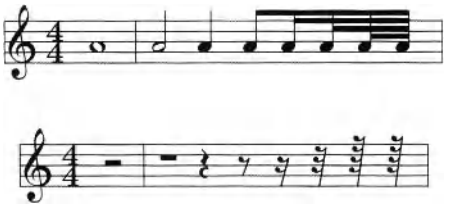

Figure 6.3 Note values (top) and corresponding rest values (bottom) with decreasing length in musical score notation

6.2.2 Note Value

The *note value* defines the relative length of a note with respect to time signature and tempo. Notational convention requires that the sum of note values and rest values per bar (except a few special cases) must result in the numerator of the time signature. Thus the absolute onset time of each note is specified by the bar index and the note's position in the bar, given a specific tempo.

The offset time (also the *note off time*) is determined by the note's onset time and its note value in the score but is not necessarily as clearly defined in a real-world performance. Sometimes a note value is shortened and a rest is appended to give the performing artists indications of the preferred articulation. In general, however, it is not unusual to place the responsibility for such articulation decisions on the performers rather than the musical score, but this depends also on epoch, style, and composer.

Figure 6.3 shows the most common note values (top) starting from a whole note and decreasing the value down to two sixty-fourth notes and the corresponding rests (bottom).

6.3 Onset Detection

Segmenting the audio stream into separate musical events can be an important processing step in applications such as tempo detection or the automatic transcription of music.

The flowchart of a typical *onset detection* system (also called *onset tracking* system) is shown in Fig. 6.4. First, a novelty function is computed extracting the amount of "new" information in the audio signal for each analysis block. The second processing step consists of identifying the locations of the significant maxima which can then be regarded as onset times. This second processing step is usually referred to as peak picking.

Overview articles for different approaches to detecting onsets have been published by Bello et al. and Dixon [179, 199].

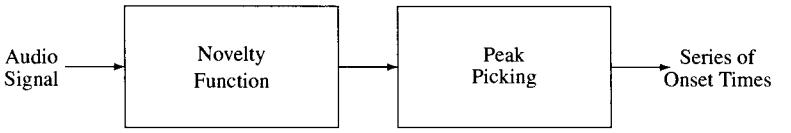

Figure 6.4 General flowchart of an onset detection system

6.3.1 Novelty Function

An important property of the beginning of musical sound events is that "something new happens." Thus, the first step toward automatic onset detection is the computation of a *novelty function* which indicates the amount of audio signal changes over time [200]. Other names of this function are *detection function* [179] or *difference function* [201].

The first step in the computation of the novelty function is usually the calculation of the difference between current and preceding feature values. The result is then smoothed and negative values are discarded by applying HWR. The latter processing step is usually helpful for onset detection as an (amplitude or energy) increase might be expected at onset times while a decrease should make an onset less likely.

In one of the first publications on onset detection in (percussive) music signals, Schloss presented an algorithm that makes direct use of the audio signal's envelope slope, extracted in the time domain [202]. He extracts the envelope of the audio signal by computing the maximum of the magnitude of the signal within a block of samples and recommends to adjust the block length to the length of the period of the lowest frequency present. The envelope slope is then computed by using linear regression over several points of the peak amplitude.

As pointed out in Chap. 4, there exist different possibilities to extract the envelope of a signal, including taking the block's maximum amplitude and low-pass filtering either the signal's magnitude or its RMS. This kind of envelope analysis is nowadays usually applied to specific frequency subbands as opposed to the time domain signal.

Most of the concurrent systems use STFT-based techniques for computing the novelty function, computed from the differences between subsequent (overlapping) STFT blocks. This can either be done with each individual spectral bin or with multiple bins grouped into frequency bands. The advantage of computing the novelty function in the frequency domain is that the onset detection is not only based on amplitude and envelope differences but might also take into account spectral differences such as a pitch change. The disadvantage of the frequency domain computation is the comparably poor time resolution which affects the algorithm's detection accuracy.

The number of frequency bands of onset detection systems varies. Scheirer uses a filterbank of 6 bands basically covering a one-octave range [203]. Klapuri uses 21 non-overlapping bands with their band width and mid-frequency inspired by Zwicker's critical bands [204]. Duxbury uses 5 bands up to 22 kHz with constant Q. The individual results per subband are either combined into one overall novelty function or they are processed per frequency band to be combined only for the final result.

While spectral domain onset detection systems differ in the number of frequency bands they analyze, their main difference is the distance measure $d(n)$ between consecutive STFTs.

Laroche used a distance similar to the spectral flux with an additional square root function to increase lower signal amplitudes [205]:

$$d_{\mathrm{lar}}(n) = \sum_{k=k(f_{\min})}^{k(f_{\max})} \sqrt{|X(k,n)|} - \sqrt{|X(k,n-1)|}, \tag{6.3}$$

Duxbury et al. proposed the distance between complex STFT bins [206]:

$$d_{\mathrm{dux}}(n) = \sum_{k=0}^{K/2-1} |X(k,n) - X(k,n-1)|, \tag{6.4}$$

while Hainsworth and Macleod calculated a logarithmic distance [201]:

$$d_{\mathrm{hai}}(n) = \sum_{k=0}^{K/2-1} \log_2 \left(\frac{|X(k,n)|}{|X(k,n-1)|} \right). \tag{6.5}$$

It is also possible to compute the distance with the cosine distance between two STFT frames as suggested by Foote [200]:

$$d_{\mathrm{foo}}(n) = 1 - \frac{\sum_{k=0}^{K/2-1} |X(k,n)| \cdot |X(k,n-1)|}{\sqrt{\left(\sum_{k=0}^{K/2-1} |X(k.n)|^2 \right) \cdot \left(\sum_{k=0}^{K/2-1} |X(k,n-1)|^2 \right)}}. \tag{6.6}$$

Bello et al. pointed out that phase relations may be used for the detection of novelty in an audio stream as well [207]. Here, the principles of instantaneous frequency computation (see Sect. 2.2.3.1) are applied and the difference of the unwrapped phases $\tilde{\Phi}$ is used:

$$d_{\mathrm{bel}}(k,n) = \mathrm{princarg} \left[\tilde{\Phi}_X(k,n) - 2\tilde{\Phi}_X(k,n-1) + \tilde{\Phi}_X(k,n-2) \right]. \tag{6.7}$$

Goto and Muraoka proposed a distance which compensates for slow frequency variation over time [208]. They identify all bin indices k with

(a) higher power than the maximum of the four closest preceding bins:

$$\begin{aligned} A &= X(k,n-1)^2, \\ B &= X(k-1,n-1)^2, \\ C &= X(k+1,n-1)^2, \\ D &= X(k,n-2)^2, \\ E_{\max}(k,n) &= \max(A,B,C,D), \end{aligned} \tag{6.8}$$

and

(b) the same condition fulfilled for the maximum power of the three closest neighboring bins:

$$E_{k,n+1} = \max\left(X(k,n+1)^2, X(k-1,n+1)^2, X(k+1,n+1)^2\right). \tag{6.9}$$

The distance is then computed from the maximum of the current and following power value $E_t(k,n) = \max\left(X(k,n)^2, X(k,n+1)^2\right)$ by

$$d_{\text{got}}(k,n) = \begin{cases} E_t(k,n) - E_{\max}(k,n), & \text{if } \left(X(k,n)^2 > E_{\max}(k,n)\right) \wedge \\ & \left(E_{k,n+1} > E_{\max}(k,n)\right) \\ 0, & \text{otherwise} \end{cases}, \tag{6.10}$$

$$d_{\text{got}}(n) = \sum_{k=0}^{\mathcal{K}/2-1} d(k,n). \tag{6.11}$$

This distance strongly depends on the ratio of STFT size and sample rate as well as the block overlap ratio.

Röbel proposed a transient detection that utilizes the COG of the instantaneous energy [209]. He calculates this COG per arbitrary frequency band with

$$t_{\text{cg}}(t_m) = \frac{\int_{\omega_l}^{\omega_h} -\frac{\partial\Phi(\omega,t_m)}{\partial\omega} |X(\omega,t_m|^2 \, d\omega}{\int_{\omega_l}^{\omega_h} |X(\omega,t_m)|^2 \, d\omega}. \tag{6.12}$$

The derivation of time reassignment is closely related to frequency reassignment and is a way of virtually improving the time resolution.

Zhou and Reiss presented an onset detector utilizing a filterbank of first-order complex resonators with an overall number of 960 frequency bands and 10 filters covering the range of a semi-tone, respectively [210]. They distinguish between hard and soft onsets and use the half-wave rectified energy difference per band for the detection of hard onsets and a pitch-based detection for so-called soft onsets by detecting steady-state tonal components and locating the corresponding onset position by searching for a salient energy increase.

6.3.2 Peak Picking

Although some systems for the automatic extraction of tempo and rhythm features utilize the extracted novelty function directly, other systems use it only as an intermediate result from which a series of onset times is derived. This is done by *peak picking* the novelty function. The final tempo and rhythm features are then derived from this series of onsets.

A flawless novelty function would indicate an onset at each local maximum. In reality, however, using the locations of local maxima directly as onset times will cause a large number of falsely detected onsets [*False Positives (FPs)*]. To suppress peaks of no interest, a threshold G is applied to the novelty function and only peaks above this threshold are considered as onset position candidates.

In the simplest case, the threshold is a fixed threshold:

$$G_{d,\text{c}} = \lambda_1. \tag{6.13}$$

An alternative to this fixed threshold is using a signal-adaptive threshold. This adaptive threshold could be computed from the smoothed version of the novelty function. A typical smoothing filter is the MA filter:

$$G_{d,\text{ma}} = \lambda_2 + \sum_{j=0}^{\mathcal{O}-1} b(j) \cdot d(i-j) \tag{6.14}$$

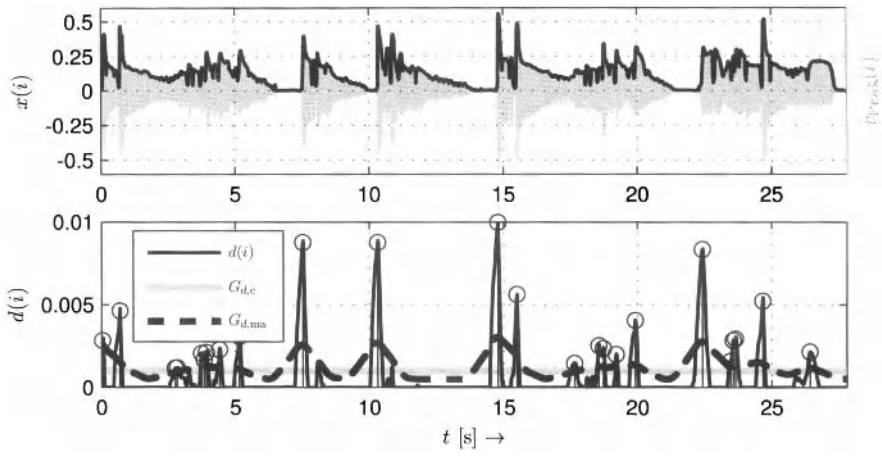

Figure 6.5 Audio signal and extracted envelope (top) and novelty function with exemplary thresholds for the peak picking process

with $b(j)$ representing a user-defined window function. The weight λ_2 shifts the threshold for adjusting the algorithm's sensitivity. Alternatively, a *median filter* may replace the MA filter; its output $\hat{Q}_d(0.5)$ is an estimate of the median in the block of an appropriate length $\mathcal{K}$:

$$G_{d,\mathrm{me}} = \lambda_2 + \hat{Q}_d(0.5). \tag{6.15}$$

Figure 6.5 exemplifies the process of peak picking by displaying a simple amplitude-based novelty function and two thresholds (static and adaptive) for the detection of local maxima. The input signal is a monophonic saxophone recording.

There are other ways of increasing the robustness of the onset detection system. Examples of additional criteria are the usage of the "amplitude" distance between the local maximum and the preceding local minimum as an indicator of onset strength and the extraction of the novelty function's slope before the local maximum as a cue for the likelihood of an onset.

An additional post-processing step is applied occasionally: the detected onsets may have very close proximity in time, and depending on the task at hand it might be beneficial to combine two or more closely neighbored onsets into one. The exact process of combining several onsets is, however, not as straightforward as one could wish. Possibilities are to choose the earliest onset, to choose the onset with the highest weight, or to compute some kind of average as the resulting (combined) onset time.

The final result of the onset detection system is a series of estimated onset times $\hat{t}_\mathrm{o}(j)$, possibly including an additional weight or intensity information as additional property. The markers in Fig. 6.5 indicate the estimated onset positions.

6.3.3 Evaluation

The *evaluation* of onset detection systems is a good example for the evaluation of ACA systems in general. Early publications on onset detection described their evaluation methodology and evaluation metrics as far less elaborate than the algorithm itself. To give an example, the number of correct detections was commonly reported with only a fuzzy de-

scription of what the definition of a correct detection actually is. The lack of information on both the test procedure and the test signals made it nearly impossible to estimate the algorithm's detection performance and to compare the results between different publications on onset detection. Only during the last decade has the problem of proper evaluation moved into the researcher's focus. In the context of the *Music Information Retrieval Evaluation eXchange (MIREX)*,[3] an annual evaluation campaign for MIR algorithms coupled to the *International Society for Music Information Retrieval (ISMIR)*, effort has been made in proposing a standardized test environment for audio onset detection systems.

The main problems in evaluating audio onset detection systems can be summarized as:

- *Lack of proper definition of the term onset*: Frequently it is neither entirely clear what the system aims to detect exactly (e.g., AOT vs. POT) nor what the required measurement accuracy is.
- *Lack of an adequate amount of test material*: The effort and error-proneness of the manual annotation of onset times in real-world signals makes it difficult to produce a sufficient amount of ground truth test data.
- *Lack of standardized and critical test material*: Comparison between different algorithms is hard without publicly available training and test databases.
- *Lack of simple yet meaningful evaluation metrics*: The evaluation results have to be computed and presented in a way that enables the estimation and comparison of the system's accuracy and robustness.

6.3.3.1 Procedure

The following parameters may be of interest when evaluating onset detection systems:

- detection performance,
- detection accuracy,
- robustness for noisy and band-limited input signals, and
- workload of the algorithm.

For each of these parameters, the definition of meaningful rating metrics with a predefined range is desirable. The type, the properties, and the amount of test signals and ground truth has to be specified to make results as comparable as possible.

Detection Performance

The *detection performance* is probably of the highest interest in the evaluation of onset detection systems. Obviously it should be a measure of how many onsets were correctly detected and how many onsets were incorrectly detected. The extracted onset times have to be compared with previously defined reference onset times as given by the ground truth. Two possible errors can occur: a *False Negative (FN)* indicating that no onset is detected at the time of a reference onset, and a *False Positive (FP)* which is an onset that is wrongly detected where no reference onset is found. Both of these measurements presume the definition of a time tolerance window around the reference onset time in which a detection

[3]MIREX Home. http://www.music-ir.org/mirex. Last retrieved on Nov. 25, 2011.

Table 6.1 Overview of different descriptions of the number of correct and incorrect detections and their relation

	Det. Positives	*Det. Negatives*	Σ
Ref. Positives	O_{TP}	O_{FP}	O_{DP}
Ref. Negatives	O_{FN}	O_{TN}	O_{DN}
Σ	O_{RP}	O_{RN}	

is counted as correct; usually, the length of this window is set to 50–100 ms. Similar to the definition of FPs and FNs, the *True Positives (TPs)* are the correctly detected onsets and the *True Negatives (TNs)* are the positions at which correctly no onset has been detected. In summary, the detection performance can depend on the following values:

- the number of TPs (correctly detected onsets) O_{TP},
- the number of FPs (falsely detected onsets) O_{FP},
- the number of FNs (missed onsets) O_{FN},
- the number of onsets in the reference data set O_{RP}, and
- the number of detected onsets $O_{\mathrm{DP}} = O_{\mathrm{TP}} + O_{\mathrm{FP}}$

Table 6.1 visualizes these numbers and their inter-relationship. It refers to onsets as *positives* and non-onsets as *negatives* to generalize the table to a two-class problem.

The internal parameters of the onset detection system should be adjusted for the desired "working point" before the evaluation itself can be carried out. The so-called *Receiver Operating Curve (ROC)* is an intuitive way to visualize the trade-off between TPs and false detections [211]. In the case of the evaluation of onset detection we would plot on one axis the TPs, on the other axis the sum of FPs and FNs. Each different parametrization of the algorithm results in exactly one point in the two-dimensional space of the ROC plot. The parameterizations of interest are the ones that maximize the TPs and at the same time minimize the false detections. The ratio of FPs and FNs can also be of interest for algorithm parameterization. If these two errors are considered to be equally bad, then the ratio should be near the value 1.

Several measurements of detection performance have been proposed in the past. Cemgil et al. proposed the relation of the total number of detections O_{DP}, the number of FNs O_{FN}, and the total number of reference onsets O_{RP} as a measure of detection performance [212]:

$$q_{\mathrm{cemgil},1} = \frac{O_{\mathrm{DP}} - O_{\mathrm{FN}}}{O_{\mathrm{RP}}}. \tag{6.16}$$

While this is a simple definition of the detection rate, it does not take into account the falsely detected additional onsets, and thus can result in misleading values in the case of many FPs.

Liu et al. proposed a similar value for the detection rate, additionally taking into account the number of FPs O_{FP} [213]:

$$q_{\mathrm{liu}} = \frac{\max(O_{\mathrm{DP}}, O_{\mathrm{RP}}) - (O_{\mathrm{FN}} + O_{\mathrm{FP}})}{\max(O_{\mathrm{DP}}, O_{\mathrm{RP}})}. \tag{6.17}$$

At least theoretically, the result can be negative; this is not desirable for a detection rate measure that should be in the range between 0 and 1.

A different reliability measurement (a measurement of relative error) has been proposed by Lerch [214]

$$: q_{\text{lerch}} = \frac{O_{\text{DP}} - (O_{\text{FN}} + O_{\text{FP}})}{O_{\text{RP}} + (O_{\text{FN}} + O_{\text{FP}})}. \tag{6.18}$$

The resulting value has the desired range between 0 and 1. The number of missing detections has the same weight as the number of false positives. In some contexts it might be desirable to weight O_{FN} and O_{FP} by different values when one should have a higher influence than the other. In these cases, a scaling factor λ in the range of $[0; 1]$ can be introduced which weights the sum of missing and falsely detected onsets: $\lambda \cdot O_{\text{FN}} + (1 - \lambda) \cdot O_{\text{FP}}$. Then, however, there is the possibility of negative results of Eq. (6.18).

The established statistical evaluation measures precision P and recall R allow a more systematic approach. Precision is the fraction of correctly detected onsets from all detected onsets:

$$P = \frac{O_{\text{TP}}}{O_{\text{TP}} + O_{\text{FP}}} = \frac{O_{\text{TP}}}{O_{\text{DP}}} \tag{6.19}$$

and recall is the fraction of correct detections from all reference onsets:

$$R = \frac{O_{\text{TP}}}{O_{\text{TP}} + O_{\text{FN}}} = \frac{O_{\text{TP}}}{O_{\text{RP}}}. \tag{6.20}$$

Precision and recall can be combined into the so-called *F-Measure* F. Mathematically it is the harmonic mean of precision and recall:

$$F = \frac{2PR}{P + R} = \frac{2 \cdot O_{\text{TP}}}{2 \cdot O_{\text{TP}} + O_{\text{FP}} + O_{\text{FN}}} = \frac{2 \cdot O_{\text{TP}}}{O_{\text{RP}} + O_{\text{DP}}}. \tag{6.21}$$

Note that for the F-Measure to produce reliable results, the number of positives has to roughly equal the number of negatives in the test set. If this is not the case (which may very well be true in the case of onset detection) the results will be biased.

Until now we only evaluated correct versus incorrect detection. In addition, a detection performance measurement could also include the time distance between the reference and the detected onset time and thus include a measure of *detection accuracy* (see below). A detected onset would be weighted with respect to its proximity to the reference. An intuitive way to do so could be to weight the distance $\Delta t_{\text{R,D}}$ between reference and detected time with a window function $w(\Delta t)$. Cemgil et al. proposed such a measure in the context of evaluation of beat tracking systems with a Gaussian window function [compare the RBF in Eq. (5.20)] [215]. Adapted to the onset detection evaluation problem the evaluation measure would be

$$q_{\text{cemgil},2} = \frac{\sum_{\forall r} \max_{\forall t} w(\Delta t_{\text{R,D}})}{(O_{\text{RP}} + O_{\text{DP}})/2}. \tag{6.22}$$

This measurement has again the limitation that it is not able to correctly handle FPs.

Detection Accuracy

The *detection accuracy* evaluates the timing accuracy of the algorithm, as opposed to the *detection performance* which evaluated the number of correct and false detection within a relatively large tolerance window. The accuracy can be measured by investigating the time difference $\Delta t_{\text{R,D}}$ between reference and detected onset times. The distribution of the resulting time differences contains the necessary data for timing evaluation; values of

interest are the arithmetic mean which indicates the tendency of the system detecting onsets systematically too early or too late:

$$d_{\text{mean}} = \sum_{\forall j} \Delta t_{\text{R,D}}(j), \tag{6.23}$$

the absolute mean value indicating the average time distance between detected and reference onset is

$$d_{\text{abs}} = \sum_{\forall j} |\Delta t_{\text{R,D}}(j)|, \tag{6.24}$$

the standard deviation or a confidence interval

$$\sigma_d = \sqrt{\frac{1}{O_{\text{DP}}} \sum_{\forall j} (\Delta t_{\text{R,D}}(j) - d_{\text{mean}})^2}, \tag{6.25}$$

and the absolute maximum value of the deviation

$$d_{\text{max}} = \max_{\forall j} |\Delta t_{\text{R,D}}(j)|. \tag{6.26}$$

Furthermore, a measure of statistical significance such as the p-value (see, e.g., [216]) can be given to attest the reliability of the results.

Robustness and Workload

Some target applications require robustness of the algorithm against noisy and distorted signals or have specific requirements on the workload of a system.

The robustness against noise and bandwidth limitations can be carried out using the test scenario and metrics described above but with modified test signals. Possible modifications depend on the target application but could include added noise, down-sampling, and encoding with lossy audio encoders or speech encoders.

The evaluation of the algorithmic *workload* gives an estimate of the complexity and real-time capabilities of the system. This might be of particular interest if the algorithm has to run on embedded devices or has to process vast amounts of data. The actual investigation of the workload is more complex than it might seem at first glance. Even estimating the theoretical algorithmic complexity in a number of operations gets complicated as soon as specific functions such as trigonometric or exponential functions are used. The measurable execution time itself may be influenced by many different conditions, for example:

- the *hardware*: processor clock speed, vectorization (SIMD) functionality, cache size as well as memory access speed,
- the *implementation*: optimization of the source code, optimization of the compiler, and
- the *software*: operating system efficiency, audio and file IO.

Thus workload measurements usually give only rough impressions of the algorithms processing performance even on comparable hardware. The result of workload measurements can be given as the ratio between the required computation time t_{C} and the overall length of the tested audio data of the test database t_{A} by calculating $t_{\text{C}}/t_{\text{A}}$ with respect to the used processor.

6.3.3.2 Test Signal Databases

The test signal database for the evaluation of onset detection performance should preferably contain real-world signals such as signals in CD quality with the onset times annotated per hand as the ground truth. However, as several researchers point out, the manual annotation is a tedious and time-consuming task [184, 185]. Therefore, two alternatives for the generation of test sequences may be considered: acoustic recordings with a symbolic trigger (such as recordings of the Yamaha Disklavier) and audio data synthesized from symbolic data. In both cases, the reference onsets are available in the MIDI format, allowing the easy automated extraction of NOTs. Given the range of the typical tolerance interval of 50 ms, the difference between NOT and POT can sometimes be neglected.

The test database should include the following signal types to make the evaluation as general as possible:

- *various genres* (pop, rock, symphonic, chamber music, electronic, etc.),
- *various instrumentations*,
- *different tempi and musical complexity*, and
- *critical signals* which are signals with very low detection performance. In the case of onset detection these might include noisy signals and signals containing various kinds of tremolo and vibrato.

As mentioned above, databases with manually annotated audio files are difficult to find. This is on the one hand due to intellectual property issues, on the other hand due to the time-consuming task of annotation. Two examples for publicly available databases are the data set published by Leveau[4] and the data set published by Glover.[5]

6.4 Beat Histogram

The *beat histogram* or *beat spectrum* is a way to visualize some rhythmic properties of the signal. Similar to the "normal" magnitude spectrum, the frequency (in this case with the unit BPM) is assigned to the abscissa and the magnitude (beat strength) is assigned to the ordinate. Peaks in the histogram should therefore correspond to the main tactus and its integer multipliers and divisors. The beat histogram can be interpreted as the frequency domain representation of the novelty function. There are multiple ways of computing such a beat histogram.

Scheirer used a closely spaced filterbank of comb resonance filters and used the filter's output energy as indication of the *beat strength* [203].

Foote and Uchihashi proposed to construct a similarity matrix (compare the distance matrix $\boldsymbol{D}$ in Sect. 7.1) from the cosine distance between all pairs of STFTs from the audio file and then derive the beat histogram by summing the similarity matrix along its diagonal [217].

Tzanetakis and Cook split the audio signal into four octave bands and extract the envelope per band by applying four processing steps [60]:

1. *Full-Wave Rectification (FWR)* by computing the absolute value,

[4]Leveau, Pierre. https://sites.google.com/site/pierreleveau/research. Last retrieved on Nov. 25, 2011.
[5]Glover, John. http://www.johnglover.net/audiosoftware.html. Last retrieved on Nov. 25, 2011.

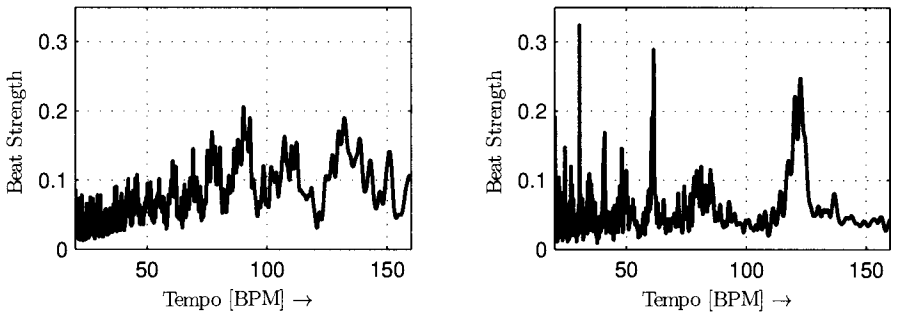

Figure 6.6 Beat histogram of a piece of popular music (left) and of a string quartet performance (right)

2. *envelope smoothing* by low-pass filtering,
3. *down-sampling* to reduce the complexity by reducing the sample rate, and
4. *DC removal* by subtracting the arithmetic mean.

An ACF (with harmonic processing as described in Sect. 5.3.4.2) is then computed in order to identify (rhythmic) envelope regularities. The beat histogram is construed by taking three peaks in the search range and adding their amplitude to the beat histogram. This is done for each texture window.

Figure 6.6 visualizes the beat histogram of an excerpt of popular music in comparison with the histogram extracted from a string quartet performance. Note that for this plot the beat histogram calculation is based on a very simple novelty function derived in the time domain from the signal's magnitude. After computing the ACF, the amplitude of each individual lag is mapped into the beat domain as the beat strength (this leads to a high resolution at low frequencies and a low resolution at high frequencies). The beat histogram computed from the popular music example has clearly defined peaks at multiples of a base frequency; such a pattern is not identifiable in the right beat histogram computed from a string quartet recording.

6.4.1 Beat Histogram Features

Similar to an audio magnitude spectrum, the beat histogram can be represented by (simple yet meaningful) features. Widely used is the set of features introduced by Tzanetakis and Cook consisting of [60]

- the overall sum of the histogram,
- the relative amplitude of the highest peak,
- the relative amplitude of the second highest peak,
- the amplitude ratio of second highest to highest peak, and
- the BPM frequencies of the highest and second highest peak.

Burred and Lerch evaluated a feature set including statistical features of the beat histogram such as its arithmetic mean, standard deviation, kurtosis, skewness, and entropy; they additionally used a measure for what they called *rhythmic regularity*. The rhythmic regularity is a measure of how much the computed ACF differs from the linear weighting function of a block-wise ACF [218].

6.5 Detection of Tempo and Beat Phase

Systems for *tempo detection* (also referred to as *tempo induction* or *tempo tracking*) usually compute some kind of novelty function in a first processing step. Regardless of whether discrete onset times are picked from the novelty function or the novelty function is used directly, it is far from trivial to derive the pulse which would be perceived as tactus since the main periodicity is frequently not clearly identifiable.

The detection of specific beat positions is not necessarily required for detecting the tempo itself as the tempo is based on the *distance* between the beats rather than their absolute position. Therefore, a periodicity analysis of the novelty function is sufficient for tempo estimation when knowledge of the absolute beat positions is not required.

Given a tempo, the absolute beat position is — due to the similarity to the relation of frequency and phase — referred to as the *beat phase*. This beat phase is, for example, required for the so-called beat matching technique for which two or more pieces of music with different musical content (but the same tempo) are synchronized at their beat positions to generate a so-called *mash-up*. Without knowing the exact beat positions it would not be possible to mix those pieces in a musically meaningful way.

The first beat tracking systems avoided the complexity of extracting a novelty function and focused exclusively on the beat tracking part; they used onset times from symbolic data such as MIDI files as input of their beat tracking system. Thus, they assumed a perfect computation of the novelty function and of the onset detection, respectively. Allen and Dannenberg used a beam search to consider multiple hypotheses of the beat phase, utilizing heuristics to select the most likely hypothesis [219]. Large presented a system utilizing an oscillator for generating pulses; the system was able to simultaneously adapt its current tempo estimate and beat phase estimate to the onset times [220]. The adaption speed is based on the distance between the estimated beat position and the actual onset position.

Goto and Muraoka used multiple agents to estimate tempo and beat phase. Each agent has its own hypothesis of the tempo and the beat phase and computes its own reliability by measuring the coincidence of the estimated beat positions with the (extracted) onset positions [208]. Each agent will have different parameter and initialization settings. The agent with the highest reliability is chosen as the one providing the most likely tempo and beat phase estimate. The system has been developed for several years; a predecessor of the system is described in [221]. Later versions of the system extend the computation of the novelty function by detecting changes in the tonal components to get additional information on the salience of onsets and estimated beats [222, 223].

A system which is not based on using discrete onset times has been presented by Scheirer [203]. He computes the envelope in six frequency bands and subjects it to HWR; the six output signals are then used as inputs to a filterbank of closely spaced comb resonance filters. A major difference between this approach and most others is that on the one hand it yields the strength of all detectable tempi and can therefore be used to calculate a beat histogram (see above), but on the other hand the tempo estimate is restricted to the

filterbank's resolution. A similar system based on comb resonance filters has also been used by Klapuri [224].

Dixon's tempo detection system analyzes clusters of all IOIs within a short time window, quantizes them and uses the IOI histogram to find the best tempo estimate [225]. In order to detect the beat phase, multiple competing agents work in parallel with different initializations; the evaluation function for choosing the agent with the most likely beat phase estimate throughout the piece is a measure of regularity of the IBIs and the salience of the chosen onsets [226]. In a related publication he argues that a beat tracking system can benefit from more sophisticated information such as a salience measure based on note duration (or to be more exact, the IOI), intensity, and pitch [227]. Meudic modified Dixon's beat tracking system to work in real-time [228].

Similar to Scheirer, Gouyon and Herrera presented a system utilizing a continuous novelty function as opposed to a series of discrete onset times. The overall novelty function is derived from a set of various features [229]. The tempo is then found by seeking periodicities in the ACF of the novelty function. The choice of the most likely tempo candidate is computed by using a "harmonic grid"; the ACF values of multiples of the current tempo hypothesis are used to estimate its likelihood. A related approach to harmonic processing of the ACF can be found in the fundamental frequency detection system of Karjalainen and Tolonen as described in Sect. 5.3.4.2.

Laroche used a spectral flux-based novelty function and computed the correlation function between a set of quantized template delta pulses for various tempi and beat positions in a window with the length of several seconds [205]. The most salient 10 to 15 maxima are used as initial tempo candidates. Finally, he applies *Dynamic Programming (DP)* techniques to find the most likely overall path through all the tempo candidates for all analysis windows over the whole audio file [for a related algorithm see the description of *Dynamic Time Warping (DTW)* in Sect. 7.1]. A similar approach has been published by Peeters [230]. The main difference to Laroche's approach is that he computes both an ACF and an STFT from the novelty function and combines the results of both to estimate the tempo candidates. The advantage of this combined representation is increased robustness against octave errors, one of the typical problems of periodicity analysis. Furthermore, he adds three different meter templates as possible states to the tempo candidates; his DP approach utilizes the *Viterbi algorithm* [231] to find the most likely tempo path through the "audio file." The advantage of the latter two approaches is that their DP techniques should enable them to deal with sudden changes in tempo and, in the case of Peeters, meter.

6.6 Detection of Meter and Downbeat

The relation of *meter* and *downbeat* is very similar to the relation of tempo and beat phase. Just as the tempo is derived from the distance between two neighboring beats, the meter is (usually) the length of a bar while the downbeat marks the beginning of a bar.

The hierarchical metrical structure of music makes the differentiation of detecting the beat and the downbeat basically a question of the hierarchical level to investigate. Now, we are just interested in long-term periodicities. The algorithms are therefore quite similar; the main differences can be found in the search range and in the computation of the novelty function.

Brown weighted the series of onsets with their IOI (in order to increase the impact of long notes) and computed the ACF of this series of weighted onsets to detect the meter

[232]. Toiviainen and Eerola used a similar approach and evaluated different weighting functions for the IOIs [233].

Uhle and Herre derived bar length candidates from integer multiples of a previously detected tatum and then computed the CCF of two snippets of the novelty function per frequency band to derive a measure of the likelihood of the individual bar length candidates [190].

Escalona-Espinosa argued that it is not only the onset pattern itself that is of interest for the estimation of meter and downbeat but other features should be used for computing the novelty function as well [234]. More specifically, he assumed that in the western tradition of music (and even more so in the case of popular music) the position of a downbeat increases the likelihood of both

- a note or harmony change and
- the occurrence of a new bass note

compared to positions between downbeats. Therefore he proposed the computation of two novelty functions, one based on the pitch chroma difference and the second on the bass energy increase. The time resolution for this computation is signal adaptive: it is the previously estimated tatum. Using the tatum has the two advantages of a signal-related segmentation and higher computational efficiency of the following processing steps. As an alternative to the ACF-based approaches he constructed two matrices which contain the (self-) similarity between all pairs of samples of the two novelty functions. The matrices are called self-similarity matrices. When averaging the diagonals of each similarity matrix the result is a measure of periodicity with respect to the distance from the main diagonal. Depending on the similarity (or distance) measure used for computing the similarity matrix, this function can be closely related to the ACF. The lag of the main peak within a pre-defined search range is then the detected bar length. In combination with a tempo estimate the result allows him to derive the time signature of the piece of music. The most likely downbeat position is estimated with the extracted bar length by computing the CCF of each novelty function with a delta pulse spaced with the bar length period. The lag of the CCF's maximum indicates the downbeat position.

CHAPTER 7

ALIGNMENT

Algorithms for the automatic alignment of sequences of different lengths — *Dynamic Time Warping (DTW)* in particular — are part of various algorithms for the analysis of audio signals. The following chapter gives an introduction to the DTW algorithm and presents the two typical synchronization applications such as audio-to-audio alignment and audio-to-score alignment.

7.1 Dynamic Time Warping

The objective of *DTW* is to align or to synchronize two sequences of different length [235]. Given two sequences $A(n_\mathrm{A})$ with $n_\mathrm{A} \in [0; \mathcal{N}_\mathrm{A} - 1]$ and $B(n_\mathrm{B})$ with $n_\mathrm{B} \in [0; \mathcal{N}_\mathrm{B} - 1]$, a distance measure can be computed between all pairs of $A(n_\mathrm{A})$ and $B(n_\mathrm{B})$, resulting in a *distance matrix* $\boldsymbol{D}_\mathrm{AB}(n_\mathrm{A}, n_\mathrm{B})$. The specific path from $\boldsymbol{D}_\mathrm{AB}(0,0)$ (the start of both sequences) to $\boldsymbol{D}_\mathrm{AB}(\mathcal{N}_\mathrm{A} - 1, \mathcal{N}_\mathrm{B} - 1)$ (the end of both sequences) which minimizes the overall distance is the most likely *alignment path* (also *warping path*) between the two sequences. Figure 7.1 shows two example sequences, the corresponding distance matrix and the resulting alignment path between the two sequences computed with standard DTW as described below.

This alignment path will be referred to as $\boldsymbol{p}(n_\mathrm{P})$ and $n_\mathrm{P} \in [0; \mathcal{N}_\mathrm{P} - 1]$; it is a direct measure of how one sequence has to be warped (scaled in time) to give the best fit to the other sequence. Each path entry is a matrix index in the range $([0; \mathcal{N}_\mathrm{A} - 1], [0; \mathcal{N}_\mathrm{B} - 1])$.

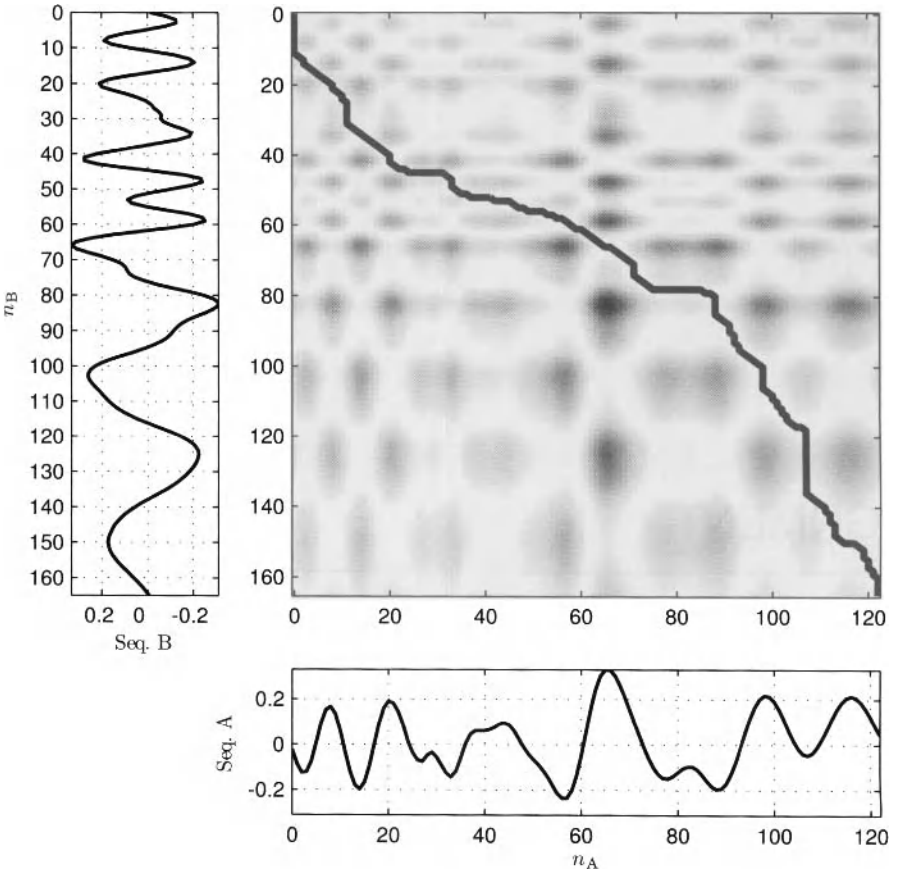

Figure 7.1 Distance matrix and alignment path for two example sequences; dark entries indicate a large distance

The following conditions apply to the path:

- *Boundaries*: the path has to start at the first index of both sequences and has to end at the end of both sequences, meaning that it covers both entire sequences from beginning to end:

$$\boldsymbol{p}(0) = [0,0], \tag{7.1}$$

$$\boldsymbol{p}(\mathcal{N}_P - 1) = [\mathcal{N}_A - 1, \mathcal{N}_B - 1]. \tag{7.2}$$

- *Causality*: the path can only move forward through both sequences, meaning that it is not allowed to "go back in time":

$$n_A\big|_{\boldsymbol{p}(n_P)} \leq n_A\big|_{\boldsymbol{p}(n_P+1)}, \tag{7.3}$$

$$n_B\big|_{\boldsymbol{p}(n_P)} \leq n_B\big|_{\boldsymbol{p}(n_P+1)}. \tag{7.4}$$

- *Continuity*: no index n_A or n_B can be omitted, meaning that the path is not allowed to jump through either sequence:

$$n_A\big|_{\boldsymbol{p}(n_P+1)} \leq (n_A+1)\big|_{\boldsymbol{p}(n_P)}, \tag{7.5}$$

$$n_B\big|_{\boldsymbol{p}(n_P+1)} \leq (n_B+1)\big|_{\boldsymbol{p}(n_P)}. \tag{7.6}$$

These path restrictions result in a theoretical maximum path length of

$$\mathcal{N}_{P,max} = \mathcal{N}_A + \mathcal{N}_B - 2 \tag{7.7}$$

when the path runs along the edges of the distance matrix[1] and a minimum path length of

$$\mathcal{N}_{P,min} = \max(\mathcal{N}_A, \mathcal{N}_B) \tag{7.8}$$

when the path runs on the matrix diagonal for as long as possible.

In order to find the optimal alignment path the concept of "cost" is used. The cost of a path $\boldsymbol{p}_j$ can be computed by accumulating the values of the distance matrix at all path points:

$$\mathfrak{C}_{AB}(j) = \sum_{n_P=0}^{\mathcal{N}_P-1} \boldsymbol{D}(\boldsymbol{p}_j(n_P)). \tag{7.9}$$

The optimal alignment path is then the path that minimizes the overall cost:

$$\mathfrak{C}_{AB,min} = \min_{\forall j}\left(\mathfrak{C}_{AB}(j)\right), \tag{7.10}$$

$$j_{opt} = \operatorname*{argmin}_{\forall j}\left(\mathfrak{C}_{AB}(j)\right). \tag{7.11}$$

The optimal path can thus be found by computing all possible paths through the matrix $\boldsymbol{D}$ and determining the path with the lowest overall cost.

By means of utilizing glsIdxDP techniques this brute force approach may be discarded as the best global solution can be computed more efficiently. As an intermediate result the *cost matrix* $\boldsymbol{C}_{AB}$ is introduced; it has the same dimensions as the distance matrix, but each matrix element contains the accumulated overall cost of the best path to this specific matrix element.

The cost matrix can be computed iteratively by

$$\boldsymbol{C}_{AB}(n_A,n_B) = \boldsymbol{D}_{AB}(n_A,n_B) + \min\begin{cases}\boldsymbol{C}_{AB}(n_A-1,n_B-1)\\ \boldsymbol{C}_{AB}(n_A-1,n_B)\\ \boldsymbol{C}_{AB}(n_A,n_B-1)\end{cases}. \tag{7.12}$$

Figure 7.2 plots both the distance matrix and the corresponding cost matrix for the example sequences from Fig. 7.1.

The first entry of the cost matrix is initialized with

$$\boldsymbol{C}_{AB}(0,0) = \boldsymbol{D}_{AB}(0,0). \tag{7.13}$$

[1]The -2 originates in the automatic avoidance of the corner of the distance matrix.

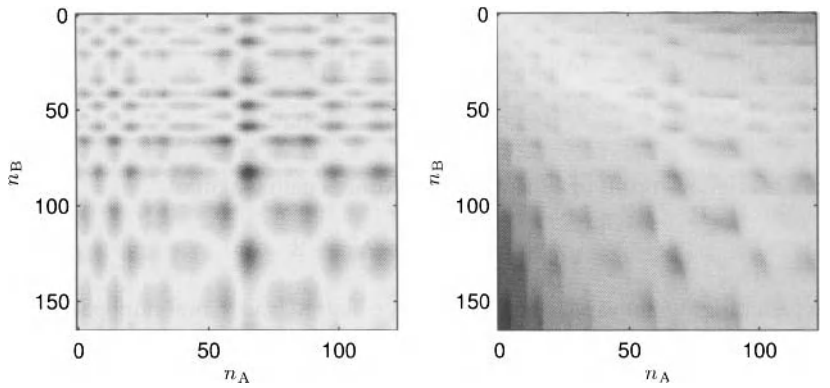

Figure 7.2 Distance matrix (left) and corresponding cost matrix (right)

Due to the alignment path restrictions given above the computation of both the first row and the first column of the cost matrix is trivial:

$$\boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}},0) = \boldsymbol{D}_{\mathrm{AB}}(n_{\mathrm{A}},0) + \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}-1,0), \tag{7.14}$$

$$\boldsymbol{C}_{\mathrm{AB}}(0,n_{\mathrm{B}}) = \boldsymbol{D}_{\mathrm{AB}}(0,n_{\mathrm{B}}) + \boldsymbol{C}_{\mathrm{AB}}(0,n_{\mathrm{B}}-1). \tag{7.15}$$

This can also be interpreted as initializing the distance matrix for indices smaller than 0 with

$$\begin{aligned} \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}},-1) &= \infty, \\ \boldsymbol{C}_{\mathrm{AB}}(-1,n_{\mathrm{B}}) &= \infty, \end{aligned}$$

and applying Eq. (7.12).

Each cost matrix element contains the minimum cost to reach that element. During the calculation of the cost matrix, the indices of the preceding cell that has been selected as the minimum cost for each matrix element have to be remembered. The optimal path can then be traced back from the current element to the beginning.

The complete iterative algorithm can thus formally be summarized by

- *Initialization*:

$$\boldsymbol{C}_{\mathrm{AB}}(0,0) = \boldsymbol{D}_{\mathrm{AB}}(0,0), \tag{7.16}$$

$$\boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}},-1) = \infty, \tag{7.17}$$

$$\boldsymbol{C}_{\mathrm{AB}}(-1,n_{\mathrm{B}}) = \infty. \tag{7.18}$$

- *Recursion*:

$$C_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}) = D_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}) + \min \begin{cases} C_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}-1) \\ C_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}) \\ C_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}-1) \end{cases}, \quad (7.19)$$

$$j = \operatorname{argmin} \begin{cases} C_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}-1) \\ C_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}) \\ C_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}-1) \end{cases}, \quad (7.20)$$

$$\Delta \boldsymbol{p}(n_{\mathrm{A}}, n_{\mathrm{B}}) = \begin{cases} [-1,-1] & \text{if } j = 0 \\ [-1, 0] & \text{if } j = 1 \\ [0,-1] & \text{if } j = 2 \end{cases}. \quad (7.21)$$

- *Termination*:

$$n_{\mathrm{A}} = \mathcal{N}_{\mathrm{A}} - 1 \wedge n_{\mathrm{B}} = \mathcal{N}_{\mathrm{B}} - 1. \quad (7.22)$$

- *Path backtracking*:

$$\boldsymbol{p}(n_{\mathrm{P}}) = \boldsymbol{p}(n_{\mathrm{P}}+1) + \Delta \boldsymbol{p}\big(\boldsymbol{p}(n_{\mathrm{P}}+1)\big), \; n_{\mathrm{P}} = \mathcal{N}_{\mathrm{P}} - 2, \mathcal{N}_{\mathrm{P}} - 3, \ldots, 0. \quad (7.23)$$

Note that the distance matrix is frequently be replaced by a *similarity matrix*; the algorithm will remain the same although the cost matrix (as introduced below) will have to be replaced by a *likelihood matrix* and some algorithmic details will have to be modified in other places as well, for instance, replacing the min operation by a max operation.

7.1.1 Example

In order to allow a better understanding of the initially rather abstract concept of DTW we will present a small example. The (one-dimensional) input sequences are

$$\begin{aligned} A &= [1,\ 2,\ 3,\ 0], \\ B &= [1,\ 0,\ 2,\ 3,\ 1], \end{aligned}$$

and we will simply use the magnitude of the difference as the distance measure. The distance matrix is then

$$\boldsymbol{D}_{\mathrm{AB}} = \begin{bmatrix} 0 & 1 & 2 & 1 \\ 1 & 2 & 3 & 0 \\ 1 & 0 & 1 & 2 \\ 2 & 1 & 0 & 3 \\ 0 & 1 & 2 & 1 \end{bmatrix}, \quad (7.24)$$

and the corresponding cost matrix is

$$\boldsymbol{C}_{\mathrm{AB}} = \begin{bmatrix} 0 & 1 & 3 & 4 \\ 1 & 2 & 4 & 3 \\ 2 & 1 & 2 & 4 \\ 4 & 1 & 1 & 4 \\ 4 & 2 & 3 & 2 \end{bmatrix}. \quad (7.25)$$

During the calculation of the cost matrix, we also memorized the direction of lowest cost for each matrix element:

$$\begin{bmatrix} \cdot & \leftarrow & \leftarrow & \leftarrow \\ \uparrow & \nwarrow & \nwarrow & \nwarrow \\ \uparrow & \nwarrow & \leftarrow & \leftarrow \\ \uparrow & \uparrow & \nwarrow & \leftarrow \\ \uparrow & \uparrow & \nwarrow & \nwarrow \end{bmatrix}$$

in order to be able to backtrack the optimal path through the distance matrix:

$$\boldsymbol{D}_{\mathrm{AB}} = \begin{bmatrix} 0 & & & \\ 1 & & & \\ & 0 & & \\ & & 0 & \\ & & & 1 \end{bmatrix}. \tag{7.26}$$

7.1.2 Common Variants

The standard DTW algorithm as described above is frequently modified; the reasons for this modification can be either the adaption to specific use cases or the optimization of its workload requirements. Several such modifications are described in detail by Müller [236].

7.1.2.1 Transition Weights

In order to favor vertical, horizontal, or diagonal path movement, weighting factors can be applied to Eq. (7.12)

$$\boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}) = \min \begin{cases} \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}} - 1, n_{\mathrm{B}} - 1) & + & \lambda_{\mathrm{d}} \cdot \boldsymbol{D}_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}) \\ \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}} - 1, n_{\mathrm{B}}) & + & \lambda_{\mathrm{v}} \cdot \boldsymbol{D}_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}) \\ \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}} - 1) & + & \lambda_{\mathrm{h}} \cdot \boldsymbol{D}_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}) \end{cases}. \tag{7.27}$$

In the default DTW approach all these weights had been set to 1. Another typical set of weights is

$$\begin{aligned} \lambda_{\mathrm{d}} &= 2, \\ \lambda_{\mathrm{v}} &= 1, \\ \lambda_{\mathrm{h}} &= 1 \end{aligned}$$

to prevent the implicit preference of the diagonal since one diagonal step corresponds to one horizontal plus one vertical step.

An algorithm closely related to DTW is the *Viterbi algorithm* which finds a path through a set of (observed and hidden) states [231].

7.1.2.2 Different Step Sizes

Instead of forcing the algorithm to only increment each index by one, one can optionally allow larger step sizes or jumps. The arguments of the minimum operation of Eq. (7.12)

could, for example, be replaced by

$$\min \begin{cases} \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}-1) \\ \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}-2, n_{\mathrm{B}}-1) \\ \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}-2) \end{cases} \tag{7.28}$$

which constrains the slope of the warping path to avoid the path containing many consecutive horizontal or vertical steps. This modification will only work if the sequence lengths $\mathcal{N}_{\mathrm{A}}$ and $\mathcal{N}_{\mathrm{B}}$ differ not by more than a factor of 2. Another alternative enforcing the alignment of all elements of both sequences is to replace the arguments of the minimum operation of Eq. (7.12) by

$$\min \begin{cases} \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}-1) \\ \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}-2, n_{\mathrm{B}}-1) + \boldsymbol{D}_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}) \\ \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}-3, n_{\mathrm{B}}-1) + \boldsymbol{D}_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}) + \boldsymbol{D}_{\mathrm{AB}}(n_{\mathrm{A}}-2, n_{\mathrm{B}}) \\ \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}-2) + \boldsymbol{D}_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}-1) \\ \boldsymbol{C}_{\mathrm{AB}}(n_{\mathrm{A}}-1, n_{\mathrm{B}}-3) + \boldsymbol{D}_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}-1) + \boldsymbol{D}_{\mathrm{AB}}(n_{\mathrm{A}}, n_{\mathrm{B}}-2) \end{cases} . \tag{7.29}$$

7.1.3 Optimizations

If the two sequences to be aligned are long, the size of the distance matrix increases drastically as the number of matrix elements is the multiplication of the length of the sequences $\mathcal{N}_{\mathrm{A}} \cdot \mathcal{N}_{\mathrm{B}}$. This results in both high memory requirements and a large overall number of operations. The effects can be alleviated by using different approaches to optimization.

One simple approach to reduce the amount of memory without changing the (standard DTW) algorithm or its results is to replace the cost matrix with two vectors, a row and a column vector of cost entries, as the cost of more distant elements is not used by the algorithm.

7.1.3.1 Maximum Time Deviation Constraint

Under the assumption that the timing of the two sequences to be aligned does not deviate more than a certain maximum deviation $T_{\max}$, neither the distances nor the cost has to be computed for matrix entries outside a band with the width of $2T_{\max}$ centered around the diagonal from start $(0,0)$ to stop $(\mathcal{N}_{\mathrm{A}}-1, \mathcal{N}_{\mathrm{B}}-1)$. This optimization has first been proposed by Sakoe and Chiba [235] and is visualized in Fig. 7.3 (left).

7.1.3.2 Maximum Tempo Deviation Constraint

If the slope of the alignment path is constrained or, in other words, the tempo difference between the two sequences is limited then matrix entries outside of a trapezoid spanned by this tempo relationship do not have to be computed. This optimization has first been proposed by Itakura [237] and is visualized in Fig. 7.3 (right).

If start and end points of the two sequences are not necessarily a perfect match, for example, in the case of silence frames at the beginning and end of a sequence, this approach should be combined with the optimization assuming a maximum time deviation (see above) to allow horizontal and vertical movement along the matrix edges at start and end. Otherwise the path restrictions are too narrow if the sequences do not start and stop at exactly the same point. This is a problem of the path shown in Fig. 7.3 (right) at both start and end.

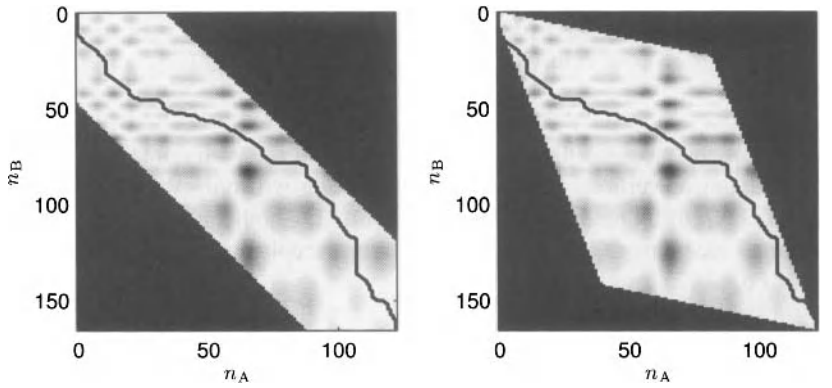

Figure 7.3 Path restrictions for performance optimizations of DTW: maximum time deviation (left) and maximum tempo deviation (right)

7.1.3.3 Sliding Window

The DTW algorithm is not a real-time algorithm as it requires the complete sequences for processing. It is possible to use DTW in a pseudo-real-time context when the alignment path is only computed within a sliding local window [238]. This results in a high algorithmic latency. Outside this window the path will not be updated anymore. The number of operations depends on the window length and is independent of the sequence length.

7.1.3.4 Multi-scale Dynamic Time Warping

Multi-scale DTW is based on the idea of processing the input sequences at different time resolution. More specifically, the time resolution of the input sequences can be reduced by both, using larger block sizes for computing the distances or low-pass filtering and down-sampling the existing sequences. The extracted path through this down-sampled distance matrix can then be used to define a sliding window to be used for a second matrix with finer resolution. This process can be done in two stages [239] or iteratively in multiple stages [236, 240].

7.2 Audio-to-Audio Alignment

Audio-to-audio alignment describes the process of retrieving corresponding points in time in between two audio signals with the same or a similar content. It requires an analysis of the audio files enabling the mapping of points in time in one signal to points in time in the other signal. The knowledge of those synchronization points enables a variety of different use cases such as

- quick browsing for certain parts in recordings in order to easily compare parts auditorily [238, 240],
- adjusting the timing of one recording to that of a second by means of a dynamic time stretching algorithm. This has practical use in a music production environment. The

different voices of a homophonic arrangement (e.g., the backing vocals in a pop song) can, for instance, be automatically synchronized to the lead voice. The same applies for different instruments playing in unison or at least in the same rhythm,

- automated synchronization of a (dubbed) studio recording with an original, possibly distorted, recording, and
- musicological analysis of the timing information contained in the alignment path of several performances of the same score in order to compare the tempo and timing to a given reference music performance.

The main difference between various publications on audio-to-audio alignment can be found in the features extracted from the audio signal as well as in the subsequent computation of the distance matrix. Differences in calculating the alignment path are mainly due to the variants and optimizations of DTW listed above.

Hu and Dannenberg compute a simple pitch chroma as input feature [241]. The distance is the Euclidean distance between all pitch chroma pairs of the two "audio" sequences.

Turetsky and Ellis use several STFT-based features such as the power and the first order difference in time and frequency. They then use the cosine distance as similarity measure [239].

Dixon and Widmer argue that the main objective of audio-to-audio alignment is the synchronization of the onset times and propose to use a spectral flux-based feature subjected to HWR; the feature is computed in semi-tone bands [238]. The distance measure is the Euclidean distance. In order to achieve pseudo-real-time alignment, they use a modified DTW approach that estimates the optimal local path within a sliding window (see Sect. 7.1.3.3).

Müller et al. compute the pitch chroma via the energy outputs of a filterbank. They use multi-scale DTW as described in Sect. 7.1.3.4 to efficiently compute the alignment path [240].

Kirchhoff and Lerch pointed out that the type of features used for audio-to-audio alignment depend on the use case; the alignment may be done for signals with either identical pitch, envelope, or timbre while onset-related features can be used in any case [242]. They evaluated a large number of features from these four categories and did not use a vector distance but trained an LDA classifier for two classes (*on path* vs. *not on path*) for the automatic feature weighting and the distance computation.

7.2.1 Ground Truth Data for Evaluation

A set of pairs of audio files with clearly defined synchronization points is required as ground truth to evaluate the accuracy of an estimated alignment path. There exist several ways of generating a ground truth data set for audio-to-audio alignment systems. Corresponding points in time can be (manually or semi-automatically) annotated, onset times can be monitored and stored during a recording of a computer-monitored instrument, or MIDI files can be rendered to audio by means of a sample player. As pointed out in Sect. 6.3.3.2 the manual annotation of points in time is a rather arduous process and thus generally only a few points per test file can be labeled. The drawback of using piano-generated data is its restriction to solo piano music; furthermore, confining the evaluation to note onsets might be sufficient for the case of solo piano music, however, other kinds of music may also require the synchronization during the time span in between onsets. The disadvantage of using synthesized samples is that their properties might differ from "real-world" signals.

Alternatively, it is possible to artificially generate modified pairs of audio signals by using a dynamic time stretching algorithm, i.e., an audio processing algorithm able to change the tempo without changing the pitch of an audio signal [243]. This allows for high accuracy of the data set while allowing to test with a wide range of musical styles and instrumentations. In this case the validity of the data set depends (a) on the realistic dynamic use of stretch factors and (b) on the audio quality of the stretching engine. Furthermore, the test data originates from the same audio signal which may give too positive results.

7.3 Audio-to-Score Alignment

Systems for the synchronization of an audio signal with a musical score (frequently in MIDI format) are usually categorized with respect to their real-time capabilities. Real-time systems are called *score following* systems, and non-real-time (or offline) implementations are referred to as *audio-to-score alignment* systems.

Possible applications of such alignment systems (compare [244]) include

- linking notation and music performance in applications for musicologists to enable working on a symbolic notation while listening to a real performance,
- using the alignment cost as a distance measure for finding the best matching document in a database (i.e., retrieve an audio signal for a score query or vice versa),
- the musicological comparison of different performances,
- automatic accompaniment systems,
- the construction of a new score describing a selected performance by adding information on dynamics, mix information, or lyrics, and
- musical tutoring or coaching system for which the timing of a recorded performance is compared to a reference performance.

7.3.1 Real-Time Systems

Historically, the research on matching a pre-defined score automatically with a music performance goes back to the year 1984. At that time, Dannenberg and Vercoe independently presented systems for the automatic (computer-based) accompaniment of a monophonic input source in real time [245, 246].

In the following years, Dannenberg and Bloch enhanced Dannenberg's system by allowing polyphonic input sources and increasing its robustness against musical ornaments and by using multiple agent systems [247, 248]. Vercoe focused on the implementation of learning from the real performance to improve the score follower's accuracy [249].

Baird et al. proposed a score following system working with MIDI input (for the performance) based on the concept of musical segments as opposed to single musical events; the tracking algorithm itself is not described in detail [250, 251].

Heijink and Desain et al. presented a score following system that takes into account structural information as well. It uses a combination of DP and strict pitch matching between performance and score [252, 253].

While many of previously presented publications focus on the score following part rather than audio processing itself, Puckette and Lippe worked on systems with audio-only input with monophonic input signals such as clarinet, flute, or vocals [254, 255].

Vantomme proposed a monophonic score following system that uses temporal patterns from the performer as its primary information [256]. From a local tempo estimate the next event's onset time is predicted and the distance between expected onset time and extracted onset time is evaluated. In the case of an "emergency," he falls back to the use of pitch information.

Grubb and Dannenberg presented a system following a monophonic vocal performance. It uses the fundamental frequency, spectral features, as well as amplitude changes as features for the tracking process [257, 258]. The estimated score position is calculated based on a PDF conditioned on a distance computed from the previous score event, from the current observation, and from a local tempo estimate.

Raphael published several approaches to score following implementing probabilistic modeling and machine learning approaches incorporating *markov models* [259–261].

Cano et al. presented a real-time score following system for monophonic signals based on an HMM [262]. They used the features zero crossing rate, energy, and its derivative plus three features computed from the fundamental frequency.

Orio et al. introduced a score following system for polyphonic music which utilizes a two-level HMM modeling each event as a state in one level, and modeling the signal with attack, sustain, and rest phase in a lower level [263, 264]. They use a so-called *Peak Structure Distance (PSD)* that represents the energy sum of band-pass filter outputs with the filters centered around the harmonic series of the pitch of the score event under consideration.

Cont presented a polyphonic score following system using hierarchical HMMs using previously learned pitch templates for multiple fundamental frequency matching [265].

7.3.2 Non-Real-Time Systems

While the publications presented above deal with score following as a real-time application, the following publications deal with the closely related topic of non-real-time audio-to-score alignment.

The importance of reliable pattern matching methods has already been recognized in early publications on score following and alignment; in most cases DP approaches have been used [266]; see, for example, Dannenberg's publications on score following mentioned above.

Orio and Schwarz presented an alignment algorithm for polyphonic music based on DTW which combined several local distances (similarity measures) [267]. It uses the PSD as described above and a *delta of PSD* (ΔPSD) modeling a kind of onset probability; it also uses a silence model for low-energy frames [263].

Meron and Hirose proposed a similar approach with several easy-to-compute audio features and suggested post-processing of the alignment results to improve the alignment robustness [268].

The system by Arifi et al. attempts to segment the audio signal into (polyphonic) pitches and performs DP to align MIDI file to the extracted data [269, 270]. The algorithm has been tuned for polyphonic piano music.

Turetsky and Ellis avoided the problems of calculating a spectral similarity measure between score and audio by generating an audio file from the (reference) MIDI file and aligning the two audio sequences with audio-to-audio alignment [239]. For the alignment itself a DP approach is used.

Similarly, Dannenberg and Hu generated an audio file from the MIDI file to align two audio sequences [241, 271]. They calculate the distance measure based on a 12-dimensional pitch chroma. The alignment path is then calculated by a DP approach.

Shalev-Shwartz et al. presented a non-real-time system for audio-to-score alignment utilizing DP [272]. Their algorithm features a training stage for the weighting the features for the distance measure. They derived a confidence measure from audio and MIDI similarity data and trained a weight vector for these features to optimize the alignment accuracy over the training set. The audio feature set contains simple pitch-style features extracted by band-pass filtering, derivatives in spectral bands to measure onset probability, and a time deviation from the local tempo estimate.

The alignment system of Müller et al. is also based on DP [273]. It is targeted at piano music, but they claim genre independence. For the pitch feature extraction, they used a (zero phase) filterbank with each band-pass' center frequency located at a pitch of the equally tempered scale; the filter outputs are used to extract onset times per pitch.

In summary, the standard approach to audio-to-score alignment consists of three major processing steps: first, the audio feature extraction which in most cases approximates a pitch-like representation with onset information; timbre and loudness are too performance specific to be used for audio-to-score alignment. Second, a similarity or distance measure between audio and symbolic (score) features is computed. Finally, the actual alignment or path finding algorithm that is either based on DP, DTW, or on HMMs is applied. With respect to the distance measure, two distinct approaches can be identified. On the one hand the score is transformed into a more audio-like representation, ultimately by directly rendering the MIDI signal into an audio signal; on the other hand the audio signal is converted into a more score-like symbolic format, ultimately by transcribing the signal completely. A first step toward combining these two approaches has been made by Lerch by computing two distance matrices for the two approaches and combining them in a weighted average [274].

CHAPTER 8

MUSICAL GENRE, SIMILARITY, AND MOOD

The automatic recognition of the musical genre or the *musical style* from an audio signal is one of the oldest subjects of ACA and can be regarded as one of the key areas leading to today's research field MIR.

The classification into musical genre can in some ways be seen as a special case of a more generalized music similarity measure in the sense that the similarity measure in musical genre classification is restricted to dimensions which are meaningful for genre. The signals are sorted into pre-defined similarity clusters, the musical genres.

Applications for these technologies can be found mainly in the annotation, sorting, and retrieval of (related) audio files from large databases or the Internet. For music similarity there are also somewhat more creative use cases such as the automatic generation of playlists and the generation of so-called mash-ups, mixes of two or more "compatible" songs.

This chapter will cover musical genre classification, music similarity measures, mood classification as well as instrument recognition. All these systems have in common that they try to represent a property (such as the genre) of the audio signal as either a feature vector or a (time-) series of feature vectors; these feature vectors are then used either for classification or for a distance measure to retrieve the required result.

8.1 Musical Genre Classification

The first publications on the automatic recognition of musical style appeared in the 1990s; at that time, the focus was mainly on the discrimination of speech and music signals [275–

An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics, First Edition. Alexander Lerch.

279], although algorithms aiming at classifying a broader range of signals can be found during that decade as well [280, 281]. In a way, the task of automatic musical genre classification is a classic machine learning task — suitable features are extracted from the audio signal and with these features a classification system is trained and used for the classification task.

As pointed out above, the classification into musical genre can be interpreted as a measure of music similarity restricted to certain genre-defining dimensions and categorized to pre-defined classes, the genres. Similarity can in principle have other dimensions as well which are unimportant for the genre definition (see Sect. 8.2.1.1).

General surveys of approaches to musical genre classification have been published by Scaringella et al. and Fu et al. [282, 283].

8.1.1 Musical Genre

At first glance the meaning of the term *musical genre* appears to be self-explanatory and its intuitive definition obvious. On a more thorough investigation, however, it has to be concluded that an objective definition of the term is hardly possible. Some of the reasons for this have been summarized by Pachet and Cazaly [284] and later by Scaringella et al. [282]:

- *Scope of the genre label*: Can an individual song be classified into a genre or does the context of album and performing artist influence or overrule the classification decision? Or may even different parts of a piece of music have different genres?
- *Non-agreement of taxonomies*: The number and definition of genre labels can strongly vary; in the year 2000 Pachet and Cazaly compared genre taxonomies from the three web sites Allmusic, Amazon, and MP3.com. The overall number of genres varied from 430 (MP3.com) to 531 (Allmusic) to 719 (Amazon), and these labels had only 70 terms in common. Furthermore, the hierarchy of the taxonomies differed.

  In practical applications of ACA the number of genres has to be more restricted due to technical limitations of the classification systems used, but the underlying problems of defining a taxonomy remain the same. Figure 8.1 shows two taxonomies defined in the context of early musical genre classification systems.
- *Ill-defined genre labels*: There is a semantic confusion between genre labels. They can be geographically defined (*Indian music*), related to an era in music history (*baroque*), refer to technical requirements (*barbershop*), the instrumentation (*symphonic music*), or usages (*Christmas songs*).
- *Scalability of genre taxonomies*: A specific genre may be split into a variety of subgenres (e.g., *Hip Hop* into *Gangsta Rap*, etc.). The number of subgenres might evolve over the years.
- *Non-orthogonality of genre categories*: One piece of music can possibly be sorted into multiple genre categories at the same time.

Given these observations it becomes clear that deriving a complete and musicologically consistent taxonomy is practically impossible [218]. Without a clear definition of the term *musical genre* it also comes as no surprise that the performance of humans annotating pieces with genre labels is far from perfect [285, 286]. Automatic systems can be hardly expected to outperform humans at this task.

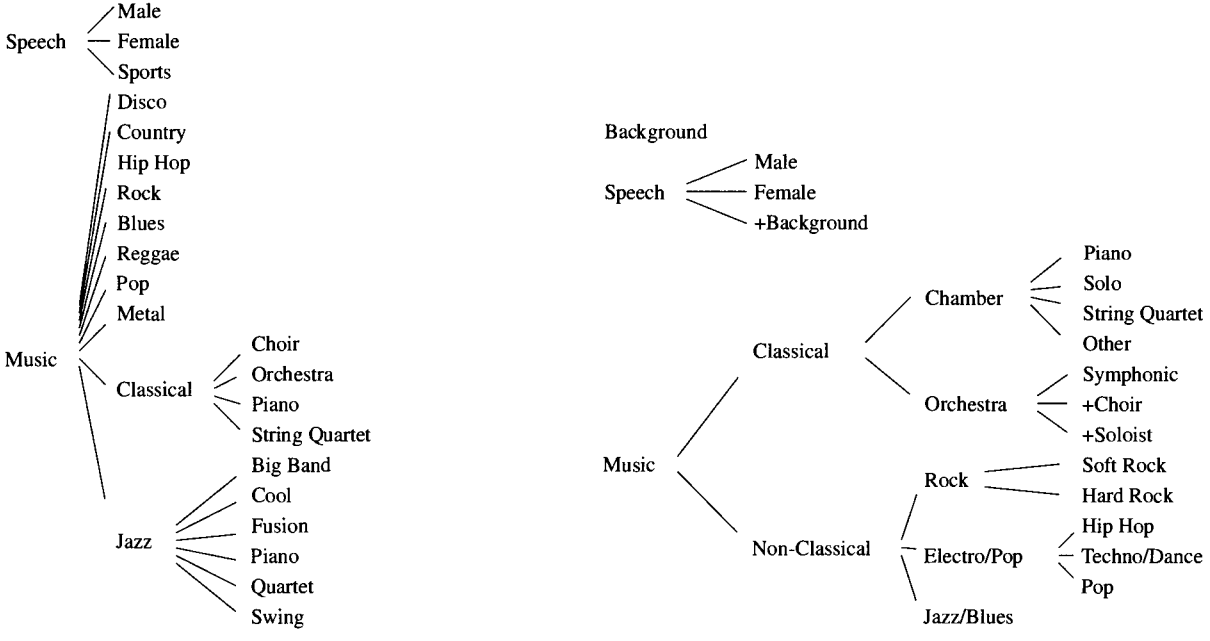

Figure 8.1 Two examples for author-defined genre taxonomies. Left: Tzanetakis and Cook [60], Right: Burred and Lerch [218]

8.1.2 Feature Extraction

From a perceptual point of view, musical genre categorization may be influenced by

- *temporal characteristics* such as the tempo, the time signature, and rhythmic patterns,
- *dynamic characteristics* such as the loudness range, the change of loudness over time, accents,
- *tonal characteristics* such as melodic properties, the complexity of harmony (progression), and prominent pitch classes,
- *production-related characteristics* such as a specific sound quality and volume relations between instruments, and stereo panning properties, and
- *instrumentational characteristics* such as the number and type of instruments used.

If for simplicity's sake the latter two are grouped into a timbre category, the resulting categories are basically tonal, temporal, intensity, and timbre (plus technical) signal properties which were introduced in Sect. 1.1.

The early audio classification systems utilized only a restricted set of features; they mainly used the zero crossing rate (see Sect. 3.4.3) and intensity-related features (see Chap. 4) [275, 276, 287, 288], although timbre-related spectral features quickly got added to the set of features [280, 281, 289]. Pitch-related features which included properties of the fundamental frequency variation could also be found in the early speech/music classification systems [278, 290, 291] but are usually not used for systems targeting polyphonic audio input.

Over the years the number of features grew more and more, eventually covering more or less all features presented in Chap. 3 and adding even more instantaneous features [61, 292].

The most common features remained related to intensity and timbre but, although the classification results with these features seem to be surprisingly good, other feature dimensions got added to the set of features. These additional features include temporal and rhythmic features derived from a beat histogram (see Sect. 6.4) [60, 218, 293], simple tonal features such as pitch histogram features [157], and stereo panning features [294].

8.1.2.1 Texture Window

The instantaneous features, i.e., the features which do not require a long time window such as histogram-based features, are usually processed per *texture window* as described in Sect. 3.5 (low-pass, derivative, subfeatures, etc.). A classification decision can be made for each texture window.

The impact of the texture window length on the classification accuracy has been studied by Tzanetakis and Cook [60], Burred and Lerch [218], and Ahrendt et al. [295] with different results: Tzanetakis and Cook found that the classification accuracy does not improve after a texture window length of 1 s, Burred and Lerch identified a texture window of approximately 15 s to yield satisfactory results, and Ahrendt et al. found a window length of 5 s to be sufficient in most cases. No conclusions can be drawn from this result except that the optimal texture window size depends strongly either on the features and subfeatures used or on the test data set and its categories.

It is worth noting that humans seem to excel at quickly identifying the musical genre; in a data set with 10 genres test subjects were able to identify the genre reliably after listening to audio snippets significantly shorter than 1 s [296].

A hierarchical form of the texture window approach is the default feature extraction mode in the software environment *Marsyas*[1] [297]: statistical subfeatures are computed for the texture window but are not directly used as classifier input but are in turn subjected to the computation of statistical subfeatures (or "*sub*subfeatures") in a longer "super"-texture window.

8.1.3 Classification

From a machine learning point of view it is obvious that any arbitrary classifier can be trained on and applied to the previously extracted features. The differences between the different approaches can be mainly found in the choice of the classifier type and in the genre categorization, mainly through the number of classes but also through the definition of a flat or a hierarchical genre tree. The latter has the advantage of branch-specific feature weighting at the cost of multiple classification steps [218].

Over the years basically all known classification approaches have been evaluated for musical genre classification. The most common are

- the *K-Nearest Neighbor (KNN)* classifier which evaluates the number of the closest training examples in the feature space,
- the *Gaussian Mixture Model (GMM)* which models the classes' feature distribution with multiple Gaussians,
- *Artificial Neural Networks (ANNs)* which are computational models inspired by the structure of biological neural networks [127], and finally
- *Support Vector Machines (SVMs)*, state-of-the-art classifiers transforming the features into a high dimensional space and finding the optimal separating hyperplane between the closest data points [298]; SVMs can nowadays be considered a standard tool in musical genre classification.

A typical way of assessing classification accuracy is *N-fold cross validation.* It is a method to ensure that training set and test set are not identical in order to avoid unrealistically good evaluation results. A standard value for N would be 10, meaning that the data is partitioned into 10 subsets. Of these subsets, 9 are used to train the classification system and the remaining subset is used for testing. This process is repeated until all 10 subsets have been used once for testing, and the overall performance can be estimated by averaging the 10 individual results. Leave-one-out cross validation is an extreme case where N equals the number of data observations minus 1.

Evaluation results of the classification performance strongly depend on the number of classes and the diversity, noisiness, and other general properties of the test set. The MIREX results indicate a classification accuracy of 50–80% for state-of-the-art systems and 10 genre categories. It is self-evident that a 2-class system will yield better results than, for instance, a system with 20 classes. During the first few years of evaluating musical genre classification systems, the aspect of having several songs by the same artist in the same class was neglected, leading to an artist-related training and comparably high classification performance [299].

[1]Marsyas. http://http://marsyas.info. Last retrieved on Dec. 1, 2011.

8.2 Related Research Fields

There are fields of MIR that share so many similarities with musical genre classification that from a engineer's point of view they can be summarized here rather then being individual chapters. Besides a general music similarity detection these technologies include mood classification, instrument recognition, and artist identification. While there exist perceptual and musicological differences between such systems, the technical approaches to solving those problems are closely related as they use similar feature sets and similar classification systems. In the following, we will focus on instructions to music similarity detection, mood classification, and instrument recognition.

8.2.1 Music Similarity Detection

As pointed out above, *music similarity detection* is similar to musical genre classification since the genre itself is a grouping of songs with similar acoustic or musical properties. From this point of view, music similarity detection differs from musical genre classification in the replacement of the classification itself by a distance or similarity measure or by a grouping rule.

Reducing music similarity to genre similarity, however, would be too simplifying since genre similarity is a subset of music similarity. This complicates the definition and evaluation of the latter even more than the definition and evaluation of musical genre. Musical genre classification can at least have a somewhat verified ground truth generated, for example, by manual annotation and categorization of music databases; there can be no such database for similarity measures as long as the term *music similarity* should mean more than just *genre similarity*. This is due to the multi-dimensional and probably associative character of music similarity; since the meaning of music similarity largely depends on both the individual user and the task at hand, this problem probably cannot be solved systematically in general. There remains a gap between what music psychologists and empirical musicologists know to be the music similarity and the simplified similarity definitions that signal processing and machine learning experts attempt to train.

8.2.1.1 Music Similarity

The similarity of two pieces of music can have many facets, and there is research on the number and characteristics of individual perceptual dimensions of music similarity. In the following, only a few publications will be named to exemplify the different approaches and the number of dimensions. Two pieces of music may, for example, be similar with respect to rhythm [198, 300], structure [301], surface and texture [302], melody and motives [303, 304], harmony [305], as well as with respect to performance attributes such as articulation, tempo variation, and dynamics [305, 306]. MacAdams et al. categorized these types of similarity into three clusters, surface and texture, figural, and structural [307]. There are indications that subjective music similarity ratings depend amongst other things on the familiarity of the test subject with the music [308]. They might also depend on the listener's expert level as well.

The associative nature of memory cannot be neglected in real-world applications as editorial data will also be influencing a human similarity decision. Editorial data refers to data that cannot be extracted from the audio signal such as recording date and studio, artists and producers who participated in other albums, the label, etc.

From an application developer point of view it is also possible to define music similarity on a technical level; one may, for instance, simply look for songs with the same tempo or related musical key.

8.2.1.2 Features

The type and number of features used in music similarity detection is closely related to those for musical genre classification. The early systems utilize very small (timbre-related) feature sets such as MFCCs [281, 309–311], then other features such as loudness-related and rhythm-related features are added [58, 312], and nowadays basically all low-level features and mid-level features introduced in the preceding chapters are investigated for music similarity detection [313, 314].

The representation of similarity features per texture window or recording differs in some cases from the statistical subfeatures known from musical genre classification. Logan and Salomon use a K-means clustering algorithm to find a good average description of the spectral envelope with MFCCs [309], Aucouturier and Pachet model this spectral envelope with GMMs [311], and Pampalk et al. use a histogram containing level classes per frequency band [58].

8.2.1.3 Similarity Measure

The pieces of music are represented as (normalized) feature vectors. The simplest approach to a similarity measure is to compute a simple vector distance between those feature vectors. Typical pairwise distances would be the Euclidean distance, the Manhattan distance, and the cosine distance.

Instead of computing the pairwise distance it is common to either automatically group the vectors or to map them to a lower-dimensional space by means of unsupervised machine learning algorithms. Two examples of such methods are

- *K-means clustering*: K-means clustering aims at clustering the vectors into K groups by minimizing the intra-cluster variance. The standard approach to K-means clustering has the following algorithmic steps:
  1. *Initialization*: randomly select K vectors from the data set as initialization.
  2. *Update*: compute the mean for each cluster.
  3. *Assignment*: assign each observation to the cluster with the mean of the closest cluster.
  4. *Iteration*: go to step 2 until the clusters converge.
- *Self-Organizing Map (SOM)*: A SOM is a form of an ANN which produces a two-dimensional map as a representation of the (higher dimensional) training samples. In its simplest form, it features the following algorithmic steps:
  1. *Initialization*: specify (randomly or deterministically) a set of nodes spanning a net. Each node is represented by a weight vector with the same dimension as feature vectors.
  2. *Update*: pick a training sample (feature vector) and compute the distance to all nodes. Update the weight vectors of nodes at a close distance to the training sample to move them toward the training vector. The amount of increment depends on the proximity of the node and the training sample and possibly decreases with an increasing number of iterations.
  3. *Iteration*: go to step 2 until the maximum number of iterations is reached.

8.2.2 Mood Classification

The mood of a piece of music is one of the cues a typical user finds helpful in finding and browsing music [315]. This has lead to an increasing amount of research targeted at automatically recognizing the emotional characteristics of recordings of music. Terms for this field are, for instance, (audio) *mood classification* and *music emotion recognition*.

8.2.2.1 Emotion and Mood in Music

Understanding the meaning of the terms *emotion* and *mood* seems to be essential for the successful design of mood classification systems. Unfortunately, there is no established understanding of what emotion and mood actually are. This is true not only in the context of music but also in general. Kleinginna and Kleinginna, for example, reviewed more than a hundred scientific definitions of emotion [316] without being able to identify a consensus. There might be better or worse definitions but in the end it is just not possible to prove the correctness of an individual definition.

Weld described the difference between emotion and mood by characterizing emotion as temporary and evanescent in contrast to mood which is more permanent and stable [317]; mood can also be seen as a diffuse affect state which can emerge without apparent cause [318]. However, whether the term *emotion* or the term *mood* is more fitting in a musical context is unclear.

Researching the relation of emotion and music exposes some of the typical problems in psychological research; one problem is, for example, the process of verbalization which confines the description of subtle and varied emotional states to the standardized words used to denote them [319]. Meyer also points out that descriptions of emotions are usually apocryphal and misleading since emotions are named and distinguished largely in terms of the external circumstances in which the response takes place; music itself, however, presents no external circumstances [319]. Scherer criticizes the tendency to assume that music evokes basic emotions such as anger, fear, etc. [320]. This led him to propose the differentiation between *utilitarian* emotions, which are the emotions usually studied in emotion research (anger, fear, joy, disgust, sadness, shame, guilt, etc.), and *aesthetic* emotions which are not driven by external influences or personal goals but rather by the appreciation of the intrinsic qualities of a work of art [321]. Zentner et al. found that negative emotions such as guilt, shame, disgust, anger, fear, etc. are practically never aroused by music [322].

Recent research indicates that the "description of musical emotions requires a more nuanced affect vocabulary and taxonomy than is provided by current scales and models of emotion" [322].

It is of importance to distinguish between the emotion aroused in the listener and the conveyed emotion perceived by a listener without particularly feeling it [319]. Ratings of *perceived* emotion differ significantly from ratings of *felt* emotion [322].

To complicate matters further, it might be of interest to differentiate between score-inherent emotions and performance-inherent emotions. To study this, however, seems to be only possible in very controlled environments [323–325]; a real-world scenario does not allow for such a differentiation as the performance is an integral part of the "music" (see also Chap. 10). Also, the differentiation between score and music performance is mainly made for non-popular or classical music as opposed to popular music.

When assessing the mood of a musical piece, the usual approach is to rely either on models or on label categories. Russel's two-dimensional emotion space is one of the frequently

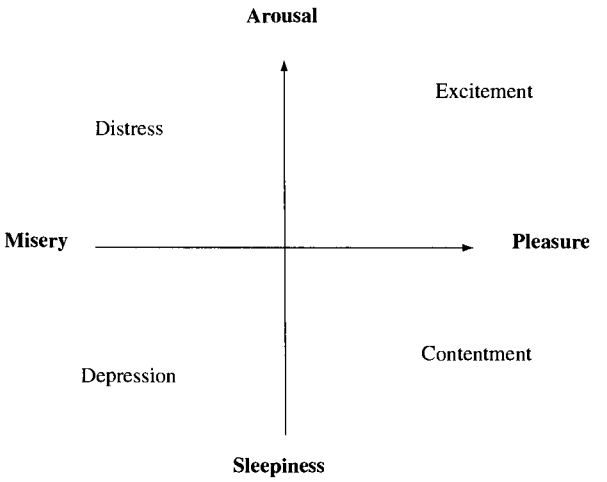

Figure 8.2 Russel's two-dimensional model of mood

used models [326]. It describes mood by the two dimensions *pleasure-misery* (horizontal) and *arousal-sleepiness* (vertical) and is displayed in Fig. 8.2.

A third dimension (for example, *dominance* or *interest*) is sometimes added to this model, but the usefulness of this dimension is less obvious than with the first two dimensions [327, 328].

Defining a set of mood categories or clusters is a way to reduce the complexity of the model significantly. Based on preceding reasearch on measuring emotional response to music, Schubert grouped mood labels into nine clusters as shown in Table 8.1 [329]. Deriving mood clusters by statistical analysis from large publicly available meta data collections has been done by Hu and Downie [330]. The resulting five clusters as shown in Table 8.2 are also used in the MIREX automatic mood classification task.

In research on music performance, there is strong evidence of a relation between moods and both the tempo and the loudness of the performance, as reported by Juslin [323], Kantor [331], Sloboda and Lehmann [332], Schubert [333], and Timmers et al. [334]. The mode (major vs. minor) has also been reported to have influence on the mood [335].

8.2.2.2 Recognition

The features and classification algorithms used for mood classification are very similar to the ones used in musical genre classification. Some mood-specific features which are not common in musical genre classification are articulation-based features estimating the smoothness of "note transitions" [336]. Mood classification systems also use tempo and rhythm-related features more frequently than systems for musical genre classification [337–339].

While the features for detecting the mood of music are relatively similar among researchers, the models and mood categories vary. Feng classifies music into the four classes *happiness*, *anger*, *sadness*, and *fear* [337], and Li uses the three dimensions *cheerful-depressing*, *relaxing-exciting* and *comforting-disturbing* [313]. Frequently used is Russell's two-dimensional arousal/valence-model as shown in Fig. 8.2 [326]; sometimes this model is simplified to define every quadrant as a category or cluster (*contentment*, *depression*,

Table 8.1 Mood Clusters as presented by Schubert

Cluster 1	*Cluster 2*	*Cluster 3*	*Cluster 4*	*Cluster 5*	*Cluster 6*	*Cluster 7*	*Cluster 8*	*Cluster 9*
Bright	Humorous	Calm	Dreamy	Dark	Heavy	Tragic	Agitated	Dramatic
Cheerful	Light	Delicate	Sentiment	Depressing	Majestic	Yearning	Angry	Exciting
Happy	Lyrical	Graceful		Gloomy	Sacred		Restless	Exhilarated
Joyous	Merry	Quiet		Melancholy	Serious		Tense	Passionate
	Playful	Relaxed		Mournful	Spiritual			Sensational
		Serene		Sad	Vigorous			Soaring
		Soothing		Solemn				Triumphant
		Tender						
		Tranquil						

Table 8.2 Mood clusters derived from meta data and used in MIREX

Cluster 1	*Cluster 2*	*Cluster 3*	*Cluster 4*	*Cluster 5*
Rowdy	Amiable/Good Natured	Literate	Witty	Volatile
Rousing	Sweet	Wistful	Humorous	Fiery
Confident	Fun	Bittersweet	Whimsical	Visceral
Boisterous	Rollicking	Autumnal	Wry	Aggressive
Passionate	Cheerful	Brooding	Campy	Tense/Anxious
		Poignant	Quirky	Intense
			Silly	

exuberance, *anxious/frantic*) [338, 340]. By using continuous values for arousal and valence a categorical taxonomy can be avoided and each piece of music can be represented by a point in the two-dimensional space [341, 342]. Other options are to use a fuzzy classification that assigns probabilities to each class [340] and a multi-class or multi-label classification system which assigns a group of labels to a test sample [313, 338, 343].

Two additional aspects have also been the subject of research for mood classification; first, the possible mood change with the temporal evolution of music leading to the requirement of a non-stationary approach to mood classification [338, 344] and, second, the personalization of the recognition systems allowing to model the emotional concepts or responses of individuals or groups [339].

The evaluation of mood classification systems is — due to the fuzzy and subjective nature of mood — even more complicated than it was in the case of musical genre classification. An overview of different approaches to generating ground truth data is given by Kim et al. [345].

For the five mood clusters defined for the MIREX evaluation, the mood classification accuracy ranges between 40 and 60%. A recent study by Huq et al. presents evidence that with the currently used features and classification approaches further improvement of classification accuracy is questionable [342].

8.2.3 Instrument Recognition

An algorithm for *instrument recognition* attempts to identify the musical instruments which compose a sound or are present in a musical recording. In contrast to the classification into genres or moods it is straightforward to find ground truth data for training and evaluation even if the problem of defining instrument taxonomies cannot be considered to be ultimately solved either [346].

The two basic forms of instrument recognition can be distinguished by their type of input signal; some systems require a single note with no other pitches or instruments present; the signal has to be properly edited to eliminate preceding or succeeding notes or noises. Other systems work on a complex mixture of different instruments and estimate the (number and) type of the instruments present in the mixture.

Since the latter case is obviously algorithmically harder to handle, it comes as no surprise that most of the early publications on instrument recognition work on monophonic snippets of sound containing only one note. Herrera et al. give a good survey on the literature on monophonic instrument recognition [347].

The restriction to short monophonic input signal snippets allows the definition and usage of a new specialized feature set that extends the set of features introduced in Chap. 3. More specifically, it allows the system to use features that cannot be used in the context of polyphonic music until it will be possible to separate the sources into individual monophonic subsignals. The additional features can be structured in two categories:

- *Temporal envelope features* are features describing the temporal evaluation of the sound. Simple examples are the sound's duration, its temporal centroid, and the (logarithmic) attack time.
- *Pitch-based features* utilize the fundamental frequency of the sound. Examples include the energy ratio of even and odd harmonics, the inharmonicity of the harmonics, and the onset asynchrony between harmonics.

Kaminskyj et al. presented one of the first systems for automatic instrument recognition [348, 349]. It used a short time RMS and some harmonicity and spectral onset asynchrony features; the classification is done with either an ANN or a nearest-neighbor classifier. Similar to the systems for music similarity and musical genre classification, cepstral coefficients and MFCCs are frequently used for automatic instrument recognition, for instance, by the systems presented by Brown [350] and Marques and Moreno [351]. With time, the number of features and the diversity of instrument classes increased, while as classification approaches basically the same systems KNN, ANN, GMM, and SVM are used [352–355].

The next level in the history of instrument recognition was reached when the input signals did not have to be individual notes anymore; while the input still needed to be monophonic, it could now contain phrases and whole melodies [356–362].

There are only a few systems estimating the instrumentation of polyphonic signals. Eggink and Brown presented a system based purely on spectral features with a GMM-based classification that automatically masks out temporary "unreliable" features [363–365]. Eisenberg proposed a system for detecting instruments using a so-called *harmonic peak spectrum* by modeling instrument sounds with harmonic sinusoidal peaks [35]. It extracts the most salient component in the input signal so that it usually detects only the most prominent instrument; according to Eisenberg the system is able to detect accompanying instruments during pauses of the solo instrument. Heittola et al. attempt to decompose the signal into a sum of spectral bases and detect the individual sound sources [366]. The classification is done with GMMs on MFCCs.

CHAPTER 9

AUDIO FINGERPRINTING

Fingerprinting aims at identifying audio recordings in a previously generated database. More specifically, each recording is represented by a fingerprint, a unique and compact digest summarizing the (perceptually) relevant aspects of the recording. The fingerprint is also referred to as *perceptual hash*. A database containing previously extracted fingerprints can be used to identify an unknown recording. In contrast to most of the other systems presented in the book, fingerprinting does not attempt to extract musical properties from the audio signal but aims at identifying a specific recording as opposed to a specific song. Different music performances (or recordings) of the same song should therefore have different fingerprints. However, a recording still has to be identified when subjected to quality degradation such as perceptual audio coding, added noise, distortions, and other typical signal manipulations.

There are two main areas of application: *broadcast monitoring* allows rights holders the verification of paid royalties and end consumer apps enable the user to either easily identify music or to make use of added value services offering extra information for a song such as the album cover image or tags and other meta data of interest. Cano et al. give a detailed overview of various other applications for which fingerprinting can be of use [367].

Fingerprinting is not to be confused with *watermarking*; the latter embeds a perceptually unnoticeable data block directly in the audio data, utilizing methods similar to perceptual audio coding. Watermarking thus enables the content provider to embed different watermarks in the same audio content. It allows, for example, to embed a user-specific watermark in the specific copy of the recording in order to identify this specific user copy of the recording later. This is not possible with fingerprinting. Watermarking also allows to

An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics, First Edition. Alexander Lerch.

Table 9.1 Main properties of fingerprinting and watermarking in comparison

Property	*Fingerprinting*	*Watermarking*
Allows Legacy Content Indexing	+	–
Allows Embedded (Meta) Data	–	+
Leaves Signal Unchanged	+	–
Identification of	Recording	User or Interaction

embed meta data directly — the user has direct access to this additional data (e.g., song title or artist name) while with fingerprinting he would have to rely on a database connection or local tags. The major disadvantage of watermarking is that the audio signal has to be modified. This can on the one hand possibly degrade the audio quality (with similar quality degradation as caused by perceptual encoders) and on the other hand cannot cover legacy audio recordings which have been either already distributed or through other (distribution) channels. Table 9.1 summarizes the main properties of fingerprinting and watermarking.

A fingerprinting system consists of two basic building blocks, the fingerprint extraction of the seed tracks and a database of previously extracted fingerprintings coupled with unique identifiers or additional meta data about the piece of music. Figure 9.1 visualizes these blocks; the upper part of the graph shows the process of adding new entries to the database (done by the service provider) and the lower part shows the query for a recording by a client.

The requirements on a general fingerprinting system have been summarized by Cano et al. [368] as:

- *Accuracy & reliability*: high number of correct identifications (TPs) compared to the number of missed identifications (FNs) and wrong identifications (FPs).
- *Robustness & security*: high accuracy even in case of a heavily distorted signal. Possible distortions include lossy compression, added noise, equalization, interference, and non-linearities of the transmission path. Sophisticated systems should also be robust against changes in tempo and pitch.
- *Granularity*: the shorter the length of an excerpt required for its identification the better (modern systems require a length of a few seconds).
- *Versatility*: independence of detection from the file format and the file origin as well as an application-independent implementation.
- *Scalability*: good performance on very large databases and a large number of simultaneous identification queries.
- *Complexity*: low computational cost of both extracting a fingerprint and finding this fingerprint in the database.

9.1 Fingerprint Extraction

Since the fingerprint should be robust against bandwidth restrictions and audio format, the two most common pre-processing steps are down-mixing to a single mono channel and

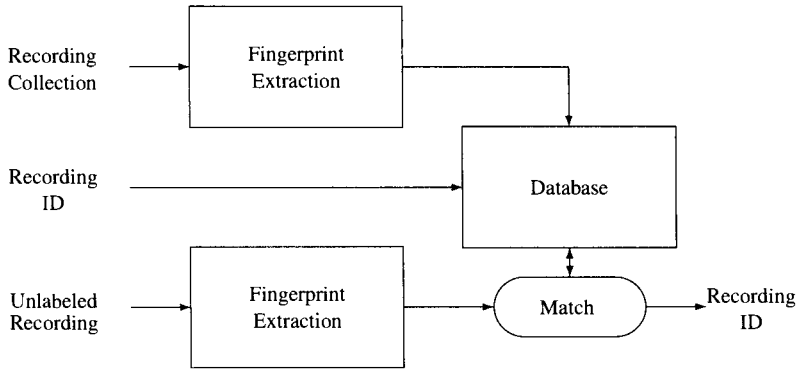

Figure 9.1 General framework for audio fingerprinting. The upper part visualizes the training phase and the lower part the query phase

down-sampling to a lower sample rate (usually 5–20 kHz). Applying a high-pass filter discards frequency components below the lowest transmittable frequency of phones in an optional pre-processing step.

The features of interest for the task of fingerprinting are robust to distortions, efficient to compute, and allow the unambiguous identification of the recording. While musical properties might help in the identification process, they are no necessity — low-level features generally should suffice.

A system that can be seen as an early predecessor of today's fingerprinting systems targeted the detection of advertisements in broadcast streams [369]. Lourens used the advertisement's energy envelope and selected a "unique" section serving as fingerprint. In most contemporary systems, the features are extracted in the frequency domain. These features include MFCCs [370, 371], a spectral flatness measure and a spectral crest factor per frequency band [372], a spectral centroid per subband [373], band energies [374] or (the sign of) energy band differences [375], carefully selected spectral peaks [376, 377], statistical moments of subbands [378], and modulation frequency features [379].

Frequently, the extraction process yields multiple features per block (usually each with a word length of 32 bits). In order to receive a compact information and to decrease the memory footprint many of these features (or the feature derivatives) are quantized into a binary or ternary representation.

The resulting fingerprint then contains a unique series of quantized feature values or feature vectors.

9.2 Fingerprint Matching

The extracted fingerprint, representing the unknown recording, has to be compared against all previously stored fingerprints in the database. The similarity (or distance) measure has to be fast for large databases. Common metrics include a correlation measure [380, 381], the Euclidean distance [373, 374, 382], and the Manhattan distance (which in the case of binary input equals the *Hamming distance*) [383, 384], but there exist many possible alternatives (see, e.g., [385]).

Even with a fast-to-compute similarity measure, it is not possible to compare every query fingerprint against all stored fingerprints in a large database due to workload and response time constraints. It is, for example, possible to pre-compute distances between the stored fingerprints in order to find different entry points for the query [374]. It is also possible to use different similarity measures, an efficient one to discard many database entries in a first run and a second more accurate similarity measure to be computed on the selected small subset [386].

There are many other ways to improve database performance; one example would be to pre-sort the database entries with their "popularity" in order to reduce search time for songs with frequent queries.

9.3 Fingerprinting System: Example

To allow a better understanding of the process of audio fingerprinting, a widespread and frequently referenced system will be explained in detail in the following. It is the Philips fingerprinting system as published by Haitsma et al. [384].

After the signal is down-mixed to one channel and down-sampled to a sample rate of 5 kHz, it is subjected to a (von-Hann-windowed) STFT. The block length is 0.37 s and the hop size is 11.6 ms. The large block overlap ratio increases the system's robustness against time-shift operations.

The magnitude spectrum is divided into 33 non-overlapping bands in the range 300–2000 Hz. The bandwidth is logarithmically increasing with frequency to take into account the non-linear frequency resolution of the human ear (see Sect. 5.1). The energy E per band with band index k is then used to derive a binary result by using both the time and frequency derivative:

$$v_{\mathrm{FP}}(k,n) = \begin{cases} 1 & \text{if } \big(\Delta E(k,n) - \Delta E(k,n-1)\big) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{9.1}$$

with

$$\Delta E(k,n) \quad = \quad E(k,n) - E(k+1,n). \tag{9.2}$$

This results in a 32-bit word per STFT; Haitsma et al. refer to this word as *subfingerprint*. One complete fingerprint consists of 256 subsequent subfingerprints and has thus a length of 3 s. Figure 9.2 shows an overview of the subfingerprint extraction.

The distance measure for the database search is the Manhattan distance; since the fingerprints are binary the Manhattan distance equals the *Hamming distance*. The length of 3 s appears to be sufficient for the identification of a song from the database. The database has to contain the series of all subfingerprints of each complete recording. Thus, if the database contains one million songs of approximately 5 min length, it holds more than 25 billion subfingerprints. Even in the case of a highly compressed subfingerprint format and the use of the computationally efficient Hamming distance, this amount of subfingerprints rules out the brute force approach of searching the whole database for each query.

Haitsma et al. suggested two methods to improve computational efficiency of the database search, a simple and a more refined method. First, a lookup table is added to the database. This table contains all possible 32-bit subfingerprints which leads to a maximum number of 2^{32} table entries. Each table entry points to a list of occurrences in the database. The lookup table can also be replaced by a hash table for efficiency.

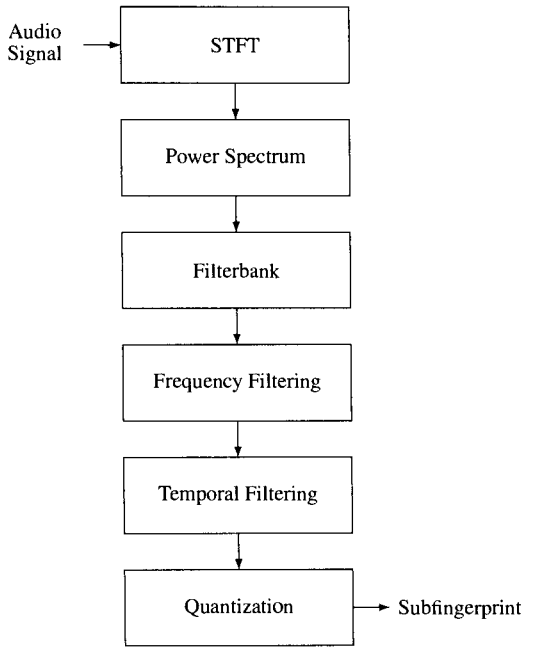

Figure 9.2 Flowchart of the extraction process of subfingerprints in the Philips system

The simple approach is based on the assumption that at least one of the 256 extracted subfingerprints has an exact match at the correct position in the database. Therefore, only the database entries listed under one of the 256 subfingerprints of the current query have to be evaluated as possible matches.

While this approach reduces the workload dramatically, the assumption that there is at least one subfingerprint without a bit error is only valid for audio with minor degradations. For highly distorted signals a larger number of bit errors can be expected. It is logical to assume bit errors in the subfingerprints. This has the disadvantage of drastically increasing the workload: if *one* bit error is expected per subfingerprint, the number of database queries and this the computational workload increases by a factor of 33. In order to reduce this additional workload while still taking into account possible bit errors, the concept of the *reliability* of a bit error is introduced in the enhanced system proposal. Since the bits of a subfingerprint are computed by energy differences, the likelihood of a bit being flipped (a bit error) is high for small energy differences and low for large differences. Thus, the bits can be ranked by their reliability and only the unreliable bits have to be flipped for the database search.

CHAPTER 10

MUSIC PERFORMANCE ANALYSIS

Music is a performing art. While the differentiation between the score (or the underlying musical ideas) and its performance is hard in the case of popular music, this is not the case with classical western music. Here, it requires a performer or a group of performers who "self-consciously enacts music for an audience" [387]. The performers render the composer's work, a score containing musical ideas and performance instructions, into a physical realization.

10.1 Musical Communication

The communication between composer and listener can be visualized as a chain of musical communication derived from Kendall and Carterette as shown in Fig. 10.1 [388]. No direct communication takes place between composer and listener. Instead, the composer translates his musical ideas into a score which is analyzed by the performer to devise a performance concept or plan and finally to render the acoustic realization — the actual music performance — which is subsequently perceived by the listener. Each of the communication stages allows or even enforces interpretation, modification, addition, and dismissal of information.

10.1.1 Score

A musical *score* standing in the tradition of western music history always contains information on pitch and (relative) duration of each note as well as instructions on musical

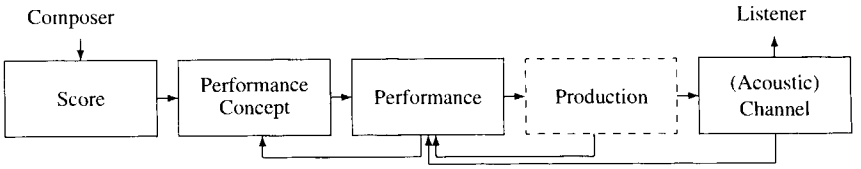

Figure 10.1 Chain of musical communication

dynamics (compare Sects. 4.2, 5.2, and 6.2). Additional instructions, for instance, on character, quality, or specific ways to perform may also be found in the score. Some of the contained information is available only implicitly (for example, information on the musical structure) or might be ambiguous or hidden, complicating its description and quantification as pointed out by many musicologists and music psychologists [319, 389–391].

All this information is subject to the performers' interpretation — they detect and evaluate implicit information, try to understand and explain performance instructions, identify ways to convey their understanding of musical ideas to the listener, and transform the discrete score representation of pitch, duration, and dynamics to continuous scales.

It can be observed that modern scores tend to be more explicit in terms of performance instructions than historic scores, indicating the composers' intention to eliminate the unspecified or ambiguous information in the score [389]. This may be due to the increasing awareness of the fact that scores often take into account performance rules that may seem "natural" at the time of composition but may change over decades and centuries, possibly leading to "unintended" performances.

Although the literature on musical performance sometimes conveys the impression that imprecision and restriction of the score representation is undesirable, there can be no doubt that there is no true, absolute, or optimal interpretation. Music is a living art and constant re-interpretation is the artistic breath giving music life.

10.1.2 Music Performance

The *music performance* is the acoustic realization of the score. Several authors defined the expressive parts of a performance as the deviations from a reference performance. Seashore saw a "neutral," mechanical score rendition as reference [45]. Other authors defined such a reference performance as a performance which is *perceived* as mechanic (which may not be necessarily a mechanical performance [191]) or as a performance with "perfectly normative rubato (and the equivalent on all other relevant expressive parameters)" [392] which is a performance matching all standard or default expectations of the (average) listener.

Every performance requires a concept or plan created by either a rigorous or a rather intuitive and unsystematic analysis of the score (for instance, in the case of sight-reading). The performance plan is a mental representation of the music, an abstract list of actions that may be realized in an indefinite number of ways and is specified only relative to the context [192, 393]. The performance plan is so closely related to the performance itself that it does not always make sense to treat them separately, and the following paragraphs will not always differentiate between the plan and the performance itself.

A music performance is highly individual in both its production and its perception. Still, a list of parameters that the performance may depend on can be compiled. The number of influencing parameters on the performance (and the performance plan) itself is probably infinite; nevertheless, the following list attempts to describe the main influences which

may explicitly or implicitly influence a musical performance. This list has been inspired by numerous texts on music performance [191, 387, 389, 390, 393–399]:

- *General interpretative rules*: These are rules, conventions, or norms which every performance follows because it would be perceived as uncommon or even unnatural otherwise.
- *Performance plan and expressive strategy*: A concept of interpretation as a list of actions that may be influenced by
  - *the interpretation of musical structure* or shape, e.g., the question of how to successfully convey melody, phrases, etc. to the listener,
  - *the addition of unexpectedness* by deviation from expected conventions or rules,
  - *the stylistic and cultural context and rules* possibly changing over time, varying between countries or following "performance fashions" [396], depending on both the historic context (the time the piece of music was composed or premiered) as well as on the context at the time of the performance. This includes instruments or instrument characteristics, tuning frequencies and temperaments, and specific performance styles with respect to articulation, ornamentation, vibrato styles, tempo, and *rubato*,
  - *the musical mood and emotional expression* the performer plans to convey to the listener, and
  - *the performance context* such as the expected audience, the style and performance plan of other performances and pieces in the concert program.
- *Performers' personal, social, and cultural background*: A broad category including, e.g., previous performing and general experiences, teachers and mentors, attitude, manners and mannerisms, etc.
- *Physical influences*: The auditory and motorical or — more generally — physical and cognitive abilities of the performer may lead to forced or unintended deviations from the performance plan. This covers general human limitations such as the motoric precision in timing as well as attributes of the musical instrument that impose limitations on, e.g., fingering and breathing.
- *Rehearsal*: The rehearsal phase allows direct feedback on the performance plan and may also train some specific motorical abilities of the performer.
- *Immediate influences*: Influences which may change the performance at the time of performance and may lead to a deviation from the performance concept such as
  - *runtime feedback control*, the feedback that the performer directly receives consisting of auditory, visual, tactile, and other cues [400]; examples include the instrument's sound and reaction, the performance of co-performers, the acoustics of the environment, and the reaction of the audience,
  - *external influences* not directly related to the performance such as humidity, temperature, distractions, and
  - *"internal" influences* such as the emotional and physical state of the performers (stress, stage fright, fatigue, illness, etc.).

10.1.3 Production

Recorded performances can differ significantly from the live performance, even in the case of so-called live recordings [397, 401]. The reason is that persons other than the performers themselves, for instance, the producer, sound engineer, and editor will influence the final result during the production stage. Furthermore, mechanical and technological restrictions enforce differences between an original and reproduced performance but also open up new possibilities of improving a recorded performance during the post-production process. To give an example, it is established recording practice to not only record several complete performances and finally choose the "best," but instead to record several so-called *takes* of passages of the musical piece. The recording process can also involve repeated listening to the recorded takes and discussions on the performance with influence on the following performances. Afterward, it is decided which parts of these takes will finally be used on the published CD, and these will be edited in a way that the edit points are inaudible. Having analyzed seven productions of Beethoven's 9th Symphony, Weinzierl and Franke found between 50 and 250 cuts between different takes in each production; the average number of edits increased with the technical evolution [402]. Modern software allows the editing of audio signals at nearly any score position.

Microphones and their positioning as well as signal processing applied by the sound and mastering engineers may impact the loudness, the timbre, the reverberation, and other parameters of the recording. These "interventions" can also vary over time to artificially increase or decrease acoustical or performance-based effects (e.g., increase the loudness of a specific instrument for its solo part). Maempel et al. give an overview on processing options and typical objectives in the post-production context [403].

The musician's and the producer team's influences are not distinguishable on the final product, for example, the CD [404]. It is common practice to refer to the resulting recording as music performance; this seems to be a valid approach as the artist usually states his final agreement with the recording.

10.1.4 Recipient

The listener, as the receiving end point of the chain of musical communication, subjectively interprets the music. He listens to a performance and conceives musical ideas and other information conveyed by the performance. As Lundin points out, the kinds of possible affective reactions of listeners are practically limitless [405] and allow a multiple of different research angles.

10.2 Music Performance Analysis

Music Performance Analysis (MPA) aims at studying the performance of a musical score rather than the musical score itself. It deals with the observation, extraction, description, interpretation, and modeling of music performance parameters as well as the analysis of attributes and characteristics of the generation and perception of music performance.

Different areas of research contribute to the field of MPA, including musicology, (music) psychology and engineering. A valuable introduction to the research field is given by Clarke [406]. Articles providing extensive overviews have been compiled, for instance, by Gabrielsson [192], Palmer [390] and Goebl et al. [407].

Three basic directions can be identified in the field of systematic performance analysis, namely the study of

- the *performance* itself, i.e., the identification of common and individual characteristics in the performance data, general performance rules, or differences between individual performances,
- the generation or *production of a performance*, i.e., the understanding of the underlying principles of performance plans, the relation of the performers' intention to objective performance parameters, and the performers' motoric and memory skills, and
- the *reception of a performance*, i.e., the investigation of how performances or the variation of specific parameters are perceived by a listener and how he is affected.

MPA could on the one hand lead to more explicit formulations of the different (objective) performance characteristics in the practice of music teaching or enable the development of teaching assisting systems giving the student direct and objective feedback on the performance parameters. On the other hand, it could assist the implementation of performance models which generate computer renditions of human-like music performances. MPA also allows us to gain valuable insights for the research fields music psychology, music aesthetics, and music history.

As Clarke points out, "musical analysis is not an exact science and cannot be relied upon to provide an unequivocal basis for distinguishing between errors and intentions" [406], emphasizing the challenge of meaningful interpretation of extracted performance data. A related difficulty that MPA has to deal with is to distinguish between inherent performance attributes and individual performance attributes. In the context of musical accents, Parncutt [191] distinguishes between *immanent accents* which are assumed to be apparent from the score (structural, harmonic, melodic, metrical, dynamic, instrumental) and *performed accents* "added" to the score by the performer. This approach may be applied to nearly all extracted parameters, and in the general case it might not be possible to distinguish score-inherent and performer-induced characteristics.

The interpretation of the meaning of parameters derived from performance data is a difficult task. In the end, final conclusions can only be drawn by taking into account subjective judgments. The methodology and questionnaire or rating scale for such subjective tests and how they relate to performances, however, has only begun to evolve to systematic approaches during the last decade [408]. The problem of extracting relevant characteristics is apparent in the design of systems intended to automatically generate music performances from a score. Clarke notes (in the context of parameters possibly influencing performances): "Whatever the attitude and strategy of different performers to this wealth of influence, it is clear that a theory of performance which is presented as a set of rules relating structure to expression is too abstract and cerebral, and that the reality is far more practical, tangible and indeed messy" [396, p. 66].

10.2.1 Analysis Data

10.2.1.1 Data Acquisition

The acquisition of empirical data is one of the crucial points in systematic MPA. Among the various methods that have been proposed and used to acquire data, two general approaches can be identified: monitoring performances (or performance parameters) by mechanical or

technical devices, or extracting the parameters from an audio recording of the performance. Both concepts have inherent advantages and disadvantages.

The monitoring approach usually provides accurate and detailed results since the measurement devices can track the performance parameters more or less directly, but the analysis is exclusively restricted to specific performances which were produced under special conditions and with the specific performers that were available.

The direct extraction of performance parameters from the audio signal — as opposed to from the instrument with sensors — is difficult and most definitely results in less accurate data. This is true for both the manual annotation of audio (such as marking onset times) and the fully automated extraction of data. Additionally, some parameters of interest may be even impossible to extract from the audio such as information on piano pedaling or note-off times. Other parameters of interest such as the performers' movements are obviously not extractable from the audio at all.

The advantage of extracting parameters directly from the audio signal is the possibility to analyze an enormous and continuously growing heritage of recordings, including outstanding and legendary performances recorded throughout the last century and until now. Hence, audio-based approaches allow to widen the empirical basis considerably with respect to the amount of available sources and their significance.

To extract the tempo curve from an audio recording, the usual approach is to either tap along with the performance [409, 410] or to manually annotate the onset times in a wave editor/display or a similar application [184, 411–419]. Both approaches have also been automated or partly automated by the use of automatic beat tracking systems (see Sect. 6.5) — followed by manual correction of beat times — [306, 420–423] or more recently by audio-to-score alignment algorithms using MIDI data as additional input [238, 269, 273] (compare Sect. 7.3.2). The main difference between tap-along and beat-tracking approaches as compared to manual onset time annotation and alignment systems is that in the former case the resulting tempo curve resolution is on the beat level, meaning that between-beat timing variations cannot be analyzed, while the latter usually takes into account each single onset time, whether this note lies on a beat or not.

Piano or Keyboard Performance

The introduction of mechanical pianos at the end of the 19th century made the acquisition of objective performance data possible through piano rolls. For example, Hartmann presented an early analysis of tempo and timing of two piano performances based on their piano rolls [424]. There are also later approaches to the analysis of performance data from piano rolls [425].

Other historic approaches used proprietary sensors that were built to extract performance data. The most prominent example is the *Iowa Piano Camera* that was used by Seashore and his team at the University of Iowa in the 1930s [45]. For each piano key, this "camera" recorded onset and note-off times and hammer velocity by optical means. Another example of a proprietary system is Shaffer's Bechstein grand piano using photo cells to detect hammer movements [187].

The introduction of the MIDI specification in the 1980s [3] resulted in an increasing number of electronic instruments and MIDI sequencers as well as compatible computer hardware and software solutions and opened up new possibilities to measure, store, and analyze pianists' performance data. Partly, music performance research has been done with the help of electronic instruments such as synthesizer keyboards and electronic pianos [426–428], but the majority concentrated on using acoustic instruments with built-in sensors which automatically output MIDI (or similar) data such as the Yamaha Disklavier

product series or Bösendorfer grand pianos with the so-called *SE-System* [180, 332, 334, 429–440].

Other Instruments or Instrumentations

Most non-piano instruments represented in the literature on music performance are monophonic, meaning that two or more notes can never occur simultaneously. In this case, common approaches to fundamental frequency detection are robust enough to extract the variation of pitch over time (compare Sect. 5.3). Proprietary as well as commercially available systems have been applied to the task of pitch extraction for MPA [45, 115, 324, 441–447]. Seashore invented the "Tonoscope" for the pitch analysis of monophonic signals [4]. It consists of a rotating drum covered with a paper containing small dots, each representing a certain frequency. The input signal is — by the means of a light-emitting gas tube — projected on the rotating paper. If the input frequency matches one of the frequencies a dot represents, this line of dots will stand still for the observer and gives a clear indication of the frequency. The "Melograph" appears to be basically of a similar design [444]. Other studies work with spectrogram visualizations, use commercially available software solutions for the detection of monophonic pitches, or implemented their own software algorithms for the pitch detection.

The majority of these systems are not able to extract onset times with sufficient accuracy, so tempo and timing information is either not analyzed or is extracted by manual annotation. However, to name two counter-examples, Kendall compared timing and dynamics of monophonic melodies performed on piano, clarinet, oboe, violin, and trumpet [388], and Ramirez et al. used automatically extracted timing data for the identification of performers of violin recordings [447].

The tempo and timing data for other, non-monophonic signals has usually been extracted by tapping along [410] or by manually setting onset time labels [186], [448], [449]. Clynes did not analyze the tempo on a beat or onset level but measured the overall duration of single movements [450].

Lerch analyzed a set of string quartet performances of a movement of a late Beethoven quartet with respect to tempo, timing, and timbre by utilizing an automated system accompanied by manual correction [274].

10.2.1.2 Instrumentation

The majority of musical performance research focuses on the piano as the instrument of main interest. One of the obvious reasons is that the piano is a very common instrument with a large (solo) repertoire, but there are more reasons that make the piano an appealing choice. The tones produced by a piano have a percussive character that makes this instrument far more suitable for accurate timing analysis than, for instance, string instruments. Its mechanics make it possible to measure data with sensors less intrusive than on other instruments that offer a more direct interaction between performer and sound production. Furthermore, the pianist is in some ways more restricted than other instrumentalists; he is limited to fixed (and equally tempered) pitch frequencies which rules out expressive intonation and other performance specifics such as vibrato. He also has little influence on the timbre of a played note, and after hitting a key, he is not able to control any of the typical note parameters such as pitch, loudness, or timbre except its duration. From a technical point of view, these restrictions seem to make the piano a rather unattractive instrument with limited degrees of freedom, but even with these limitations, piano performances are an integral part of western cultural life, meaning that the mentioned restrictions do not really impede the communication of musical expression be-

tween pianist and audience. The reduction of possible parameter dimensions is, however, beneficial in performance research because it keeps the measurement data set smaller. Last but not least, the (commercial) availability of electronic and acoustic instruments using MIDI as a universal communication protocol simplified the performance data acquisition significantly since custom-built solutions were no longer necessary. While the recording of MIDI data from other non-keyboard instruments is at least partly possible, the fact that MIDI is a keyboard-focused protocol results in limited usefulness in many cases.

Despite the good reasons for the usage of piano as the main instrument for performance analysis, it has not yet been conclusively shown that the insights gained from piano performance analysis can be applied to performances with other instruments and ensembles (although the few studies done on other instruments indicate that this might at least partly be the case).

Other solo instruments include the singing voice [45, 115, 441, 445], string instruments such as violin, viola, and violoncello [45, 324, 388, 443, 444, 446, 447], wind instruments such as flute, clarinet, oboe, and trumpet [388, 442, 444], organ [448, 449], and percussion instruments [451].

There exist also some publications on chamber music performance [186, 274, 410, 450].

10.2.1.3 Variety and Significance of Input Data

With respect to the question if and how reliably conclusions can be drawn from the extracted data, it is important to verify how and from whose performance this data has been generated.

For example, it could be argued that performance data gathered under "laboratory conditions" is insignificant per se due to the unnatural recording environment; however, these special conditions are also given for many (studio) recording sessions which resulted in recordings that are in fact perceived as convincing performances by the listeners.

Still, when the data is acquired under such laboratory conditions, it implies that the number and possibly the skill of the available performers might be limited. For example, research had partly been done on student performances [180, 332, 413, 429–431, 433, 434, 439, 443]. This fact by itself is not too remarkable, but it nevertheless emphasizes the question if and how research methods and conclusions take into account the possible discrepancies between the performances of student pianists (or just *available* pianists) and the performances of professional (and *famous*) pianists. Under the assumption that fame is related to higher professional skills of the performer this could be a noteworthy criterion.

Due to the difficulties of acquiring large sets of performance data described above, the number of performers per study is usually small. The majority of research in the presented paper database has been done with a number of 5 or less performers per publication [186, 187, 194, 324, 388, 410, 411, 420, 423, 424, 427, 428, 433, 435–438, 440, 441, 445, 449, 451–454] or 6–10 performers [180, 421, 422, 429–432, 439, 455]. Some studies evaluate a larger number of 15–25 performers [184, 274, 412, 413, 434, 444] and an analysis of an outstanding number of 108 performers (115 performances) has been presented by Repp in the late 1990s [414–416].

This raises the question if and how insights gained from a small group of performers can be extrapolated to allow general assumptions on performances.

10.2.1.4 Extracted Parameters

The basic categories of information extractable from audio signals have been introduced earlier as temporal, tonal, intensity-related, and timbral (compare Sect. 1.1).

The variation of tempo and timing is one of the most thoroughly researched aspects in MPA. The analysis of the articulation is in most cases restricted to keyboard performances captured in MIDI format. Articulation is then simply interpreted as a measure of performed note overlap or note duration with respect to the note duration as given by the score.

In order to analyze the musical dynamics in a performance, the level or loudness is extracted using sound level or psycho-acoustically motivated loudness measurements as presented in Chap. 4. Strictly speaking, such measurements do not correspond directly to musical dynamics as these would depend on the musical context, on the instrument or instrumentation, and on the timbre. Nevertheless, intensity and loudness measurements seem to provide a reasonable approximation to dynamics [84, 456].

Pitch-related performance parameters such as vibrato and intonation can be analyzed by extracting the fundamental frequency variation from the audio signal. Due to technological restrictions of current analysis systems for polyphonic music, this usually has been limited to monophonic input signals.

The analysis of timbre deviations in performances is probably one of the least-researched parameters in MPA. This may be on the one hand due to the multi-dimensional nature of timbre (see Sect. 3.3), on the other hand because it is assumed to be of least importance and partly of high correlation with dynamics. One study on the timbre variation of string quartet performances led to inconclusive results [274].

10.2.2 Research Results

10.2.2.1 Performance

Many studies focus on a rather descriptive approach to performance analysis by just analyzing extracted data such as the tempo curve [45, 184, 187, 411, 412, 414, 424, 426, 457] or the loudness/energy curve [45, 415, 429] to identify attributes of the extracted parameters between different performances and performers.

The relation of musical structure (melodic, metric, rhythmic, harmonic, etc.) or the musical gestalt to tempo and loudness deviations has been intensely researched [187, 188, 387, 390, 410, 415, 424, 429, 440, 453, 458, 459]. Most authors agree on the close relationship between musical structure such as musical phrases or accents and performance deviations mainly in tempo and timing. In particular, larger tempo changes seem to be most common at phrase boundaries. There is a general tendency to apply *ritardandi* or note lengthening at the end of a phrase and moments of musical tension [184, 274, 412, 414, 426].

There are no conclusive results on the coupling of timing with dynamic patters [415, 429].

Desain et al. and Repp report on the influence of overall tempo on expressive timing strategies [427, 460]. They find that the concept of relational invariance cannot be simply applied to expressive timing at different tempi, a result similar to Windsor's, who analyzed tempo-dependent grace note timing [435]. The overall tempo might also influence overall loudness, an effect possibly linked to the increasing amplitude of pianists' vertical finger movements toward higher tempi [461].

Goebl investigated the relationship of the composer's tempo indications (*andante*, *allegro*, etc.) with the "real" tempo and was not able to separate different tempo classes sufficiently well with the tempo extracted from the performance [194]. The number of note events per minute, however, seemed to be easier to map to the tempo indications.

Studies on the timing of pedaling in piano performance indicate some relationship between pedal timing and overall tempo [428, 431].

In the context of keyboard instruments the articulation, or the amount of key (non-) overlap, has been studied [424, 426, 432, 433, 448, 449, 462]. In summary, key overlap times for *legato* articulation seem to decrease with increasing IOIs.

Studies of the accuracy of timing synchronization of two and more performers showed that performers are highly capable of synchronizing onset times even when modulating the tempo [186, 187]. Other publications deal with the timing synchronicity between both hands or between the melody and the accompaniment in piano music [187, 424]. In many cases of piano performance, a lead of the melody before accompanying voices can be observed [426], but whether this represents a performance concept or a consequence of the higher velocity of the melody tones is subject of discussion [180, 434].

The evaluation of the consistency of repeated performances of the same performers has shown their ability to reproduce a rendition quite exactly in terms of timing [45, 187], dynamics [429], and pedal timing [428]. This seems to be the case for performances spaced by several years as well [410, 415]).

Performance data from student and professional performances has been compared in [426] and [413]. While individual differences tended to be more pronounced among the professionals, both groups seemed to share the same general performance concepts.

Repp investigated the (statistical) relationships between the extracted performance data and sociocultural variables such as the artists' gender, nationality, year of birth, and recording date but, although the correlation was sometimes significant, pointed out that these results should be regarded with caution and that individual differences are likely to outweigh any sociocultural correlation [414, 415]. In a similar study with a smaller data set, Lerch found no significant relationships [274].

Walker showed that instrumental timbre may influence several performance parameters such as timing, articulation, and dynamics [442].

The analysis of vocal performances focuses frequently on the evaluation of vibrato rates and depth and the change or stability of pitch over time [45, 115, 443, 445] or other intonation characteristics of the performance [441, 444]. Fletcher analyzed the vibrato (and other acoustical features) of flute players [116].

Statistical and machine learning approaches have been tested to use the extracted tempo and loudness information for the purpose of classification, structuring the data, or extracting general rules from the data. Dovey tried to extract general as well as individual rules from two of Rachmaninov's piano roll recordings by using a logic programming approach [425]. Supervised learners can be used to assign representations of the extracted performance data to the corresponding artists with promising results [421, 422, 436, 454]. Other machine learning methods have been used to identify general performance rules [417–419, 437, 438] and to determine individual differences between artists [420].

10.2.2.2 Performer

While the publications listed above deal mainly with the analysis of the performance itself, the second area of MPA tries to determine the capabilities and goals of performers.

Repp analyzed the type of errors (i.e., pitch deviations from score) pianists make during a performance and checked if and how severe they were perceived by listeners, coming to the conclusion that the errors concentrated in less important parts of the score and thus were hard to recognize [430].

The relationship between the performers' intentions and the parameters extracted from performances has been studied in various ways. Palmer found good correspondence between notated intentions with respect to melody and phrasing and the extracted timing parameters [426]. Also, systematic relationships between the intended emotionality of the

performance and the performance data (that is, representations of loudness and timing) can be detected [323–325, 454].

Other studies investigate the importance of the feedback of the music instrument to the performer [393]; there have been studies reporting on the effect of deprivation of auditory feedback [439, 455], investigated the performers' reaction to delayed or changed auditory feedback [463–465], or evaluated the role of tactile feedback in a piano performance [466].

Publications on the nature of memorization and learning of a musical piece (or its performance) tried to identify differences between novice and expert performers [467], to learn more on the nature of performance memory itself [468–470], and to find out more on the relation between a real and a virtual, imagined performance [439].

10.2.2.3 Recipient

It is the listener of a music performance who ultimately consumes, interprets, and probably judges it. Overall judgment ratings of performance data have been investigated in various studies. In an early publication, Repp reported some significant relations of ratings to measured timing patterns [412], while in a later study he had to conclude that "the aesthetic impression of the original recordings rested primarily on aspects other than those measured (such as texture, tone, or aspects of timing and dynamics (...))" [416]. Timmers did a similarity rating experiment and concluded that performances are judged in other ways than generally used to represent performance data [306]. In a different study, she let listeners rate the goodness of fit of two parts of different performance pairs [471]. Kendall investigated the communication of three levels of expressiveness: without expression, with appropriate expression, and with exaggerated expression [388]. Listeners were in many cases able to identify these three levels. Thompson et al. investigated the variation of listener ratings for a performance over time and found that the listening time to reach a decision was typically in the short range of 15–20 s [472]. Weinzierl and Maempel investigated how much of the listener's impression of the performance can be explained by common acoustical features [408].

Juslin detected relationships between moods and both tempo and loudness cues [323], and Kantor reported on associations of such cues and emotional reactivity [331]. Similar conclusions have been drawn from studying the time-variant emotional valence or the arousal and its relationship with performance data [332, 333]. Timmers found strong correlation between the dynamics and listener's judgments of emotionality and very good communication of emotional activity between performer and listener [334, 473]. In another study, she examined the influence of recording age and reproduction quality, observing that judgments of age and quality changed strongly with the recording date, in contrast to the perceived emotion which were mostly independent of the recording date; the communication of emotional valence tended to be more restrained for old recordings [474]. Husain varied the tempo and the mode (major, minor) of a performance and found indications that tempo modifications had an effect on arousal and mode modifications on mood [335]. Krumhansl evaluated the influences on timing and loudness variations on judgments of musical tension and found a close relationship of musical structure with both the listeners' musical tension rating and the performance data [458].

has been studied by Dixon, who found listeners to prefer smoothed beat sequences over the performed ones [423].

Lapidaki investigated the dependency of the initial tempo of a performance on the preferred tempo of a musical piece [475]; he found a general dependency although he also identified a group of listeners that were able to come to very consistent tempo preferences. Repp found systematic deviations between the tapping of listeners and metronomical time of music events, a result that seems to correspond well with the performers' inability to render a performance mechanically [476]. Aarden reported dependencies between tempo and "melodic expectancy" [477].

APPENDIX A

CONVOLUTION PROPERTIES

Convolution is one of the most regularly applied operation in audio signal processing. It applies to all linear and quasi-linear systems such as filters and rooms. In this chapter the most fundamental properties of this operation will be derived.

A.1 Identity

The result of a convolution with a delta function is the signal itself:

$$x(i) = \delta(i) * x(i). \tag{A.1}$$

The result for each individual sample can be computed by the sum of the sample itself (weighted by 1) and all other samples weighted by 0, i.e., the sample value itself [compare Eq. (B.29)].

A.2 Commutativity

Changing the order of operands does change the result of the convolution operation. That means that the distinction between impulse response and signal is of no mathematical consequence in the context of convolution:

$$h(i) * x(i) = x(i) * h(i). \tag{A.2}$$

An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics, First Edition. Alexander Lerch.

This can be shown by substituting $j' = i - j$:

$$\begin{aligned} x(i) * h(i) &= \sum_{j=-\infty}^{\infty} h(j) \cdot x(i-j) \\ &= \sum_{j'=-\infty}^{\infty} h(i-j') \cdot x(j') \\ &= \sum_{j'=-\infty}^{\infty} x(j') \cdot h(i-j'). \end{aligned} \tag{A.3}$$

A.3 Associativity

The associative property of convolution means that changing the order of subsequent convolution operations does not change the overall result. When applying two or more filters to a signal, the output will be identical for every order of filters.[1] This means that

$$\big(g(i) * h(i)\big) * x(i) = g(i) * \big(h(i) * x(i)\big). \tag{A.4}$$

This can be derived by changing the order of sums and shifting the operands as shown below:

$$\begin{aligned} \big(g(i) * h(i)\big) * x(i) &= \sum_{j=-\infty}^{\infty} \big(g(j) * h(j)\big) \cdot x(i-j) \\ &= \sum_{j=-\infty}^{\infty} \left(\sum_{l=-\infty}^{\infty} g(l) \cdot h(j-l) \right) \cdot x(i-j) \\ &= \sum_{j=-\infty}^{\infty} \left(\sum_{l=-\infty}^{\infty} g(l) \cdot h(j-l) \cdot x(i-j) \right) \\ &= \sum_{l=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} g(l) \cdot h(j-l) \cdot x(i-j) \\ &= \sum_{l=-\infty}^{\infty} g(l) \cdot \sum_{j=-\infty}^{\infty} h(j-l) \cdot x(i-j) \\ &= \sum_{l=-\infty}^{\infty} g(l) \cdot \sum_{j=-\infty}^{\infty} h(j) \cdot x(i-l-j) \\ &= \sum_{l=-\infty}^{\infty} g(l) \cdot \big(h(i-l) * x(i-l)\big) \\ &= g(i) * \big(h(i) * x(i)\big). \end{aligned} \tag{A.5}$$

[1] Strictly speaking this is only true for unlimited word length. The lower the word length the more the output signal differs from the expected signal.

A.4 Distributivity

The order of different linear operations is irrelevant due to the distributive property of the convolution, for example:

$$g(i) * \big(h(i) + x(i)\big) = g(i) * h(i) + g(i) * x(i). \tag{A.6}$$

This means that two signals, one computed by applying a filter to two different signals and summing them together afterwards, the other computed by applying the filter to the sum of the signals, are identical:

$$\begin{aligned} g(i) * \big(h(i) + x(i)\big) &= \sum_{j=-\infty}^{\infty} g(j) \cdot \big(h(i-j) + x(i-j)\big) \\ &= \sum_{j=-\infty}^{\infty} g(j) \cdot h(i-j) + g(j) \cdot x(i-j) \\ &= \sum_{j=-\infty}^{\infty} g(j) \cdot h(i-j) + \sum_{j=-\infty}^{\infty} g(j) \cdot x(i-j) \\ &= g(i) * h(i) + g(i) * x(i). \end{aligned} \tag{A.7}$$

A.5 Circularity

The convolution with a periodic signal will result in a periodic output signal. The periodic signal $x(i)$ is the sum of the shifted (fundamental) periods with length N:

$$x(i) = \sum_{n=-\infty}^{\infty} x_N(i + nN). \tag{A.8}$$

With $x_N(i) = 0$ for $i < 0 \vee i \geq N$. We can show that

$$\begin{aligned} x(i) * h(i) &= \sum_{j=-\infty}^{\infty} h(i-j) \sum_{n=-\infty}^{\infty} x_N(j + nN) \\ &= \sum_{n=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} h(i-j) \cdot x_N(j + nN) \\ &= \sum_{n=-\infty}^{\infty} x_N(i + nN) * h(i). \end{aligned} \tag{A.9}$$

The multiplication of two spectra computed with the DFT will result in a circular convolution; the result will be the convolution of the two periodically continued sample blocks.

APPENDIX B

FOURIER TRANSFORM

The Fourier Transform (FT) is widely used in audio signal analysis and synthesis. Understanding its properties is crucial for the design of audio processing systems.

Deriving the FTs fundamental properties is easier for continuous signals; we will thus focus on the continuous domain first and will then discuss the FT of windowed signals, the FT of sampled signals, and finally the Discrete Fourier Transform (DFT).

Periodic signals can be represented as a Fourier series as introduced in Eq. (2.3). The fundamental frequency ω_0 determines the "frequency resolution" of the series. For the analysis of non-periodic signals we let the period length grow $T_0 \to \infty$ (or equivalently $\omega_0 \to 0$). This has the effect that the previously discrete frequency resolution becomes continuous with $k\omega_0 \to \omega$. Due to the resulting infinite resolution of the frequency axis, the coefficients will decrease $a_k \to 0$. The formula for the Fourier series given in Eq. (2.3) thus changes into the FT:

$$X(\mathrm{j}\omega) = \mathfrak{F}\{x(t)\} = \int_{-\infty}^{\infty} x(t)e^{-\mathrm{j}\omega t}\,dt \tag{B.1}$$

and $X(\mathrm{j}\omega)$ is the so-called *spectrum* of the signal $x(t)$.

The real and imaginary parts represent the cosine and sine functions, respectively. A common form of visualizing the results is to represent the spectrum as magnitude $|X(\mathrm{j}\omega)|$ and phase $\Phi_X(\mathrm{j}\omega)$ instead of real and imaginary parts. Frequently only the *magnitude spectrum* is being used for the visualization of the spectrum while the phase spectrum

is ignored. Another common representation is the *power spectrum* which is the squared magnitude spectrum.

B.1 Properties of the Fourier Transformation

B.1.1 Inverse Fourier Transform

The FT is invertible. $X(\mathrm{j}\omega)$ can be converted back from the frequency domain into the time domain signal $x(t)$ by applying the *Inverse Fourier Transform (IFT)*:

$$x(t) = \mathfrak{F}^{-1}\{X(\mathrm{j}\omega)\} = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\mathrm{j}\omega)e^{\mathrm{j}\omega t}\,d\omega. \tag{B.2}$$

That means that time and frequency representation are equivalent, i.e., no information is gained or lost by applying the FT to a signal; it just changes the representation of the signal.

It becomes also obvious from comparing Eqs. (B.1) and (B.2) that forward and inverse transform are very similar operations (see also Sect. B.1.6).

B.1.2 Superposition

If the signal $y(t)$ is the weighted addition of the signals $x_1(t)$ and $x_2(t)$

$$y(t) = c_1 \cdot x_1(t) + c_2 \cdot x_2(t), \tag{B.3}$$

then the same relationship is true for their frequency transformation:

$$\begin{aligned} Y(\mathrm{j}\omega) &= \int_{-\infty}^{\infty} \big(c_1 \cdot x_1(t) + c_2 \cdot x_2(t)\big) \cdot e^{-\mathrm{j}\omega t}\,dt \\ &= c_1 \cdot \int_{-\infty}^{\infty} x_1(t)e^{-\mathrm{j}\omega t}\,dt + c_2 \cdot \int_{-\infty}^{\infty} x_2(t)e^{-\mathrm{j}\omega t}\,dt \\ &= c_1 \cdot X_1(\mathrm{j}\omega) + c_2 \cdot X_2(\mathrm{j}\omega). \end{aligned} \tag{B.4}$$

B.1.3 Convolution and Multiplication

The convolution of signal $x(t)$ with the impulse response $h(t)$

$$\begin{aligned} y(t) &= h(t) * x(t) \\ &= \int_{-\infty}^{\infty} h(\tau) \cdot x(t-\tau)\,d\tau \end{aligned} \tag{B.5}$$

corresponds to a multiplication in the spectral domain

$$Y(\mathrm{j}\omega) = H(\mathrm{j}\omega) \cdot X(\mathrm{j}\omega). \tag{B.6}$$

The derivation involves clever grouping and expansion:

$$\begin{aligned}
Y(\mathrm{j}\omega) &= \int_{-\infty}^{\infty} y(t)e^{-\mathrm{j}\omega t}\,dt \\
&= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} h(\tau)\cdot x(t-\tau)\,d\tau \right) e^{-\mathrm{j}\omega t}\,dt \\
&= \int_{-\infty}^{\infty} h(\tau) \int_{-\infty}^{\infty} x(t-\tau)e^{-\mathrm{j}\omega t}\,dt\,d\tau \\
&= \int_{-\infty}^{\infty} h(\tau)e^{-\mathrm{j}\omega\tau} \underbrace{\int_{-\infty}^{\infty} x(t-\tau)e^{-\mathrm{j}\omega(t-\tau)}\,d(t-\tau)}_{X(\mathrm{j}\omega)}\,d\tau \\
&= \int_{-\infty}^{\infty} h(\tau)e^{-\mathrm{j}\omega\tau}\,d\tau \cdot X(\mathrm{j}\omega) \\
&= H(\mathrm{j}\omega)\cdot X(\mathrm{j}\omega). \qquad \text{(B.7)}
\end{aligned}$$

This property allows the efficient computation of the convolution of a signal with an FIR filter with a long impulse response in the frequency domain [478].

The same relationship exists for convolution in the frequency domain. The convolution operation

$$Y(\mathrm{j}\omega) = H(\mathrm{j}\omega) * X(\mathrm{j}\omega) \qquad \text{(B.8)}$$

could be replaced by a multiplication in the time domain

$$y(t) = h(t)\cdot x(t). \qquad \text{(B.9)}$$

B.1.4 Parseval's Theorem

The energy of the signal can be calculated in both the time and the spectral domain:

$$\int_{-\infty}^{\infty} x^2(t)\,dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} |X(\mathrm{j}\omega)|^2\,d\omega. \qquad \text{(B.10)}$$

This can be shown by using the equivalence between multiplication in the frequency domain and convolution in the time domain. Writing

$$\int_{-\infty}^{\infty} h(\tau)\cdot x(t-\tau)\,d\tau = \frac{1}{2\pi}\int_{-\infty}^{\infty} H(\mathrm{j}\omega)\cdot X(\mathrm{j}\omega)e^{\mathrm{j}\omega t}d\omega \qquad \text{(B.11)}$$

and replacing $H(\mathrm{j}\omega)$ with the conjugate-complex spectrum $X^*(\mathrm{j}\omega)$ and $h(\tau)$ with $x(-\tau)$,[1] respectively, the result at $t = 0$ is

$$\begin{aligned} \int_{-\infty}^{\infty} x(-\tau) \cdot x(-\tau)\, d\tau &= \frac{1}{2\pi} \int_{-\infty}^{\infty} X^*(\mathrm{j}\omega) \cdot X(\mathrm{j}\omega)\, d\omega, \\ \int_{-\infty}^{\infty} x^2(t)\, dt &= \frac{1}{2\pi} \int_{-\infty}^{\infty} |X(\mathrm{j}\omega)|^2\, d\omega. \end{aligned} \tag{B.12}$$

B.1.5 Time and Frequency Shift

The transform of a signal shifted by a constant in time $y(t) = x(t - t_0)$ is

$$Y(\mathrm{j}\omega) = X(\mathrm{j}\omega) e^{-\mathrm{j}\omega t_0}. \tag{B.13}$$

This means that the magnitude spectrum will be identical but the phase spectrum will have a linear offset $\Phi_Y(\omega) = \Phi_X(\omega) - \omega t_0$:

$$\begin{aligned} \int_{-\infty}^{\infty} x(t - t_0) e^{-\mathrm{j}\omega t}\, dt &= \int_{-\infty}^{\infty} x(\tau) e^{-\mathrm{j}\omega(\tau + t_0)}\, d\tau \\ &= e^{-\mathrm{j}\omega t_0} \int_{-\infty}^{\infty} x(\tau) e^{-\mathrm{j}\omega\tau}\, d\tau \\ &= e^{-\mathrm{j}\omega t_0} \cdot X(\mathrm{j}\omega). \end{aligned} \tag{B.14}$$

Equivalently, the shifted spectrum[2] $Y(\mathrm{j}\omega) = X(\mathrm{j}(\omega - \omega_0))$ corresponds to the time domain signal $y(t) = x(t) \cdot e^{\mathrm{j}\omega_0 t}$ which is the original signal modulated by a sinusoidal signal:

$$\begin{aligned} \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\mathrm{j}\omega - \omega_0) e^{\mathrm{j}\omega t}\, d\omega &= \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\mathrm{j}\phi) e^{\mathrm{j}(\phi + \omega_0)t}\, d\phi \\ &= e^{\mathrm{j}\omega_0 t} \cdot x(t). \end{aligned} \tag{B.15}$$

B.1.6 Symmetry

If the time domain signal $x(t)$ is real-valued, then its frequency transform will be symmetric with $X(\mathrm{j}\omega) = X^*(-\mathrm{j}\omega)$. The magnitude is symmetric around the ordinate:

$$|X(\mathrm{j}\omega)| = |X(-\mathrm{j}\omega)| \tag{B.16}$$

while the phase is symmetric around the origin:

$$\Phi_X(\omega) = -\Phi_X(-\omega). \tag{B.17}$$

Vice versa, if the frequency transform is real-valued, then the time domain signal will be symmetric with $x(t) = x(-t)$ and if $X(\mathrm{j}\omega)$ is imaginary it means that $x(t) = -x(-t)$.

[1] Only real-valued time domain functions $x(t)$ are considered here.

[2] In real-valued time signals, this shift has to be applied symmetrically to the negative frequencies.

This can be shown by representing the time signal $x(t)$ as a sum of an even component x_e and an odd component $x_o(t)$:

$$x(t) = \underbrace{\frac{1}{2}\big(x(t) + x(-t)\big)}_{x_e(t)} + \underbrace{\frac{1}{2}\big(x(t) - x(-t)\big)}_{x_o(t)}. \tag{B.18}$$

The FT of the even and odd signal components is then

$$X_e(j\omega) = \int_{-\infty}^{\infty} x_e(t)\cos(\omega t)\,dt - j\underbrace{\int_{-\infty}^{\infty} x_e(t)\sin(\omega t)\,dt}_{=0}, \tag{B.19}$$

$$X_o(j\omega) = \underbrace{\int_{-\infty}^{\infty} x_o(t)\cos(\omega t)\,dt}_{=0} - j\int_{-\infty}^{\infty} x_o(t)\sin(\omega t)\,dt. \tag{B.20}$$

The transform of the even signal is thus purely real $X_e(j\omega) = \Re\mathfrak{e}[X(j\omega)]$, and the transform of the odd part is purely imaginary $X_o(j\omega) = \Im\mathrm{m}[X(j\omega)]$. Furthermore, due to the property $\cos(\omega t) = \cos(-\omega t)$, it becomes clear that the real part is again an even function symmetric around $\omega = 0$. The imaginary part is odd due to $\sin(\omega t) = -\sin(-\omega t)$. It follows that the magnitude spectrum is an even function and the phase spectrum is an odd function.

We have seen that the FT is very similar to the IFT; thus, if $X(j\omega)$ is the FT of the signal $x(t)$, then it would also be true that $2\pi \cdot x(-j\omega)$ is the transform of $X(t)$. This can be shown by substituting t with $-\omega$ in the IFT:

$$\begin{aligned} x(t) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\omega)e^{j\omega t}\,d\omega, \\ x(-t) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\omega)e^{-j\omega t}\,d\omega, \\ x(-j\omega) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} X(t)e^{-j\omega t}\,dt. \end{aligned} \tag{B.21}$$

B.1.7 Time and Frequency Scaling

The FT of a signal modified in the time domain by scaling the time axis $y(t) = x(c \cdot t)$ will be scaled inversely:

$$Y(j\omega) = \frac{1}{|c|}X\left(j\frac{\omega}{c}\right). \tag{B.22}$$

The derivation (for positive c) is

$$\begin{aligned} Y(\mathrm{j}\omega) &= \int_{-\infty}^{\infty} x(c \cdot t) e^{-\mathrm{j}\omega t}\, dt \\ &= \int_{-\infty}^{\infty} x(\tau) e^{-\mathrm{j}\omega \frac{\tau}{c}}\, d\frac{\tau}{c} \\ &= \frac{1}{c} \int_{-\infty}^{\infty} x(\tau) e^{-\mathrm{j}\frac{\omega}{c}\tau}\, d\tau \\ &= \frac{1}{c} X\left(\mathrm{j}\frac{\omega}{c}\right). \end{aligned} \tag{B.23}$$

For negative c, the result is $Y(\mathrm{j}\omega) = -\frac{1}{c} X(\mathrm{j}\frac{\omega}{c})$. The spectrum of a stretched signal ($c > 1$) will thus be compressed and vice versa.

From the above equation it directly follows for $c = -1$ that

$$\mathfrak{F}\{x(-t)\} = X(-\mathrm{j}\omega) \tag{B.24}$$

and for a real-valued signal $x(t)$

$$\mathfrak{F}\{x(-t)\} = X^*(\mathrm{j}\omega). \tag{B.25}$$

B.1.8 Derivatives

The transform of the nth derivative of the signal has the following property (without derivation):

$$\mathfrak{F}\left\{\frac{d^n x(t)}{dt^n}\right\} = (\mathrm{j}\omega)^n X(\mathrm{j}\omega). \tag{B.26}$$

B.2 Spectrum of Example Time Domain Signals

B.2.1 Delta Function

The *delta function* $\delta(t)$, sometimes also named *dirac impulse* or *delta impulse*, equals zero for all points in time except $t = 0$. It represents an ideal impulse and is defined by

$$\int_{-\infty}^{\infty} \delta(t)\, dt = 1, \tag{B.27}$$

$$\delta(t) = 0 \text{ for all } t \neq 0. \tag{B.28}$$

This also means that the integration of the multiplication of signal $x(t)$ with this delta function results in

$$\int_{-\infty}^{\infty} x(t) \cdot \delta(t)\, dt = x(0). \tag{B.29}$$

Thus, the result of the FT is

$$\Delta(\mathrm{j}\omega) = \int_{-\infty}^{\infty} \delta(t) e^{-\mathrm{j}\omega t}\, dt = e^{\mathrm{j}\omega \cdot 0} = 1. \tag{B.30}$$

The spectrum is therefore a real-valued constant; it follows that the delta function incorporates all frequencies with the same strength.

B.2.2 Constant

The symmetry of FT and IFT shown in Eq. (B.21), in combination with Eq. (B.30), tells us also that the spectrum of a constant valued time domain signal $x(t) = 1/2\pi$ will be $X(\mathrm{j}\omega) = \delta(\omega)$.

B.2.3 Cosine

A sinusoidal time domain signal can be interpreted as a modulated constant value. Applying the frequency shift property from Eq. (B.15) thus shows that the spectrum of a cosine is the spectrum of a constant value shifted by the cosine's frequency ω_0, the delta function $\delta(\omega - \omega_0)$.

B.2.4 Rectangular Window

The rectangular window is defined by

$$w_{\mathrm{R}}(t) = \begin{cases} 1, & -\frac{1}{2} \le t \le \frac{1}{2} \\ 0, & \text{otherwise} \end{cases}. \tag{B.31}$$

The spectrum of this window function is

$$\begin{aligned} W_{\mathrm{R}}(\mathrm{j}\omega) &= \int_{-\infty}^{\infty} w_{\mathrm{R}}(t) e^{-\mathrm{j}\omega t}\, dt \\ &= \int_{-1/2}^{1/2} e^{-\mathrm{j}\omega t}\, dt \\ &= \frac{1}{-\mathrm{j}\omega} \underbrace{\left(e^{-\mathrm{j}\omega/2} - e^{\mathrm{j}\omega/2}\right)}_{=-2j\sin(\omega/2)} \\ &= \frac{\sin(\omega/2)}{\omega/2} = \mathrm{sinc}\left(\frac{\omega}{2}\right). \end{aligned} \tag{B.32}$$

B.2.5 Delta Pulse

The *delta pulse* is a series of individual delta impulses, i.e., a superposition of delta functions. It is defined by

$$\delta_{\mathrm{T}}(t) = \sum_{i=-\infty}^{\infty} \delta(t - iT_0). \tag{B.33}$$

Each delta impulse has a distance T_0 from its neighbor. Using FT of a delta function given by Eq. (B.27) and the superposition property from Eq. (B.3) in combination with the time shift property from Eq. B.13, the FT of $\delta_{\mathrm{T}}(t)$ is

$$\Delta_{\mathrm{T}}(\mathrm{j}\omega) = \sum_{i=-\infty}^{\infty} e^{-\mathrm{j}\omega i T_0}. \tag{B.34}$$

With help from the geometric series it can be shown [7] that

$$\begin{aligned} \Delta_{\mathrm{T}}(\mathrm{j}\omega) &= \frac{2\pi}{T} \sum_{i=-\infty}^{\infty} \delta\left(\omega - \frac{2\pi i}{T}\mathrm{j}\omega T\right) \\ &= \omega_T \delta_{\omega_{\mathrm{T}}}(\omega). \end{aligned} \tag{B.35}$$

B.3 Transformation of Sampled Time Signals

A sampled time signal can be represented by the multiplication of a continuous time signal $x(t)$ multiplied by a delta pulse $\delta_{\mathrm{T}}(t)$. Equation (B.8) states that a multiplication of two time signals corresponds to the convolution of their frequency transforms. This means that

$$\begin{aligned} \mathfrak{F}\{x(i)\} &= \mathfrak{F}\{x(t) \cdot \delta_{\mathrm{T}}(t)\} \\ &= \mathfrak{F}\{x(t)\} * \mathfrak{F}\{\delta_{\mathrm{T}}(t)\} \\ &= X(\mathrm{j}\omega) * \Delta_{\mathrm{T}}(\mathrm{j}\omega). \end{aligned} \tag{B.36}$$

Note that although the time domain signal is discrete, the resulting spectrum is still continuous. As can be seen from Eq. (B.36), the spectrum is repeated periodically with ω_{T}, the sample rate. This allows a very intuitive explanation of the sampling theorem stated in Eq. (2.9) since the periodically repeated spectra would overlap if signal $x(t)$ contains higher frequencies than $\omega_{\mathrm{T}}/2$ (see Fig. B.1). In that case, reconstruction of the original signal $x(t)$ is impossible, while otherwise perfect reconstruction is possible by applying an ideal low-pass filter with a cut-off frequency of $\omega_{\mathrm{T}}/2$ to the sampled signal $x(i)$. The effect of overlapping spectra is called aliasing and is visualized in Fig. B.1.

B.4 Short Time Fourier Transform of Continuous Signals

Up to this point, we have dealt mostly with signals unlimited in time. In the real world, signals will usually have a defined start and stop time. We might also be interested in transforming only segments of such signals. This can be seen as multiplying an infinite time signal with a window function that equals zero outside the time boundaries of interest. In signal analysis, typical segment lengths range — dependent on the task at hand — between 10 and 300 ms. Smith points out three reasons for choosing segments of this length [8]:

- "Perhaps most fundamentally, the ear similarly Fourier analyzes only a short segment of audio signals at a time (on the order of 10–20 ms worth). Therefore, to match our spectrum analysis to human hearing, we desire to limit the time window of the analysis."

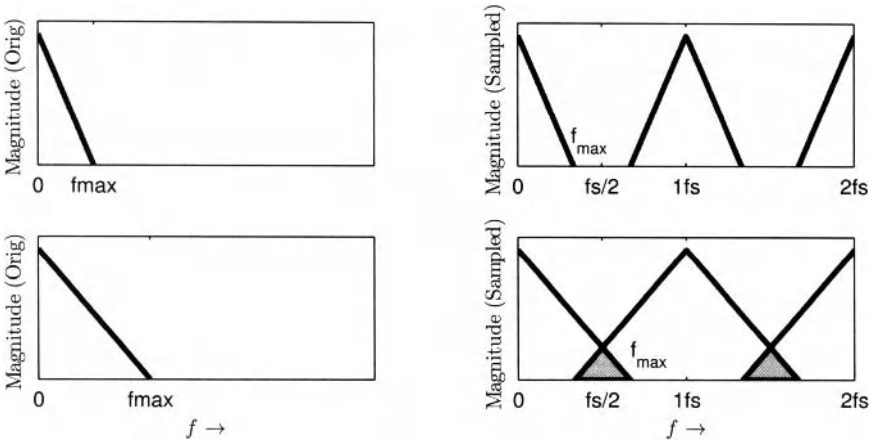

Figure B.1 Schematic visualization of the spectrum of a continuous time domain signal (left) and the sampled signal in accordance (top) and violation (bottom) of the sampling theorem

- "Audio signals typically have spectra which change over time. It is therefore usually most meaningful to restrict analysis to a time window over which the spectrum stays rather constant."
- "It can be extremely time consuming to compute the Fourier transform of an audio signal of typical length, and it will rarely fit in computer memory all at once."

B.4.1 Window Functions

Since every multiplication in the time domain corresponds to a convolution of the corresponding spectra, the spectrum of the signal is convolved with the spectrum of the window. The spectrum of the window thus has influence on the resulting spectrum. The most simple window in the time domain is a rectangular window introduced in Sect. B.2.4. The typical spectral shape of a window consists of a main lobe and many side lobes with more or less decreasing amplitude.

When the signal $x(t)$ of interest is a sinusoid, then the resulting FT of the windowed signal will therefore be a superposition of two window functions with their main lobes located at the signal's frequency $\omega_0, -\omega_0, \ldots$, so the delta functions are effectively "smeared" by windowing artifacts. This undesired side effect is referred to as spectral leakage. It is usually characterized by

- the width of the main lobe,
- the height of the first (closest) side lobe peak, and
- the rolloff or attenuation of the subsequent side lobe peaks.

In order to optimize these properties toward individual use cases, different window functions have been suggested in the past. Figure B.2 shows the presented window functions in time domain (left) and frequency domain (right).

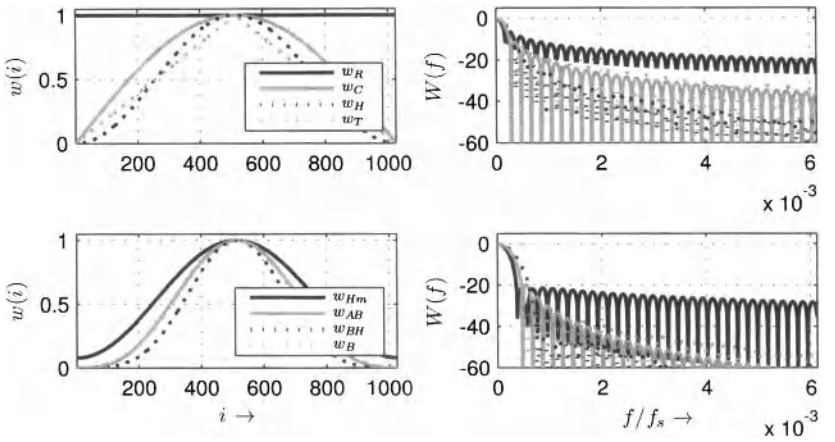

Figure B.2 Windows in time domain (left) and frequency domain (right)

B.4.1.1 Rectangular Window

The FT of the *rectangular window* has already been derived in Sect. B.2.4 to be a sinc function:

$$W_{\mathrm{R}}(\mathrm{j}\omega) = \mathrm{sinc}\left(\frac{\omega}{2}\right) \tag{B.37}$$

B.4.1.2 Bartlett Window

The *Bartlett window* has a triangular shape. It is defined by

$$\begin{aligned} w_{\mathrm{T}}(t) &= \begin{cases} t+1, & -1/2 \le t \le 0 \\ 1-t, & 0 \le t \le 1/2 \\ 0, & \text{otherwise} \end{cases} \\ &= w_{\mathrm{R}}(2t) * w_{\mathrm{R}}(2t). \end{aligned} \tag{B.38}$$

Using Eq. (B.7) it can be deduced that

$$\begin{aligned} W_{\mathrm{T}}(\mathrm{j}\omega) &= \mathfrak{F}\{w_{\mathrm{R}}(2t)\} \cdot \mathfrak{F}\{w_{\mathrm{R}}(2t)\} \\ &= \frac{1}{2} \cdot \mathrm{sinc}^2\left(\frac{\omega}{4}\right) \end{aligned} \tag{B.39}$$

B.4.1.3 Generalized Superposed Cosines

It is possible to generalize many window functions with

$$w_{\mathrm{sup}}(t) = w_{\mathrm{R}}(t) \sum_{j=0}^{\mathcal{O}-1} b_j \cos\left(\frac{\pi}{2} jt\right). \tag{B.40}$$

Different values for $\mathcal{O}$ result in different window families:

- $\mathcal{O} = 1$: *rectangular window* $w_{\mathrm{R}}(t)$

- $\mathcal{O} = 2$: Hamming family of windows:
  - *cosine window* $w_{\mathrm{C}}(t)$:
    $b_0 = 0,\ b_1 = 1$
  - *von-Hann window* $w_{\mathrm{H}}(t)$:
    $b_0 = {}^1/_2,\ b_1 = {}^1/_2$
  - *Hamming window* $w_{\mathrm{Hm}}(t)$:
    $b_0 = {}^{25}/_{46},\ b_1 = {}^{42}/_{46}$
- $\mathcal{O} = 3$: Blackman-Harris family of windows:
  - classic *Blackman window* $w_{\mathrm{B}}(t)$:
    $b_0 = {}^{7938}/_{18608},\ b_1 = {}^{9240}/_{18608},\ b_2 = {}^{1430}/_{18608}$
  - *Blackman-Harris window* $w_{\mathrm{BH}}(t)$:
    $b_0 = 0.4243801,\ b_1 = 0.4973406,\ b_2 = 0.0782793$

B.4.1.4 Generalized Power of Cosine

A different generalization of window functions is

$$w_{\mathrm{pow}}(t) = w_{\mathrm{R}}(t)\cos^{\beta}\left(\frac{\pi}{2}t\right). \tag{B.41}$$

Again, different windows can be derived for different β:

- *rectangular window* $w_{\mathrm{R}}(t)$:
  $\beta = 0$
- *cosine window* $w_{\mathrm{C}}(t)$:
  $\beta = 1$
- *von-Hann window* $w_{\mathrm{H}}(t)$:
  $\beta = 2$
- *alternative Blackman window* $w_{\mathrm{AB}}(t)$:
  $\beta = 0$

B.5 Discrete Fourier Transform

In computer applications, a discrete representation of the signal's spectrum is required; it can only be defined at discrete frequency bins. The frequency bins are evenly distributed over the interesting range of frequencies with the distance

$$\Delta\Omega = \frac{2\pi}{\mathcal{K}T_{\mathrm{S}}} = \frac{2\pi f_{\mathrm{S}}}{\mathcal{K}}. \tag{B.42}$$

The DFT of the nth block of the signal $x(i)$ will be referred to as STFT and is defined by

$$X(\mathrm{j}\Delta\Omega) = \sum_{i=0}^{\mathcal{K}-1} x(i)\exp\left(-\mathrm{j}ki\frac{2\pi}{\mathcal{K}}\right) \tag{B.43}$$

with $k = 0, 1, \ldots, \mathcal{K} - 1$.

Thus, the DFT of a block of samples of length $\mathcal{K}$ also consists of exactly $\mathcal{K}$ complex values; however, since the signal $x(i)$ is real, the result will be symmetric with

$$X(\mathcal{K} - k) = X^*(k) \tag{B.44}$$

and only $\mathcal{K}/2$ complex results need to be computed.

The spectrum $X(k, n)$ can be interpreted as the (continuous) FT of the block n of signal $x(i)$ sampled at equidistant bins at the positions $k \cdot \Delta\Omega$. It has to be periodic

$$X(k) = X(k + \mathcal{K}) \tag{B.45}$$

because the time domain signal is discrete.

The spectrum can only be discrete if the time domain signal is periodic (compare the Fourier series). Therefore, the DFT can be interpreted as the FT applied to the current block of samples periodically continued.

The *Inverse Discrete Fourier Transform (IDFT)* allows reconstruction of the time samples that had been transformed:

$$x(i) = \sum_{k=0}^{\mathcal{K}-1} X(k) e^{\mathrm{j}ki\Delta\Omega}. \tag{B.46}$$

The properties of the DFT correspond to the properties introduced for the continuous FT, but a few details have to be kept in mind: the multiplication of two DFTs corresponds to a circular convolution (similar to the CiCF) in the time domain. The same is true for time and frequency shift operations.

B.5.1 Window Functions

The discrete window functions are sampled (a potentially shifted) versions of the continuous window functions given above. The rectangular window is

$$w_{\mathrm{R}}(i) = \begin{cases} 1, & -\frac{\mathcal{K}-1}{2} \leq i \leq \frac{\mathcal{K}-1}{2} \\ 0, & \text{otherwise} \end{cases}, \tag{B.47}$$

and the superposed cosine window is

$$w_{\mathrm{sup}}(i) = w_{\mathrm{R}}(i) \sum_{j=0}^{\mathcal{O}-1} a_j \cos\left(\frac{j \cdot \pi}{\mathcal{K}} i\right). \tag{B.48}$$

The DFT of a rectangular window is

$$W_{\mathrm{R}}(k, n) = \exp\left(-\mathrm{j}\frac{\mathcal{K}-1}{2}\frac{2\pi k}{\mathcal{K}}\right) \cdot \frac{\sin\left(\frac{\mathcal{K}\frac{2\pi k}{\mathcal{K}}}{2}\right)}{\sin\left(\frac{\frac{2\pi k}{\mathcal{K}}}{2}\right)}. \tag{B.49}$$

Note that the phase shift term originates in moving the first window sample to sample 0. When transforming a windowed sine, the result will be shifted in the frequency domain

$$X(k, n) = \exp\left(-\mathrm{j}\frac{\mathcal{K}-1}{2}\frac{2\pi k}{\mathcal{K}} - \Omega_0\right) \cdot \frac{\sin\left(\frac{\mathcal{K}\frac{2\pi k}{\mathcal{K}}}{2} - \Omega_0\right)}{\sin\left(\frac{\frac{2\pi k}{\mathcal{K}}}{2} - \Omega_0\right)}. \tag{B.50}$$

Table B.1 Frequency domain properties of the most common windows (from [479])

Window	ΔB [Bins]	A_{SL} [dB]	S_{SL} [dB/Oct]	A_{WC} [dB]
w_R	0.89	−13	−6	3.92
w_T	1.28	−27	−12	3.07
w_C	1.20	−23	−12	3.01
w_H	1.44	−32	−18	3.18
w_{Hm}	1.30	−43	−6	3.10
w_B	1.68	−58	−18	3.47
w_{AB}	1.66	−39	−24	3.47
w_{BH}	1.66	−67	−6	3.45

If the sinusoidal frequency exactly fits the frequency of a bin with index k, then all bins will be zero for $k \neq k_0$. In this case, all the zero crossings of the window function fall at the spectral bin positions. However, if k_0 is between two frequency bins, then two artifacts appear: the main peak has lower level, the so-called *process loss*, and the frequency bins are now directly located at the main peaks of the side lobes. Two special cases are the *best case* and the *worst case* scenario with k_0 being directly on a bin or exactly between two bins. In the time domain, the best case means that one or more periods of the sinusoidal fit exactly into the window with length $\mathcal{K}$.

B.5.1.1 Discrete Window Properties

The following properties can be used to characterize the frequency domain representation of a window function:

- width of main lobe ΔB (3 dB bandwidth in bins),
- peak level of highest side lobe A_{SL} (dB),
- side lobe fall-off S_{SL} (dB/Oct),
- worst case process loss A_{WC} (dB).

A smaller main lobe width yields better frequency resolution and both a smaller side lobe peak level and higher side lobe fall-off results in less cross-talk between sinusoids of different frequencies. The smaller the worst case process loss, the higher the resulting amplitude accuracy.

Table B.1 summarizes these properties for common windows. For detailed introductions to spectral leakage see Harris [479] and Smith [8].

B.5.2 Fast Fourier Transform

An efficient way to compute the DFT is the *Fast Fourier Transform (FFT)*. The FFT is equivalent to the "normal" DFT; it just computes the result more efficiently. More specifically, the difference in the number of operations is approximately $O(\mathcal{K}^2)$ for the normal DFT compared to $O(\mathcal{K} \log \mathcal{K})$ for the FFT (compare [480, 481]). There are different algorithms to compute the FFT; most FFT implementations require an input block length which equals a power of 2.

APPENDIX C

PRINCIPAL COMPONENT ANALYSIS

Principal Component Analysis (PCA) maps the input variables — in our case usually a vector of features $\boldsymbol{v}$ — to a new coordinate system by a linear combination of the individual features:

$$\boldsymbol{u}(n) = \boldsymbol{T}^{\mathrm{T}} \cdot \boldsymbol{v}(n). \tag{C.1}$$

The resulting vector $\boldsymbol{u}(n)$ is the data in the new coordinate system for observation n and transformation matrix $\boldsymbol{T}^{\mathrm{T}}$ contains different linear combinations for the input feature vector $\boldsymbol{v}(n)$. The number of features in the vector will be referred to as $\mathcal{F}$. Formulating Eq. (C.1) not only for one observation but for a series of feature vectors $\boldsymbol{V}$ leads to

$$\boldsymbol{U} = \boldsymbol{T}^{\mathrm{T}} \cdot \boldsymbol{V}. \tag{C.2}$$

The transformation matrix is a square matrix with the dimensions $\mathcal{F} \times \mathcal{F}$. It is composed of vectors defining the linear combinations of the input features:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{c}_0 & \boldsymbol{c}_1 & \dots & \boldsymbol{c}_{\mathcal{F}-1} \end{bmatrix}. \tag{C.3}$$

The transformation matrix has the following main properties:

- the vectors $\boldsymbol{c}_i$ are in the direction of the highest variance in the data and the variance is concentrated in as few output components as possible,
- the vectors $\boldsymbol{c}_i$ are orthogonal to each other

$$\boldsymbol{c}_i^{\mathrm{T}} \cdot \boldsymbol{c}_j = 0 \quad \forall \; i \neq j \tag{C.4}$$

An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics, First Edition. Alexander Lerch.

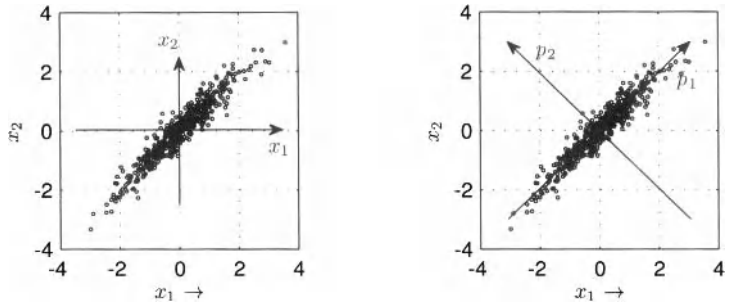

Figure C.1 Scatter plot of a two-dimensional data set with variables x_1, x_2, and the rotated coordinate system after PCA with the component axes p_1, p_2

- and the transformation is invertible:

$$\boldsymbol{v}(n) = \boldsymbol{T} \cdot \boldsymbol{u}(n). \tag{C.5}$$

Figure C.1 shows the scatter plots of two example variables x_1 and x_2 with the original axes on the left and rotated axes on the right.

C.1 Computation of the Transformation Matrix

The first step in computing the matrix $\boldsymbol{T}$ is the calculation of the feature covariance matrix:

$$\boldsymbol{R} = \frac{1}{\mathcal{F}-1} \cdot (\boldsymbol{v} - \boldsymbol{\mu}_v)\left(\boldsymbol{v}^{\mathrm{T}} - \boldsymbol{\mu}_v^{\mathrm{T}}\right) \tag{C.6}$$

with the vector $\boldsymbol{\mu}_v$ containing the arithmetic mean of each feature. The covariance matrix is square, symmetric, and has only positive entries.

The eigenvectors of the covariance matrix represent the axes of the new coordinate system and thus comprise the transformation matrix. Usually, they are ordered with decreasing eigenvalues; the vector in the first column $\boldsymbol{c}_0$ is the vector with the highest eigenvalue and the vector in the last column has the lowest eigenvalue.

C.2 Interpretation of the Transformation Matrix

The transformation matrix $\boldsymbol{T}$ contains useful information on the input data set. Each column is a different linear combination of the input features, and they are sorted according to their eigenvalues or, in other words, according to the variance this component contributes to the overall variance. Features with high influence on the first components can be assumed to be of higher importance than features with high influence on the last components. Thus, PCA can be useful for both feature subset selection and feature space transformation. As the last components have only limited impact on the result, they might be discarded. The matrix $\boldsymbol{T}$ can then be truncated from the dimensions $\mathcal{F} \times \mathcal{F}$ to $\mathcal{F} \times \mathcal{L}$ with $\mathcal{L}$ being the required number of components in the space transformation process.

APPENDIX D

SOFTWARE FOR AUDIO ANALYSIS

The process of developing software for audio analysis can generally be split into two steps, the algorithmic design including the prototyping as well as the evaluation and the implementation of "production-quality" software which is made available to customers and users. The final software is commonly implemented in the programming languages *C++* [482] and *Java*.[1] Many developers consider C++ as the language that allows the most workload-efficient implementations and Java as the language that allows a comparably rapid development cycle.

While C++ and Java are also used for algorithm prototyping, other languages such as *Python*,[2] often extended by the packages *NumPy*, *SciPy*, and *IPython*,[3] are more common. *Matlab*[4] (or the largely compatible open-source software *Octave*[5]) provides an environment with a simple script language with a large set of already included routines as well as various possibilities of data visualization. Furthermore, visual audio programming environments such as *Max*[6] or *Pure Data (PD)*[7] are used especially in the context of designing real-time audio algorithms.

[1] Java. http://www.java.com. Last retrieved on Jan. 28, 2012.
[2] Python. http://www.python.org. Last retrieved on Jan. 28, 2012.
[3] NumPy. http://numpy.scipy.org, SciPy. http://www.scipy.org, IPython. http://ipython.org. Last retrieved on Jan. 28, 2012.
[4] Matlab. http://www.mathworks.com/products/matlab. Last retrieved on Jan. 28, 2012.
[5] Octave. http://www.octave.org. Last retrieved on Jan. 28, 2012.
[6] Max. http://cycling74.com/products/max. Last retrieved on Jan. 28, 2012.
[7] Pure Data. http://puredata.info. Last retrieved on Jan. 28, 2012.

The aim of this appendix is to provide a short overview about software supporting the design and implementation of audio analysis algorithms. The solutions presented below include toolboxes, libraries, frameworks, and applications. They cover a multitude of ACA-related tasks, and each solution focuses on improving or accelerating the prototyping or development process in different areas. The main areas of interest can be identified as

- *annotation* of (audio) files and generation of ground truth data,
- *feature extraction*,
- *pattern recognition* and machine learning algorithms, and
- *visualization* of features and properties of (collections of) audio files.

The software may also differ in the level of user expertise (developer vs. non-technical user), in its real-time capabilities, and in the input data sources (e.g., audio vs. MIDI).

D.1 Software Frameworks and Applications

This section presents software frameworks offering comprehensive possibilities for file input and output, signal processing, audio analysis, and machine learning. Thus, they offer the prototyping and possibly the building of ACA systems. These frameworks do not depend on any specific environment and do not require the usage of other software.

D.1.1 Marsyas

Marsyas is a software framework in C++ designed by George Tzanetakis. It offers both real-time and offline processing of audio data. While Marsyas features also audio synthesis and (effect) processing, its emphasis is on MIR. It is written in C++ but allows users access to configuring and using Marsyas, for instance, through a Python front end.

Since its first publication [297], Marsyas has grown into a powerful set of classes which allow to construct basically any MIR-related system [483]. Numerous of the algorithms described in this book are already implemented in Marsyas.

The design of Marsyas is modular; it consists of processing blocks which can be composed into data flow networks. A block can represent different functionality; it can, for instance, be an audio or text file IO, a sound card audio IO or an algorithmic operation on audio samples or features. Marsyas also includes a set of machine learning tools for training and classification.

The latest information and the source code of Marsyas is available from the project's web page.[8]

D.1.2 CLAM

A software framework developed by a research team of the Music Technology Group of Pompeu Fabra University, Spain, is *C++ Framework for Audio and Music (CLAM)*. It is implemented in C++ and its emphasis is on spectral modeling and spectral processing.

The first version of CLAM was presented in 2002 [484, 485]. The basic design principles seem to be similar to Marsyas in the way that the researcher can build networks of

[8]Marsyas. http://marsyas.info. Last retrieved on Jan. 28, 2012.

individual processing blocks. Nowadays, CLAM comes with a visual network editor that allows the user to build the processing networks through a graphical user interface [486]; its main focus remains on audio processing and less on audio analysis. It does, for instance, not include any machine learning blocks.

The latest information and the source code of CLAM is available from the project's web page.[9]

D.1.3 jMIR

The software framework *jMIR*, implemented in Java, is developed and maintained at the Music Technology Group at McGill University. Its focus is on machine-learning-related tasks on music such as musical genre classification and mood classification.

The framework consists of a set of tools for feature extraction and machine learning algorithms as well as tools for meta data annotation. The tools can also be used individually [487, 488]. One of the most notable differences to the functionality of other frameworks is the built-in functionality to extract features directly from symbolic data such as MIDI. It comes with both an audio and a MIDI ground truth data set for musical genre classification.

The latest information and the source code of jMIR is available from the project's web page.[10]

D.1.4 CoMIRVA

CoMIRVA is a Java framework for the visualization of large collections of music data. It is developed at the Johannes Kepler University Linz.

The software offers various possibilities of analyzing and visualizing the relationship (such as the similarity) of music files in large collections. This functionality is complemented by some feature extraction routines and web data mining tools [489].

The latest information and the source code of CoMIRVA is available from the project's web page.[11]

D.1.5 Sonic Visualiser

Sonic Visualiser is a software for the visualization of various properties of one or more audio files. The project was initiated at the Centre for Digital Music at Queen Mary University of London.

Sonic Visualiser was first presented in 2006 and has been under development since then [490]. It offers numerous possibilities to visualize and annotate audio. Compared to the frameworks presented above, it does not focus as much on the algorithm developer but on (non-technical) expert users such as musicologists who need access to objective information on the audio recording. In order to be able to easily integrate new tools providing analysis data for the visualization, it offers a plugin interface called *VAMP* (see below).

The latest information and the source code of Sonic Visualiser is available from the project's web page.[12]

[9]CLAM. http://clam-project.org. Last retrieved on Jan. 28, 2012.
[10]jMIR. http://jmir.sourceforge.net. Last retrieved on Jan. 31, 2012.
[11]CoMIRVA: Collection of Music Information Retrieval and Visualization Applications. www.cp.jku.at/CoMIRVA. Last retrieved on Jan. 31, 2012.
[12]Sonic Visualiser. http://www.sonicvisualiser.org. Last retrieved on Jan. 31, 2012.

D.2 Software Libraries and Toolboxes

In contrast to the frameworks presented above, the following software solutions focus on specific aspects such as feature extraction. Many of them require either a programming environment (for instance, Matlab) or they are libraries to be linked against a custom-designed software.

D.2.1 Feature Extraction

The *MIRtoolbox* is a toolbox for Matlab and has been developed by a team at the University of Jyväskylä. It provides an extensive set of functions for the extraction of low-level features as well as for the analysis of tonal, structural, and temporal properties [491]. The latest information and the source code of the MIRtoolbox is available from the project's web page.[13]

The *libXtract* library has been presented by Bullock from the Birmingham Conservatoire [492]. It is programmed in C and deals with the extraction of low-level features. The libXtract library allows to arbitrarily cascade the feature extraction routines for the computation of subfeatures on different hierarchical levels. The latest information and the source code of libXtract is available from the project's web page.[14]

Content-based Audio and Music Extraction Library (CAMEL) is a comparably new library for feature extraction and aggregation aspiring to become a framework for various MIR-related tasks. It is implemented in C++ and was presented by a team of the University of Lethbridge [493]. The latest information and the source code of CAMEL is available from the project's web page.[15]

Yet Another Audio Feature Extractor (YAAFE) is a command line tool for the extraction of low-level features published by the Telecom ParisTech [494]. It is implemented in C++ but provides Python bindings as well. YAAFE aims at very efficient feature extraction by utilizing a *feature plan parser*, determining and removing redundant processing steps in the calculation of different features. The latest information and the source code of YAAFE is available from the project's web page.[16]

The *Timbre Toolbox* is a Matlab toolbox for the extraction of a large set of (low-level) features. It is the result of a joint effort of IRCAM and McGill University [495]. The source code of the Timbre Toolbox is available online.[17]

SCMIR is an extension for the *SuperCollider* audio programming language. It is developed by Collins [496] and allows access to ACA routines through a high-level programming language. The emphasis is on feature extraction in order to use the extracted information for "creative musical activity." SCMIR offers the extraction of standard features, the analysis of tempo and beat, as well as structural analyses. The latest information and the source code of SCMIR is available from the project's web page.[18]

[13] MIRtoolbox. https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mirtoolbox. Last retrieved on Jan. 31, 2012.
[14] libXtract. http://libxtract.sourceforge.net. Last retrieved on Jan. 31, 2012.
[15] CAMEL – A Framework for Audio Analysis. http://camel-framework.sourceforge.net. Last retrieved on Jan. 31, 2012.
[16] Yaafe – audio features extraction. http://yaafe.sourceforge.net. Last retrieved on Jan. 31, 2012.
[17] http://recherche.ircam.fr/pub/timbretoolbox. Last retrieved on Jan. 31, 2012.
[18] SCMIR. http://www.sussex.ac.uk/Users/nc81/code.html. Last retrieved on Jan. 31, 2012.

Maaate is an audio analysis toolkit written in C++. It was designed to perform feature extraction and analysis routines directly on encoded MPEG Layer 1–3 files [497]. The latest information and the source code of Maaate is available from the project's web page.[19]

A library designed for feature extraction in a real-time context is *aubio*. It is written in C++ and enables the user to extract several low-level features as well as both onsets and tempo in a causal way in order to support real-time applications. The latest information and the source code of aubio is available from the project's web page.[20]

OpenSMILE is a C++ library for feature extraction. It addresses both speech processing and music processing by combining features typically used in both domains in one feature extractor [498]. The latest information and the source code of OpenSMILE is available from the project's web page.[21]

D.2.2 Plugin Interfaces

One idea to face the problem of multiple and apparently redundant implementations of the same (low-level) features in various libraries and frameworks is to define a plugin *Application Programmer's Interface (API)*. The advantage of using plugins for feature extraction is the possibility of dynamically loading new plugins at runtime without compilation or static linking, complemented by the simple exchange of plugins (features) between researchers. A plugin API itself is thus a rather general interface definition of how to extract an unknown feature as opposed to the actual implementation of various features.

D.2.2.1 FEAPI

The *Feature Extraction Application Programmer's Interface (FEAPI)*, named here for historic reasons, was a plugin API for the extraction of low-level features. It was a joint effort with participants from the four institutions Ghent University, IRCAM, Technical University of Berlin and zplane.development [499].

The project is not actively maintained or developed anymore. One of the most likely reasons why FEAPI was not accepted by the research community was the lack of host applications. Although a Max port existed and had been used, other (graphical) user interfaces except for a simple command line interface did not exist.

The documentation and source code of FEAPI, complemented by example implementations of several spectral and loudness features, can still be found on the FEAPI project web page.[22]

D.2.2.2 VAMP

VAMP is the only reasonably widespread plugin interface for feature extraction from audio signals. It has been defined and developed by a team of the Centre for Digital Music of the Queen Mary University of London [500]. VAMP plugins can be hosted by the *Sonic Visualiser* (see above). The basic functionality is similar to the functionality of FEAPI.

Several plugins, as well as the latest information and the source code of VAMP is available from the project's web page.[23]

[19]Maaate!. http://maaate.sourceforge.net. Last retrieved on Jan. 31, 2012.
[20]aubio, a library for audio labelling. http://aubio.sourceforge.net. Last retrieved on Jan. 31, 2012.
[21]openSMILE: The Munich Versatile and Fast Open-Source Audio Feature Extractor. http://opensmile.sourceforge.net. Last retrieved on Jan. 31, 2012.
[22]FEAPI. http://feapi.sf.net. Last retrieved on Jan. 31, 2012.
[23]VAMP Plugins: The Vamp audio analysis plugin system. http://vamp-plugins.org. Last retrieved on Jan. 31, 2012.

D.2.3 Other Software

There is a multitude of other software applications and libraries that can be used in the context of ACA. Very important are the various machine learning and data mining solutions; probably the most-cited software is the *Waikato Environment for Knowledge Analysis (WEKA)* software [501].[24] WEKA is a comprehensive collection of machine learning algorithms and data pre-processing tools such as algorithms for regression, classification and clustering. *Torch* provides a Matlab-like environment for machine learning tasks [502].[25] It can be used with the programming language *Lua*.[26] Other libraries focus on specific areas of machine learning; examples are *libsvm*[27] [503] and *The Spider*, a machine learning environment for Matlab.[28] *HTK* is a toolkit for building and manipulating hidden Markov models [62].[29] Due to its focus on speech recognition HTK also offers the extraction of various features.

[24]Weka 3: Data Mining Software in Java. http://www.cs.waikato.ac.nz/ml/weka. Last retrieved on Jan. 31, 2012.
[25]Torch 5. http://torch5.sourceforge.net. Last retrieved on Jan. 31, 2012.
[26]The Programming Language Lua. http://www.lua.org. Last retrieved on Jan. 31, 2012.
[27]LIBSVM – A Library for Support Vector Machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm. Last retrieved on Jan. 31, 2012.
[28]The Spider. http://people.kyb.tuebingen.mpg.de/spider. Last retrieved on Jan. 31, 2012.
[29]HTK Speech Recognition Toolkit. http://htk.eng.cam.ac.uk. Last retrieved on Jan. 31, 2012.

REFERENCES

[1] J. S. Downie, "Music Information Retrieval," *Annual Review of Information Science and Technology*, vol. 37, B. Cronin, Ed., pp. 295–340, 2003. DOI: 10.1002/aris.1440370108.

[2] N. Orio, "Music Retrieval: A Tutorial and Review," *Foundations and Trends in Information Retrieval*, vol. 1, no. 1, J. Callan and F. Sebastiani, Eds., pp. 1–90, 2006. DOI: 10.1561/1500000002.

[3] MIDI Manufacturers Association, "Complete midi 1.0 detailed specification v96.1, 2nd ed.," MMA, Standard, 2001.

[4] C. E. Seashore, "A Voice Tonoscope," *Studies in Psychology*, vol. 3, pp. 18–28, 1902.

[5] D. P. W. Ellis, "Extracting Information from Music Audio," *Communications of the ACM*, vol. 49, no. 8, pp. 32–37, Aug. 2006, ISSN: 00010782. DOI: 10.1145/1145287.1145310.

[6] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 2nd. Englewood Cliffs, NJ: Prentice Hall, 1999, ISBN: 0-13-754920-2.

[7] J. Ohm and H. Lüke, *Signalübertragung — Grundlagen der digitalen und analogen Nachrichtenübertragungssysteme*, 10th ed. Berlin: Springer, 2007.

[8] J. Smith, "Spectral Audio Signal Processing — March 2010 Draft," Stanford University, Center for Computer Research in Music and Acoustics (CCRMA), Stanford, CA, Tech. Rep., 2010, Last retrieved on Dec. 19, 2011. [Online]. Available: http://www.dsprelated.com/dspbooks/sasp.

[9] J. W. Gibbs, "Fourier's Series," *Nature*, vol. 59, no. 1522, pp. 200–200, Dec. 1898, ISSN: 0028-0836. DOI: 10.1038/059200b0.

[10] ——, "Fourier's Series," *Nature*, vol. 59, no. 1539, pp. 606–606, Apr. 1899, ISSN: 0028-0836. DOI: 10.1038/059606a0.

[11] V. A. Kotelnikov, "On the Transmission Capacity of "Ether" and Wire in Electric Communications," *Reprint by the Institute of Radioengineering and Electronics of the Moscow Power Engineering Institute (Technical University) of the original article from 1933*, 2003. DOI: 10.1070/PU2006v049n07ABEH006048.

[12] C. E. Shannon, "Communication in the Presence of Noise," *Proceedings of the IRE*, vol. 37, pp. 10–21, 1949, Reprinted in Proceedings of the IEEE 86(2), 1998. DOI: 10.1109/JPROC.1998.659497.

[13] J. M. Whittaker, "The "Fourier" Theory of the Cardinal Function," *Proceedings of the Edinburgh Mathematical Society*, vol. 1, pp. 169–176, Jan. 1929, ISSN: 0013-0915. DOI: 10.1017/S0013091500013511.

[14] H. Nyquist, "Certain Topics in Telegraph Transmission Theory," *Transactions of the American Institute of Electrical Engineers*, vol. 47, no. 2, pp. 617–644, Apr. 1928, ISSN: 0096-3860. DOI: 10.1109/T-AIEE.1928.5055024.

[15] S. W. Smith, *Digital Signal Processing (The Scientist's and Engineer's Guide to)*, 2nd. San Diego: California Technical Publishing, Jul. 1999, ISBN: 978-096601763-2.

[16] S. Fulop and K. Fitz, "Algorithms for Computing the Time-Corrected Instantaneous Frequency (Reassigned) Spectrogram, with Applications," *Journal of the Acoustical Society of America (JASA)*, vol. 119, no. 1, pp. 360–371, 2006. DOI: 10.1121/1.2133000.

[17] M. Lagrange and S. Marchand, "Estimating the Instantaneous Frequency of Sinusoidal Components Using Phase-Based Methods," *Journal of the Audio Engineering Society (JAES)*, vol. 55, no. 5, pp. 385–399, 2007.

[18] F. Auger and P. Flandrin, "Improving the Readability of Time-Frequency and Time-Scale Representations by the Reassignment Method," *Transactions on Signal Processing*, vol. 43, no. 5, pp. 1068–1089, 1995. DOI: 10.1109/78.382394.

[19] J. C. Brown, "Calculation of a constant Q spectral transform," *Journal of the Acoustical Society of America (JASA)*, vol. 89, no. 1, pp. 425–434, 1991, ISSN: 00014966. DOI: 10.1121/1.400476.

[20] J. C. Brown and M. S. Puckette, "An Efficient Algorithm for the Calculation of a Constant Q Transform," *Journal of the Acoustical Society of America (JASA)*, vol. 92, no. 5, p. 2698, 1992, ISSN: 00014966. DOI: 10.1121/1.404385.

[21] C. Schörkhuber and A. P. Klapuri, "Constant-Q Transform Toolbox for Music Processing," in *Proceedings of the 7th Sound and Music Computing Conference (SMC)*, Barcelona, 2010.

[22] R. F. Lyon, A. G. Katsiamis, and E. M. Drakakis, "History and Future of Auditory Filter Models," in *Proceedings of the International Symposium on Circuits and Systems (ISCAS)*, Institute of Electrical and Electronics Engineers (IEEE), May 2010, pp. 3809–3812, ISBN: 978-1-4244-5308-5. DOI: 10.1109/ISCAS.2010.5537724.

[23] E. D. Boer and H. R. de Jongh, "On Cochlear Encoding: Potentialities and Limitations of the Reverse-Correlation Technique," *Journal of the Acoustical Society of America (JASA)*, vol. 63, no. 1, p. 115, Jan. 1978, ISSN: 00014966. DOI: 10.1121/1.381704.

[24] A. M. H. J. Aertsen and P. I. M. Johannesma, "Spectro-Temporal Receptive Fields of Auditory Neurons in the Grassfrog," *Biological Cybernetics*, vol. 38, no. 4, pp. 223–234, Nov. 1980, ISSN: 0340-1200. DOI: 10.1007/BF00337015.

[25] M. Slaney, "An Efficient Implementation of the Patterson-Holdsworth Auditory Filter Bank," Apple Perception Group, Tech. Rep. 35, 1993.

[26] L. Van Immerseel and S. Peeters, "Digital Implementation of Linear Gammatone Filters: Comparison of Design Methods," *Acoustics Research Letters Online*, vol. 4, no. 3, p. 59, 2003, ISSN: 15297853. DOI: 10.1121/1.1573131.

[27] M. Slaney, "Auditory Toolbox — Version 2," Interval Research Corporation, Vienna, Tech. Rep. TR-1998-010, 1998, Last retrieved on Dec. 12, 2011. [Online]. Available: http://cobweb.ecn.purdue.edu/~malcolm/interval/1998-010.

[28] T. Tolonen and M. Karjalainen, "A Computationally Efficient Multipitch Analysis Model," *Transactions on Speech and Audio Processing*, vol. 8, no. 6, pp. 708–716, 2000. DOI: 10.1109/89.876309.

[29] N. Levinson, "The Wiener RMS (Root Mean Square) Error Criterion in Filter Design and Prediction," *Journal of Mathematical Physics*, vol. 25, no. 4, pp. 261–278, 1947.

[30] J Durbin, "The Fitting of Time-Series Models," EN, *Review of the International Statistical Institute*, vol. 28, no. 3, pp. 233–244, Nov. 1960.

[31] N. S. Jayant, "Digital Coding of Speech Waveforms: PCM, DPCM, and DM Quantizers," *Proceedings of the IEEE*, vol. 62, no. 5, pp. 611–632, 1974, ISSN: 0018-9219. DOI: 10.1109/PROC.1974.9484.

[32] P. P. Vaidyanathan, *The Theory of Linear Prediction*. Morgan & Claypool, 2008, ISBN: 978-159829576-4. DOI: 10.2200/S00086ED1V01Y200712SPR03.

[33] ISO/IEC JTC1/SC29 15938-4:2002, "Information Technology — Multimedia Content Description Interface — Part 4: Audio," ISO/IEC, Standard, 2002.

[34] G. Peeters, "A Large Set of Audio Features for Sound Description (Similarity and Classification) in the CUIDADO Project," IRCAM, Project Report (CUIDADO), 2004.

[35] G. Eisenberg, *Identifikation und Klassifikation von Musikinstrumentenklängen in monophoner und polyphoner Musik*. Göttingen: Cuvillier, 2008, ISBN: 978-3-86727-825-6.

[36] J. Smith and P. Gosset, "A Flexible Sampling-Rate Conversion Method," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, San Diego: Institute of Electrical and Electronics Engineers (IEEE), Mar. 1984. DOI: 10.1109/ICASSP.1984.1172555.

[37] R. Rasch and R. Plomb, "The Perception of Musical Tones," in *The Psychology of Music*, D. Deutsch, Ed., New York: Academic Press, 1982.

[38] E. Zwicker and R. Feldtkeller, *Das Ohr als Nachrichtenempfänger*, 2nd. Stuttgart: S Hirzel, 1967.

[39] B. Moore, *An Introduction to the Psychology of Hearing*, 4th. London: Academic Press, 1997.

[40] ASA S1.1-1960, "Acoustical Terminology," American Standards Association (ASA), Standard, 1960.

[41] A. Bregman, *Auditory Scene Analysis*. MIT Press, 1994.

[42] J. Blumenbach, *The Elements of Physiology*, 4th. London: Longman, Reese, Orme, Brown, and Green, 1828.

[43] H. Helmholtz, *Die Lehre von den Tonempfindungen als physiologische Grundlage für die Theorie der Musik*, 3rd. Braunschweig: Vieweg, 1870.

[44] C. Stumpf, *Tonpsychologie II*. Hilversum/Amsterdam: Knuf and Bonset, 1890, Reprint 1965.

[45] C. E. Seashore, *Psychology of Music*. New York: McGraw-Hill, 1938, Reprint 1967, Dover Publications, New York.

[46] C. Reuter, *Der Einschwingvorgang nichtperkussiver Musikinstrumente*. Frankfurt: Peter Lang, 1995.

[47] E. Zwicker and H. Fastl, *Psychoacoustics. Facts and Models*, 2nd. Berlin: Springer, 1999.

[48] G. Bismarck, "Sharpness as an Attribute of the Timbre of Steady Sounds," *Acustica*, vol. 30, pp. 159–172, 1974.

[49] P. Iverson and C. Krumhansl, "Isolating the Dynamic Attributes of Musical Timbre," *Journal of the Acoustical Society of America (JASA)*, vol. 94, no. 5, pp. 2595–2603, 1993. DOI: 10.1121/1.407371.

[50] S. McAdams, S. Winsberg, S. Donnadieu, G. Soete, and J. Krimphoff, "Perceptual Scaling of Synthesized Musical Timbres: Common Dimensions, Specifities, and Latent Subject Classes," *Psychological Research*, vol. 58, pp. 177–192, 1995. DOI: 10.1007/BF00419633.

[51] S. Lakatos, "A Common Perceptual Space for Harmonic and Percussive Timbres," *Perception & Psychophysics*, vol. 62, no. 7, pp. 1426–1439, 2000. DOI: 10.3758/BF03212144.

[52] J. Marozeau, A. Cheveigné, S. McAdams, and S. Winsberg, "The Dependency of Timbre on Fundamental Frequency," *Journal of the Acoustical Society of America (JASA)*, vol. 114, no. 5, pp. 2946–2957, 2003. DOI: 10.1121/1.1618239.

[53] E. Schubert, J. Wolfe, and A. Tarnopolsky, "Spectral Centroid and Timbre in Complex, Multiple Instrumental Textures," in *Proceedings of the 8th International Conference on Music Perception & Cognition (ICMPC)*, Evanston, Aug. 2004.

[54] A. Caclin, S. McAdams, B. Smith, and S. Winsberg, "Acoustic Correlates of Timbre Space Dimensions: A Confirmatory Study Using Synthetic Tones," *Journal of the Acoustical Society of America (JASA)*, vol. 118, no. 1, pp. 471–482, 2005. DOI: 10.1121/1.1929229.

[55] S. B. Davis and P. Mermelstein, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," *Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357–366, Aug. 1980, ISSN: 0096-3518. DOI: 10.1109/TASSP.1980.1163420.

[56] J. Foote, "Content-Based Retrieval of Music and Audio," in *Proceedings on Multimedia Storage and Archiving Systems II, Part of SPIE's International Symposium on Voice, Video and Data Communications*, Dallas, Nov. 1997. DOI: 10.1117/12.290336.

[57] B. Logan, "Mel Frequency Cepstral Coefficients for Music Modeling," in *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, Plymouth, Oct. 2000.

[58] E. Pampalk, S. Dixon, and G. Widmer, "On the Evaluation of Perceptual Similarity Measures for Music," in *Proceedings of the 6th International Conference on Digital Audio Effects (DAFX)*, London, Sep. 2003.

[59] J. Burred and A. Lerch, "A Hierarchical Approach to Automatic Musical Genre Classification," in *Proceedings of the 6th International Conference on Digital Audio Effects (DAFX)*, London, Sep. 2003.

[60] G. Tzanetakis and P. Cook, "Musical Genre Classification of Audio Signals," *Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002. DOI: 10.1109/TSA.2002.800560.

[61] M. McKinney and J. Breebart, "Features for Audio and Music Classification," in *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, Baltimore, Oct. 2003.

[62] S. Young, G. Evermann, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *THE HTK BOOK*, 2002. DOI: 10.1.1.124.3972.

[63] S. Molau, M. Pitz, R. Schlüter, and H. Ney, "Computing Mel-frequency Cepstral Coefficients on the Power Spectrum," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing. (ICASSP)*, Institute of Electrical and Electronics Engineers (IEEE), 2001, pp. 73–76, ISBN: 0-7803-7041-4. DOI: 10.1109/ICASSP.2001.940770.

[64] N. Jayant and P. Noll, *Digital Coding of Waveforms — Principles and Applications to Speech and Video*. Upper Saddle River, NJ: Prentice Hall, 1984.

[65] ISO/IEC JTC1/SC29 11172-3:1993, "Information Technology — Coding of Moving Pictures and Associated Audio for Digital Storage Media up to about 1.5 Mbit/s — Part 3: Audio," ISO/IEC, Standard, 1993.

[66] S. Aksoy and R. M. Haralick, "Feature Normalization and Likelihood-Based Similarity Measures for Image Retrieval," *Pattern Recognition Letters*, vol. 22, no. 5, pp. 563–582, Apr. 2001, ISSN: 01678655. DOI: 10.1016/S0167-8655(00)00112-4.

[67] G. Box and D. Cox, "An Analysis of Transformations," *Journal of the Royal Statistical Society*, vol. Series B, no. 26, pp. 211–246, 1964.

[68] S. Albada and P. Robinson, "Transformation of Arbitrary Distributions to the Normal Distribution with Applications to EEG Test-Retest Reliability," *Journal of Neuroscience Methods*, vol. 161, no. 2, pp. 205–211, 2007. DOI: 10.1016/j.jneumeth.2006.11.004.

[69] S. Shapiro and M. Wilk, "An Analysis of Variance Test for Normality (Complete Samples)," *Biometrica*, vol. 52, no. 3/4, pp. 591–611, 1965.

[70] T. Anderson and D. Darling, "Asymptotic Theory of Certain "Goodness of Fit" Criteria Based on Stochastic Processes," *Annals of Mathematical Statistics*, vol. 23, no. 2, pp. 193–212, 1952. DOI: 10.1214/aoms/1177729437.

[71] H. Thode, *Testing for Normality*. New York: Marcel Dekker, Inc., 2002, ISBN: 0-8247-9613-6.

[72] J. Miles and M. Shevlin, *Applying Regression & Correlation: A Guide for Students and Researchers*. London: Sage Publications, 2001, ISBN: 0761962298.

[73] F. Mörchen, A. Ultsch, M. Thies, and I. Löhken, "Modeling Timbre Distance with Temporal Statistics from Polyphonic Music," *Transactions on Audio, Speech and Language Processing*, vol. 14, no. 1, pp. 81–90, Jan. 2006, ISSN: 1558-7916. DOI: 10.1109/TSA.2005.860352.

[74] S. Wegener, M. Haller, J. Burred, T. Sikora, S. Essid, and G. Richard, "On the Robustness of Audio Features for Musical Instrument Classification," in *Proceedings of the 16th European Signal Processing Conference (EUSIPCO)*, Lausanne, Aug. 2008.

[75] R. Fiebrink and I. Fujinaga, "Feature Selection Pitfalls and Music Classification," in *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, Victoria, 2006.

[76] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[77] E. Cantú-Paz, S. Newsam, and C. Kamath, "Feature Selection in Scientific Applications," in *Proceedings of the 10th Conference on Knowledge Discovery and Data Mining (KDD)*, Seattle, Aug. 2004. DOI: 10.1145/1014052.1016915.

[78] S. S. Stevens, *Hearing — Its Psychology and Physiology*, 1983 reprint. New York: American Institute of Physics for the Acoustical Society of America, 1938.

[79] S. S. Stevens, "The Measurement of Loudness," *Journal of the Acoustical Society of America (JASA)*, vol. 27, no. 5, pp. 815–829, 1955. DOI: 10.1121/1.1908048.

[80] H. Fletcher and W. A. Munson, "Loudness, Its Definition, Measurement and Calculation," *Journal of the Acoustical Society of America (JASA)*, vol. 5, no. 2, p. 82, 1933, ISSN: 00014966. DOI: 10.1121/1.1915637.

[81] H. Fletcher, "Auditory Patterns," *Reviews of Modern Physics*, vol. 12, no. 1, pp. 47–65, 1940. DOI: 10.1103/RevModPhys.12.47.

[82] S. S. Stevens, J. Volkmann, and E. B. Newman, "A Scale for the Measurement of the Psychological Magnitude Pitch," *Journal of the Acoustical Society of America (JASA)*, vol. 8, no. 3, pp. 185–190, 1937, ISSN: 00014966. DOI: 10.1121/1.1915893.

[83] B. Moore and B. Glasberg, "Suggested Formulae for Calculating Auditory-Filter Bandwidths and Excitation Patterns," *Journal of the Acoustical Society of America (JASA)*, vol. 74, no. 3, pp. 750–753, 1983. DOI: 10.1121/1.389861.

[84] T. Nakamura, "The Communication of Dynamics between Musicians and Listeners through Musical Performance," *Perception & Psychophysics*, vol. 41, no. 6, pp. 525–533, 1987. DOI: 10.3758/BF03210487.

[85] W. Goebl and R. Bresin, "Measurement and Reproduction Accuracy of Computer-Controlled Grand Pianos," *Journal of the Acoustical Society of America (JASA)*, vol. 114, no. 4, pp. 2273–2283, 2003. DOI: 10.1121/1.1605387.

[86] R. B. Dannenberg, "The Interpretation of MIDI Velocity," in *Proceedings of the International Computer Music Conference (ICMC)*, New Orleans, Nov. 2006.

[87] T. Taguti, "Mapping a Physical Correlate of Loudness into the Velocity Space of MIDI-Controlled Piano Tones," in *Proceedings of the Stockholm Music Acoustics Conference (SMAC)*, Stockholm, Aug. 2003.

[88] IEC 61672:2002, "Electroacoustics — Sound Level Meters — Part1: Specifications," IEC, Standard, 2002.

[89] ITU-R BS.1770:2006, "Algorithms to Measure Audio Programme Loudness and True-Peak Audio Level," ITU, Recommendation, 2006.

[90] ITU-R BS.468:1986, "Measurement of Audio-Frequency Noise Voltage Level in Sound," ITU, Recommendation, 1986.

[91] U. Zölzer, *Digitale Audiosignalverarbeitung*, 2nd. Stuttgart: Teubner, 1997.

[92] G. Soulodre and S. Norcross, "Objective Measures of Loudness," in *Proceedings of the 115th Audio Engineering Society Convention, Preprint 5896*, New York: Audio Engineering Society (AES), Oct. 2003.

[93] DIN 45631:1991, "Berechnung des Lautstärkepegels und der Lautheit aus dem Geräuschspektrum," DIN, Standard, 1991.

[94] EBU — Recommendation R128, "Loudness Normalisation and Permitted Maximum Level of Audio Signals," EBU, Recommendation, 2010.

[95] B. J. Shannon and K. K. Paliwal, "A Comparative Study of Filter Bank Spacing for Speech Recognition," in *Proceedings of Microelectronic Engineering Research Conference*, Brisbane, Nov. 2003.

[96] J. Burred and T. Sikora, "On the Use of Auditory Representations for Sparsity-Based Sound Source Separation," in *5th International Conference on Information Communications & Signal Processing*, Bangkok: Institute of Electrical and Electronics Engineers (IEEE), 2005, pp. 1466–1470, ISBN: 0-7803-9283-3. DOI: 10.1109/ICICS.2005.1689302.

[97] S. S. Stevens and J. Volkmann, "The Relation of Pitch to Frequency: A Revised Scale," *The American Journal of Psychology*, vol. 53, no. 3, pp. 329–353, 1940. DOI: 10.2307/1417526.

[98] R. J. Siegel, "A Replication of the Mel Scale of Pitch," *The American Journal of Psychology*, vol. 78, no. 4, p. 615, Dec. 1965, ISSN: 00029556. DOI: 10.2307/1420924.

[99] G. Fant, *Speech Sounds and Features*. Cambridge: MIT Press, 1973, ISBN: 0-262-06051-5.

[100] D. O'Shaughnessy, *Speech Communication: Human and Machine*, reprint. Reading, MA: Addison-Wesley, 1987, ISBN: 978-0-201-16520-3.

[101] S Umesh, L Cohen, and D Nelson, "Fitting the Mel Scale," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP99)*, Phoenix: Institute of Electrical and Electronics Engineers (IEEE), 1999, pp. 217–220, ISBN: 0-7803-5041-3. DOI: 10.1109/ICASSP.1999.758101.

[102] M. R. Schroeder, B. S. Atal, and J. L. Hall, "Optimizing Digital Speech Coders by Exploiting Masking Properties of the Human Ear," *Journal of the Acoustical Society of America (JASA)*, vol. 66, no. 6, pp. 1647–1652, 1979. DOI: 10.1121/1.383662.

[103] E. Terhardt, "Calculating Virtual Pitch," *Hearing Research*, vol. 1, pp. 155–182, 1979. DOI: 10.1016/0378-5955(79)90025-X.

[104] E. Zwicker and E. Terhardt, "Analytical expressions for critical-band rate and critical bandwidth as a function of frequency," *Journal of the Acoustical Society of America (JASA)*, vol. 68, no. 5, pp. 1523–1525, 1980. DOI: 10.1121/1.385079.

[105] H. Traunmüller, "Analytical Expressions for the Tonotopic Sensory Scale," en, *Journal of the Acoustical Society of America (JASA)*, vol. 88, no. 1, pp. 97–100, Jul. 1990, ISSN: 00014966. DOI: 10.1121/1.399849.

[106] E. Terhardt, "The SPINC Function for Scaling of Frequency in Auditory Models," *Acustica*, vol. 77, pp. 40–42, 1992.

[107] D. D. Greenwood, "Critical Bandwidth and the Frequency Coordinates of the Basilar Membrane," *Journal of the Acoustical Society of America (JASA)*, vol. 33, no. 10, p. 1344, 1961, ISSN: 00014966. DOI: 10.1121/1.1908437.

[108] R. N. Shepard, "Circularity in Judgments of Relative Pitch," *Journal of the Acoustical Society of America (JASA)*, vol. 36, no. 12, p. 2346, 1964, ISSN: 00014966. DOI: 10.1121/1.1919362.

[109] D. Deutsch, K. Dooley, and T. Henthorn, "Pitch Circularity from Tones Comprising Full Harmonic Series.," *Journal of the Acoustical Society of America*, vol. 124, no. 1, pp. 589–597, Jul. 2008, ISSN: 1520-8524. DOI: 10.1121/1.2931957.

[110] ISO 16:1975, "Acoustics — Standard Tuning Frequency (Standard Musical Pitch)," ISO, Standard, 1975.

[111] E. Briner, *Musikinstrumentenführer*, 3rd. Stuttgart: Philipp Reclam, 1998.

[112] Y. Zhu, M. S. Kankanhalli, and S. Gao, "Music Key Detection for Musical Audio," in *Proceedings of the 11th International Multimedia Modelling Conference*, Melbourne, Jan. 2005. DOI: 10.1109/MMMC.2005.56.

[113] A. Lerch, "On the Requirement of Automatic Tuning Frequency Estimation," in *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, Victoria, Oct. 2006.

[114] W. A. Sethares, *Tuning, Timbre, Spectrum, Scale*. London: Springer, 2005, ISBN: 978-1-85233-797-1.

[115] M. Schoen, *The Psychology of Music*. New York: Ronald Press, 1940.

[116] N. H. Fletcher, "Acoustical Correlates of Flute Performance Technique," *Journal of the Acoustical Society of America (JASA)*, vol. 57, no. 1, pp. 233–237, 1975. DOI: 10.1121/1.380430.

[117] N. H. Fletcher and T. D. Rossing, *The Physics of Musical Instruments*, 2nd. New York: Springer, 1998, ISBN: 9780387983745.

[118] B Yegnanarayana, C. D'Alessandro, and V. Darsinos, "An Iterative Algorithm for Decomposition of Speech Signals into Periodic and Aperiodic Components," *Transactions on Speech and Audio Processing*, vol. 6, no. 1, pp. 1–11, 1998, ISSN: 10636676. DOI: 10.1109/89.650304.

[119] T. W. Parsons, "Separation of Speech from Interfering Speech by Means of Harmonic Selection," *Journal of the Acoustical Society of America (JASA)*, vol. 60, no. 4, p. 911, 1976, ISSN: 00014966. DOI: 10.1121/1.381172.

[120] E. Terhardt, G. Stoll, and M. Seewann, "Algorithm for Extraction of Pitch and Pitch Salience from Complex Tonal Signals," *Journal of the Acoustical Society of America (JASA)*, vol. 71, no. 3, p. 679, 1982, ISSN: 00014966. DOI: 10.1121/1.387544.

[121] X. Serra, "A System for Sound Analysis/Transformation/Synthesis Based on a Deterministic Plus Stochastic Decomposition," Dissertation, Stanford University, Stanford, 1989.

[122] G. Peeters and X. Rodet, "Signal Characterization in Terms of Sinusoidal and Non-Sinusoidal Components," in *Proceedings of the 1st Conference on Digital Audio Effects (DAFX)*, Barcelona, 1998.

[123] M. Lagrange, S. Marchand, and J.-B. Rault, "Sinusoidal Parameter Extraction and Component Selection in a Non Stationary Model," in *Proceedings of the 5th International Conference on Digital Audio Effects (DAFX)*, Hamburg, 2002.

[124] M. R. Every, "Separation of Musical Sources and Structure from Single-Channel Polyphonic Recordings," Dissertation, University of York, 2006.

[125] A. Röbel, M. Zivanovic, and X. Rodet, "Signal Decomposition by Means of Classification of Spectral Peaks," in *Proceedings of the International Computer Music Conference (ICMC)*, Miami: International Computer Music Association (ICMA), 2004.

[126] M. Kulesza and A. Czyzewski, "Frequency Based Criterion for Distinguishing Tonal and Noisy Spectral Components," *International Journal of Computer Science and Security*, vol. 4, no. 1, p. 1, Mar. 2010.

[127] S. Haykin, *Neural Networks — A Comprehensive Foundation*. Upper Saddle River, NJ: Prentice Hall, 1994, ISBN: 0-02-352761-7.

[128] L. Rabiner and R. Schafer, *Digital Processing of Speech Signals*. Englewood Cliffs, NJ: Prentice Hall, 1978.

[129] A. Röbel and X. Rodet, "Efficient Spectral Envelope Estimation and Its Application to Pitch Shifting and Envelope Preservation," in *Proceedings of the 8th International Conference on Digital Audio Effects (DAFX)*, Madrid, Sep. 2005.

[130] M. J. Ross, H. L. Shaffer, A. Cohen, R. Freudberg, and H. J. Manley, "Average Magnitude Difference Function Pitch Extractor," *Transactions on Acoustics, Speech, and Signal Processing*, vol. 22, no. 5, pp. 353–362, Oct. 1974, ISSN: 0096-3518. DOI: 10.1109/TASSP.1974.1162598.

[131] A. Cheveigné and H. Kawahara, "YIN, a Fundamental Frequency Estimator for Speech and Music," *Journal of the Acoustical Society of America (JASA)*, vol. 111, no. 4, pp. 1917–1930, 2002, ISSN: 00014966. DOI: 10.1121/1.1458024.

[132] T. Shimamura and H. Kobayashi, "Weighted Autocorrelation for Pitch Extraction of Noisy Speech," *Transactions on Speech and Audio Processing*, vol. 9, no. 7, pp. 727–730, 2001, ISSN: 10636676. DOI: 10.1109/89.952490.

[133] M. R. Schroeder, "Period Histogram and Product Spectrum: New Methods for Fundamental-Frequency Measurement," *Journal of the Acoustical Society of America (JASA)*, vol. 43, no. 4, p. 829, 1968, ISSN: 00014966. DOI: 10.1121/1.1910902.

[134] A. M. Noll, "Pitch Determination of Human Speech by the Harmonic Product Spectrum, the Harmonic Sum Spectrum, and a Maximum Likelihood Estimate," in *Proceedings of the Symposium on Computer Processing in Communications*, vol. 19, Brooklyn: Polytechnic Press of the University of Brooklyn, 1969, pp. 779–797.

[135] ——, "Short-Time Spectrum and "Cepstrum" Techniques for Vocal-Pitch Detection," *Journal of the Acoustical Society of America (JASA)*, vol. 36, no. 2, p. 296, 1964, ISSN: 00014966. DOI: 10.1121/1.1918949.

[136] R. Meddis and L. O'Mard, "A Unitary Model of Pitch Perception," *Journal of the Acoustical Society of America (JASA)*, vol. 102, no. 3, pp. 1811–20, Sep. 1997, ISSN: 0001-4966. DOI: 10.1121/1.420088.

[137] A. Klapuri, "Auditory Model-Based Methods for Multiple Fundamental Frequency Estimation," in *Signal Processing Methods for Music Transcription*, A. Klapuri and M. Davy, Eds., Berlin: Springer, 2006, pp. 229–265. DOI: 10.1007/0-387-32845-9_8.

[138] C. Chafe, D. Jaffe, K. Kashima, B. Mont-Reynaud, and J. O. Smith, "Techniques for Note Identification in Polyphonic Music," in *Proceedings of the International Computer Music Conference (ICMC)*, International Computer Music Association (ICMA), 1985.

[139] A. Klapuri, T. Virtanen, and J.-M. Holm, "Robust Multipitch Estimation for the Analysis and Manipulation of Polyphonic Musical Signals," in *Proceedings of COST-G6 Conference on Digital Audio Effects (DAFX)*, Verona, Dec. 2000.

[140] A. P. Klapuri, "Multipitch Estimation and Sound Separation by the Spectral Smoothness Principle," *Transactions on Speech and Audio Processing*, vol. 11, no. 6, pp. 804–816, Nov. 2003, ISSN: 1063-6676. DOI: 10.1109/TSA.2003.815516.

[141] A. Cheveigné and H. Kawahara, "Multiple Period Estimation and Pitch Perception Model," *Speech Communication*, vol. 27, pp. 175–185, 1999. DOI: 10.1016/S0167-6393(98)00074-0.

[142] A. Cheveigné, "Multiple f0 estimation," in *Computational Auditory Scene Analysis*, D. Wang and G. Brown, Eds., Hoboken, NJ: IEEE Press/Wiley, 2006, pp. 45–79. DOI: 10.1109/9780470043387.ch2.

[143] R. Meddis and M. J. Hewitt, "Modeling the Identification of Concurrent Vowels with Different Fundamental Frequencies," *Journal of the Acoustical Society of America (JASA)*, vol. 91, no. 1, pp. 233–245, Jan. 1992, ISSN: 0001-4966. DOI: 10.1121/1.402767.

[144] M. Karjalainen and T. Tolonen, "Multi-Pitch and Periodicity Analysis Model for Sound Separation and Auditory Scene Analysis," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Phoenix: Institute of Electrical and Electronics Engineers (IEEE), Mar. 1999. DOI: 10.1109/ICASSP.1999.759824.

[145] A. Klapuri, "A Perceptually Motivated Multiple-F0 Estimation Method," in *Proceedings of the Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz: Institute of Electrical and Electronics Engineers (IEEE), Oct. 2005. DOI: 10.1109/ASPAA.2005.1540227.

[146] H. Kameoka, T. Nishimoto, and S. Sagayama, "A Multipitch Analyzer Based on Harmonic Temporal Structured Clustering," *Transactions on Audio, Speech and Language Processing*, vol. 15, no. 3, pp. 982–994, Mar. 2007, ISSN: 1558-7916. DOI: 10.1109/TASL.2006.885248.

[147] D. D. Lee and H. S. Seung, "Algorithms for Non-Negative Matrix Factorization," *Advances in Neural Information Processing Systems*, vol. 13, pp. 556–562, 2001.

[148] P. Smaragdis and J. C. Brown, "Non-Negative Matrix Factorization for Polyphonic Music Transcription," in *Proceedings of the Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA))*, New Paltz: Institute of Electrical and Electronics Engineers (IEEE), 2003, ISBN: 0-7803-7850-4. DOI: 10.1109/ASPAA.2003.1285860.

[149] E. Scheirer, "Extracting Expressive Performance Information from Recorded Music," Master's Thesis, Massachusetts Institute of Technology, Sep. 1995.

[150] S. Dixon, "A Dynamic Modelling Approach to Music Recognition," in *Proceedings of the International Computer Music Conference (ICMC)*, Hong Kong, Aug. 1996.

[151] M. Ryynänen, "Probabilistic Modelling of Note Events in the Transcription of Monophonic Melodies," Master's Thesis, Tampere University of Technology, Mar. 2004.

[152] K. Dressler and S. Streich, "Tuning Frequency Estimation using Circular Statistics," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Vienna, Sep. 2007.

[153] A. Lerch, "Ein Ansatz zur automatischen Erkennung der Tonart in Musikdateien," in *Proceedings of the VDT International Audio Convention (23. Tonmeistertagung)*, Leipzig, Nov. 2004.

[154] M. Riedmiller and H. Braun, "A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm," in *Proceedings of the International Conference on Neural Networks*, San Francisco: Institute of Electrical and Electronics Engineers (IEEE), Mar. 1993. DOI: 10.1109/ICNN.1993.298623.

[155] C. Krumhansl, *Cognitive Foundations of Musical Pitch*. New York: Oxford University Press, 1990.

[156] M. A. Bartsch and G. H. Wakefield, "To Catch a Chorus: Using Chroma-Based Representations for Audio Thumbnailing," in *Proceedings of the Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz: Institute of Electrical and Electronics Engineers (IEEE), Oct. 2001. DOI: 10.1109/ASPAA.2001.969531.

[157] G. Tzanetakis, A. Ermolinskyi, and P. Cook, "Pitch Histograms in Audio and Symbolic Music Information Retrieval," in *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, Paris, Oct. 2002.

[158] M. Cremer and C. Derboven, "A System for Harmonic Analysis of Polyphonic Music," in *Proceedings of the 25th International Audio Engineering Society Conference: Metadata for Audio*, Audio Engineering Society (AES), Jun. 2004.

[159] A. V. Oppenheim, D. H. Johnson, and K Steiglitz, "Computation of Spectra with Unequal Resolution Using the Fast Fourier Transform," *Proceedings of the IEEE*, vol. 59, no. 2, pp. 299–301, 1971, ISSN: 0018-9219. DOI: 10.1109/PROC.1971.8146.

[160] H. Purwins, B. Blankertz, and K. Obermayer, "A New Method for Tracking Modulations in Tonal Music in Audio Data Format," in *IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)*, vol. 6, Como: Institute of Electrical and Electronics Engineers (IEEE), 2000, pp. 6270–6275. DOI: 10.1109/IJCNN.2000.859408.

[161] H. Papadopoulos and G. Peeters, "Large-Scale Study of Chord Estimation Algorithms Based on Chroma Representation and HMM," in *Proceedings of the International Workshop on Content-Based Multimedia Indexing (CBMI)*, Bordeaux, 2007. DOI: 10.1109/CBMI.2007.385392.

[162] J. Weil and J.-L. Durrieu, "An HMM-Based Audio Chord Detection System Attenuating the Main Melody," in *Proceedings of the Music Information Retrieval EXchange (MIREX) at the 9th International Conference on Music Information Retrieval (ISMIR)*, Philadelphia, Sep. 2008.

[163] J. Pauwels, M. Varewyck, and J.-P. Martens, "Audio Chord Extraction Using a Probabilistic Model," in *Proceedings of the Music Information Retrieval EXchange (MIREX) at the 9th International Conference on Music Information Retrieval (ISMIR)*, Philadelphia, Sep. 2008.

[164] J. Bello and J. Pickens, "A Robust Mid-level Representation for Harmonic Content in Music Signals," in *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, London, Sep. 2005.

[165] C. A. Harte, M. B. Sandler, and M. Gasser, "Detecting Harmonic Change in Musical Audio," in *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia (AMCMM)*, Santa Barbara: ACM, Oct. 2006. DOI: 10.1145/1178723.1178727.

[166] S. Pauws, "Musical Key Extraction from Audio," in *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Oct. 2004.

[167] M. Varewyck, J. Pauwels, and J.-P. Martens, "A Novel Chroma Representation of Polyphonic Music Based on Multiple Pitch Tracking Techniques," in *Proceedings of the 16th ACM International Conference on Multimedia (MM)*, New York: ACM Press, 2008, p. 667. DOI: 10.1145/1459359.1459455.

[168] M. P. Ryynänen and A. P. Klapuri, "Automatic Transcription of Melody, Bass Line, and Chords in Polyphonic Music," *Computer Music Journal*, vol. 32, no. 3, pp. 72–86, Sep. 2008, ISSN: 0148-9267. DOI: 10.1162/comj.2008.32.3.72.

[169] D. Temperley, "The Tonal Properties of Pitch-Class Sets: Tonal Implication, Tonal Ambiguity, and Tonalness," *Computing in Musicology*, vol. 15, pp. 24–38, 2007.

[170] A. J. Milne, W. A. Sethares, R. Laney, and D. B. Sharp, "Modelling the Similarity of Pitch Collections with Expectation Tensors," *Journal of Mathematics and Music*, vol. 5, no. 1, pp. 1–20, Mar. 2011, ISSN: 1745-9737. DOI: 10.1080/17459737.2011.573678.

[171] E. Chew, "Towards a Mathematical Model of Tonality," Dissertation, Massachusetts Institute of Technology, 1998.

[172] C.-H. Chuan and E. Chew, "Polyphonic Audio Key Finding Using the Spiral Array CEG Algorithm," in *International Conference on Multimedia and Expo*, Institute of Electrical and Electronics Engineers (IEEE), 2005, pp. 21–24, ISBN: 0-7803-9331-7. DOI: 10.1109/ICME.2005.1521350.

[173] G. Gatzsche, M. Mehnert, and C. Stöcklmeier, "Interaction with Tonal Pitch Spaces," in *Proceedings of the 8th International Conference on New Interfaces for Musical Expression (NIME)*, Genova, Jun. 2008.

[174] O. Izmirli, "Template Based Key Finding from Audio," in *Proceedings of the International Computer Music Conference (ICMC)*, Barcelona, Sep. 2005.

[175] K. Lee and M. Slaney, "Automatic Chord Recognition from Audio Using a Supervised HMM Trained with Audio-from-Symbolic Data," in *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia (AMCMM)*, Santa Barbara: ACM, Oct. 2006. DOI: 10.1145/1178723.1178726.

[176] ——, "A Unified System for Chord Transcription and Key Extraction Using Hidden Markov Models," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Vienna, Sep. 2007.

[177] E. W. Large and C. Palmer, "Perceiving Temporal Regularity in Music," *Cognitive Science*, vol. 26, no. 1, pp. 1–37, Jan. 2002, ISSN: 03640213. DOI: 10.1207/s15516709cog2601_1.

[178] M. Wright, "The Shape of an Instant: Measuring and Modeling Perceptual Attack Time with Probability Density Functions," Dissertation, Stanford University, Stanford, 2008.

[179] J. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. Sandler, "A Tutorial on Onset Detection in Music Signals," *Transactions on Speech and Audio Processing*, vol. 13, no. 5 Part 2, pp. 1035–1047, 2005. DOI: 10.1109/TSA.2005.851998.

[180] B. H. Repp, "Patterns of Note Onset Asynchronies in Expressive Piano Performance," *Journal of the Acoustical Society of America (JASA)*, vol. 100, no. 6, pp. 3917–3932, 1996. DOI: 10.1121/1.417245.

[181] J. W. Gordon, "Perception of Attack Transients in Musical Tones," Dissertation, Stanford University, Center for Computer Research in Music and Acoustics (CCRMA), Stanford, 1984.

[182] I. J. Hirsh, "Auditory Perception of Temporal Order," *Journal of the Acoustical Society of America (JASA)*, vol. 31, no. 6, pp. 759–767, 1959. DOI: 10.1121/1.1907782.

[183] A. Friberg and J. Sundberg, "Perception of Just Noticeable Time Displacement of a Tone Presented in a Metrical Sequence at Different Tempos," *STL-QPSR*, vol. 33, no. 4, pp. 97–108, 1992. DOI: 10.1121/1.407650.

[184] B. H. Repp, "Diversity and Commonality in Music Performance: An Analysis of Timing Microstructure in Schumann's "Träumerei"," *Journal of the Acoustical Society of America (JASA)*, vol. 92, no. 5, pp. 2546–2568, 1992. DOI: 10.1121/1.404425.

[185] P. Levau, L. Daudet, and G. Richard, "Methodology and Tools for the Evaluation of Automatic Onset Detection Algorithms in Music," in *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Oct. 2004.

[186] R. A. Rasch, "Synchronization in Performed Ensemble Music," *Acustica*, vol. 43, pp. 121–131, 1979.

[187] L. Shaffer, "Timing in Solo and Duet Piano Performances," *The Quarterly Journal of Experimental Psychology*, vol. 36A, pp. 577–595, 1984. DOI: 10.1080/14640748408402180.

[188] P. Desain and H. Honing, "Tempo Curves Considered Harmful," *Time in Contemporary Musical Thought, Contemporary Music Review*, vol. 7, no. 2, J. D. Kramer, Ed., pp. 123–138, 1993. DOI: 10.1080/07494469300640081.

[189] F. Lerdahl and R. Jackendorf, *A Generative Theory of Tonal Music*. Cambridge: MIT Press, 1983, ISBN: 978-0-262-62107-6.

[190] C. Uhle and J. Herre, "Estimation of Tempo, Micro Time and Time Signature from Percussive Music," in *Proceedings of the 6th International Conference on Digital Audio Effects (DAFX)*, London, 2003.

[191] R. Parncutt, "Accents and Expression in Piano Performance," in *Perspektiven und Methoden einer Systemischen Musikwissenschaft*, K. W. Niemöller, Ed., Frankfurt/Main: Peter Lang, 2003.

[192] A. Gabrielsson, "The Performance of Music," in *The Psychology of Music*, D. Deutsch, Ed., 2nd, San Diego: Academic Press, 1999.

[193] B. Repp, "On Determining the Basic Tempo of an Expressive Music Performance," *Psychology of Music*, vol. 22, no. 2, pp. 157–167, 1994. DOI: 10.1177/030573564222005.

[194] W. Goebl and S. Dixon, "Analysis of Tempo Classes in Performances of Mozart Sonatas," in *Proceedings of the 7th International Symposium on Systematic and Comparative Musicology (ISSCM), 3rd International Conference on Cognitive Musicology (ICCM)*, H Lappalainen, Ed., Jyväskyla, Jun. 2001.

[195] M. F. McKinney and D. Moelants, "Deviations from the Resonance Theory of Tempo Induction," in *Proceedings of the Conference on Interdisciplinary Musicology*, R. Parncutt, A Kessler, and F Zimmer, Eds., Graz, 2004.

[196] P. Desain, "A (De)Composable Theory of Rhythm Perception," *Music Perception*, vol. 9, no. 4, pp. 439–454, 1992.

[197] V. Iyer, J. Bilmes, M. Wright, and D. L. Wessel, "A Novel Representation for Rhythmic Structure," in *Proceedings of the International Computer Music Conference (ICMC)*, Thessaloniki: International Computer Music Association (ICMA), 1997, pp. 1–5.

[198] L. Hofmann-Engl, "Rhythmic Similarity: A Theoretical and Empirical Approach," in *Proceedings of the 7th International Conference on Music Perception and Cognition (ICMPC)*, Sydney, 2002, pp. 564–567.

[199] S. Dixon, "Onset Detection Revisited," in *Proceedings of the 9th International Conference on Digital Audio Effects (DAFX)*, Montreal, Sep. 2006.

[200] J. Foote, "Automatic Audio Segmentation Using a Measure of Audio Novelty," in *Proceedings of the International Conference on Multimedia and Expo (ICME)*, New York: Institute of Electrical and Electronics Engineers (IEEE), Jul. 2000, pp. 452–455. DOI: 10.1109/ICME.2000.869637.

[201] S. Hainsworth and M. Macleod, "Onset Detection in Musical Audio Signals," in *Proceedings of the International Computer Music Conference (ICMC)*, Singapore, Sep. 2003.

[202] W. A. Schloss, "On the Automatic Transcription of Percussive Music — From Acoustic Signal to High-Level Analysis," Dissertation, Stanford University, Center for Computer Research in Music and Acoustics (CCRMA), Stanford, 1985.

[203] E. D. Scheirer, "Tempo and Beat Analysis of Acoustic Musical Signals," *Journal of the Acoustical Society of America (JASA)*, vol. 103, no. 1, pp. 588–601, 1998. DOI: 10.1121/1.421129.

[204] A. Klapuri, "Sound Onset Detection by Applying Psychoacoustic Knowledge," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Phoenix: Institute of Electrical and Electronics Engineers (IEEE), Mar. 1999. DOI: 10.1109/ICASSP.1999.757494.

[205] J. Laroche, "Efficient Tempo and Beat Tracking in Audio Recordings," *Journal of the Audio Engineering Society (JAES)*, vol. 51, no. 4, pp. 226–233, 2003.

[206] C. Duxbury, J. Bello, M. Davies, and M. Sandler, "Complex Domain Onset Detection for Musical Signals," in *Proceedings of the 6th International Conference on Digital Audio Effects (DAFX)*, London, Oct. 2003.

[207] J. Bello, G. Monti, and M. Sandler, "Phase-Based Note Onset Detection for Music Signals," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 5, Hong Kong: Institute of Electrical and Electronics Engineers (IEEE), Mar. 2003. DOI: 10.1109/ICASSP.2003.1200001.

[208] M. Goto and Y. Muraoka, "Music Understanding at the Beat Level — Real-Time Beat Tracking for Audio Signals," in *Proceedings of the Workshop on Computational Auditory Scene Analysis (IJCAI)*, Montreal, Aug. 1995.

[209] A. Röbel, "Onset Detection in Polyphonic Signals by means of Transient Peak Classification," in *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, London, Sep. 2005.

[210] R. Zhou and J. Reiss, "Music Onset Detection Combining Energy-Based and Pitch-Based Approaches," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Vienna, Sep. 2007.

[211] T. Fawcett, "An Introduction to ROC Analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, Jun. 2006, ISSN: 01678655. DOI: 10.1016/j.patrec.2005.10.010.

[212] A. T. Cemgil, P. Desain, and B. Kappen, "Rhythm Quantization for Transcription," *Computer Music Journal*, vol. 24, no. 2, pp. 60–76, 2000. DOI: 10.1162/014892600559218.

[213] R. Liu, N. Griffith, J. Walker, and P. Murphy, "Time Domain Note Average Energy Based Music Onset Detection," in *Proceedings of the Stockholm Music Acoustics Conference (SMAC)*, Stockholm, Aug. 2003.

[214] A. Lerch and I. Klich, "On the Evaluation of Automatic Onset Tracking Systems," zplane.development, Berlin, White Paper, 2005, Last retrieved on Dec. 12, 2011. [Online]. Available: https://www.zplane.de/downloads/onset_eval.pdf.

[215] A. Cemgil, B. Kappen, P. Desain, and H. Honing, "On Tempo Tracking: Tempogram Representation and Kalman Filtering," *Journal of New Music Research*, vol. 28, no. 4, pp. 259–273, 2001. DOI: 10.1080/09298210008565462.

[216] J. Berkson, "Tests of Significance Considered as Evidence," *Journal of the American Statistical Association*, vol. 37, no. 219, pp. 325–335, 1942. DOI: 10.2307/2279000.

[217] J. Foote and S. Uchihashi, "The Beat Spectrum: A New Approach to Rhythm Analysis," in *Proceedings of the International Conference on Multimedia and Expo (ICME)*, Tokyo: Institute of Electrical and Electronics Engineers (IEEE), Aug. 2001, pp. 881–884. DOI: 10.1109/ICME.2001.1237863.

[218] J. Burred and A. Lerch, "Hierarchical Automatic Audio Signal Classification," *Journal of the Audio Engineering Society (JAES)*, vol. 52, no. 7/8, pp. 724–739, 2004.

[219] P. E. Allen and R. B. Dannenberg, "Tracking Musical Beats in Real Time," in *Proceedings of the International Computer Music Conference (ICMC)*, Glasgow, Sep. 1990, pp. 140–143.

[220] E. Large, "Beat Tracking with a Nonlinear Oscillator," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, Montreal, Aug. 1995.

[221] D. Rosenthal, M. Goto, and Y. Muraoka, "Rhythm Tracking Using Multiple Hypotheses," in *Proceedings of the International Computer Music Conference (ICMC)*, Aarhus, Sep. 1994.

[222] M. Goto and Y. Muraoka, "Beat Tracking Based on Multiple-agent Architecture — A Real-time Beat Tracking System for Audio Signals," in *Proceedings of the Second International Conference on Multiagent Systems (ICMAS)*, Dec. 1996.

[223] ——, "Real-Time Beat Tracking for Drumless Audio Signals: Chord Change Detection for Musical Decisions," *Speech Communication*, vol. 27, pp. 311–335, 1999. DOI: 10.1016/S0167-6393(98)00076-4.

[224] A. P. Klapuri, "Musical Meter Estimation and Music Transcription," in *Proceedings of the Cambridge Music Processing Colloquium*, Cambridge, 2003.

[225] S. Dixon, "A Beat Tracking System for Audio Signals," in *Proceedings of the Conference on Mathematical and Computational Methods in Music*, Vienna, Dec. 1999.

[226] ——, "A Lightweight Multi-Agent Musical Beat Tracking System," in *Proceedings of the Pacific Rim International Conference on Artificial Intelligence (PRICAI)*, Melbourne, Aug. 2000. DOI: 10.1007/3-540-44533-1_77.

[227] S. Dixon and E. Cambouropoulos, "Beat Tracking with Musical Knowledge," in *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI)*, Berlin, Aug. 2000.

[228] B. Meudic, "A Causal Algorithm for Beat Tracking," in *Proceedings of the 2nd International Conference on Understanding and Creating Music*, Caserta, Nov. 2002.

[229] F. Gouyon and P. Herrera, "A Beat Induction Method for Musical Audio Signals," in *Proceedings of the 4th European Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, London, Apr. 2003. DOI: 10.1142/9789812704337_0051.

[230] G. Peeters, "Time Variable Tempo Detection and Beat Marking," in *Proceedings of the International Computer Music Conference (ICMC)*, Barcelona, Sep. 2005.

[231] A. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *Transactions on Information Theory*, vol. 13, pp. 260–269, 1967. DOI: 10.1109/TIT.1967.1054010.

[232] J. Brown, "Determination of the Meter of Musical Scores by Autocorrelation," *Journal of the Acoustical Society of America (JASA)*, vol. 94, no. 4, pp. 1953–1957, 1993. DOI: 10.1121/1.407518.

[233] P. Toiviainen and T. Eerola, "Autocorrelation in Meter Induction: The Role of Accent Structure," *Journal of the Acoustical Society of America (JASA)*, vol. 119, no. 2, pp. 1164–1170, 2006. DOI: 10.1121/1.2146084.

[234] B. Escalona-Espinosa, "Downbeat and Meter Estimation in Audio Signals," Master's Thesis, Technische Universität Hamburg-Harburg, 2008.

[235] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," *Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978. DOI: 10.1109/TASSP.1978.1163055.

[236] M. Müller, *Information Retrieval for Music and Notion*. Berlin: Springer, 2007, ISBN: 978-3-540-74047-6.

[237] F. Itakura, "Minimum Prediction Residual Principle Applied to Speech Recognition," *Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, no. 1, pp. 67–72, Feb. 1975, ISSN: 0096-3518. DOI: 10.1109/TASSP.1975.1162641.

[238] S. Dixon and G. Widmer, "MATCH: A Music Alignment Tool Chest," in *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, London, Sep. 2005.

[239] R. Turetsky and D. Ellis, "Ground-Truth Transcriptions of Real Music from Force-Aligned MIDI Syntheses," in *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, Baltimore, Oct. 2003.

[240] M. Müller, H. Mattes, and F. Kurth, "An Efficient Multiscale Approach to Audio Synchronization," in *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, Victoria, Oct. 2006.

[241] N. Hu, R. B. Dannenberg, and G. Tzanetakis, "Polyphonic Audio Matching and Alignment for Music Retrieval," in *Proceedings of the Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz: Institute of Electrical and Electronics Engineers (IEEE), Oct. 2003. DOI: 10.1109/ASPAA.2003.1285862.

[242] H. Kirchhoff and A. Lerch, "Evaluation of Features for Audio-to-Audio Alignment," *Journal of New Music Research*, vol. 40, no. 1, pp. 27–41, 2011. DOI: 10.1080/09298215.2010.529917.

[243] G. Evangelista, "Time and Frequency Warping Musical Signals," in *DAFX — Digital Audio Effects*, U. Zölzer, Ed., Hoboken, NJ: Wiley, 2002, ch. 11, pp. 439–463, ISBN: 0-471-49078-4.

[244] F. Soulez, X. Rodet, and D. Schwarz, "Improving Polyphonic and Poly-Instrumental Music to Score Alignment," in *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, Baltimore, Oct. 2003.

[245] R. B. Dannenberg, "An On-Line Algorithm for Real-Time Accompaniment," in *Proceedings of the International Computer Music Conference (ICMC)*, Paris, 1984.

[246] B. Vercoe, "The Synthetic Performer in the Context of Live Performance," in *Proceedings of the International Computer Music Conference (ICMC)*, Paris, 1984, pp. 199–200.

[247] J. J. Bloch and R. B. Dannenberg, "Real-Time Computer Accompaniment of Keyboard Performances," in *Proceedings of the International Computer Music Conference (ICMC)*, Vancouver, Oct. 1985.

[248] R. B. Dannenberg and H. Mukaino, "New Techniques for Enhanced Quality of Computer Accompaniment," in *Proceedings of the International Computer Music Conference (ICMC)*, Cologne, Sep. 1988.

[249] B. Vercoe and M. Puckette, "Synthetic Rehearsal: Training the Synthetic Performer," in *Proceedings of the International Computer Music Conference (ICMC)*, Vancouver, 1985, pp. 275–278.

[250] B. Baird, D. Blevins, and N. Zahler, "The Artificially Intelligent Computer Performer: The Second Generation," *Interface — Journal of New Music Research*, vol. 19, pp. 197–204, 1990. DOI: 10.1080/09298219008570566.

[251] ——, "Artificial Intelligence and Music: Implementing an Interactive Computer Performer," *Computer Music Journal*, vol. 17, no. 2, pp. 73–79, 1993.

[252] H. Heijink, "Matching Scores and Performances," M.A. Thesis, Nijmegen University, Jun. 1996.

[253] P. Desain, H. Honing, and H. Heijink, "Robust Score-Performance Matching: Taking Advantage of Structural Information," in *Proceedings of the International Computer Music Conference (ICMC)*, Thessaloniki: International Computer Music Association (ICMA), Sep. 1997.

[254] M. Puckette and C. Lippe, "Score Following in Practice," in *Proceedings of the International Computer Music Conference (ICMC)*, San Francisco, 1992.

[255] M. Puckette, "Score Following Using the Sung Voice," in *Proceedings of the International Computer Music Conference (ICMC)*, Banff, 1995.

[256] J. D. Vantomme, "Score Following by Temporal Pattern," *Computer Music Journal*, vol. 19, no. 3, pp. 50–59, 1995.

[257] L. Grubb and R. Dannenberg, "A Stochastic Method of Tracking a Vocal Performer," in *Proceedings of the International Computer Music Conference (ICMC)*, Thessaloniki, Sep. 1997.

[258] L. Grubb and R. B. Dannenberg, "Enhanced Vocal Performance Tracking Using Multiple Information Sources," in *Proceedings of the International Computer Music Conference (ICMC)*, Ann Arbor, Oct. 1998.

[259] C. Raphael, "Automatic Segmentation of Acoustic Musical Signals Using Hidden Markov Models," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 4, pp. 360–370, 1999. DOI: 10.1109/34.761266.

[260] ——, "A Probabilistic Expert System for Automatic Musical Accompaniment," *Journal of Computational and Graphical Statistics*, vol. 10, no. 3, pp. 487–512, 2001. DOI: 10.1198/106186001317115081.

[261] ——, "A Hybrid Graphical Model for Aligning Polyphonic Audio with Musical Scores," in *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Oct. 2004.

[262] P. Cano, A. Loscos, and J. Bonada, "Score-Performance Matching Using HMMs," in *Proceedings of the International Computer Music Conference (ICMC)*, Beijing, Oct. 1999.

[263] N. Orio and F. Déchelle, "Score Following Using Spectral Analysis and Hidden Markov Models," in *Proceedings of the International Computer Music Conference (ICMC)*, Habana, Sep. 2001.

[264] N. Orio, S. Lemouton, and D. Schwarz, "Score Following: State of the Art and New Developments," in *Proceedings of the Conference of New Interfaces for Musical Expression (NIME)*, Montreal, May 2003.

[265] A. Cont, "Realtime Audio to Score Alignment for Polyphonic Music Instruments Using Sparse Non-Negative Constraints and Hierarchical HMMs," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 5, Toulouse: Institute of Electrical and Electronics Engineers (IEEE), May 2006. DOI: 10.1109/ICASSP.2006.1661258.

[266] E. Large, "Dynamic Programming for the Analysis of Serial Behaviors," *Behavior Research Methods, Instruments, and Computers*, vol. 25, no. 2, pp. 238–241, 1993. DOI: 10.3758/BF03204504.

[267] N. Orio and D. Schwarz, "Alignment of Monophonic and Polyphonic Music to a Score," in *Proceedings of the International Computer Music Conference (ICMC)*, Habana, Sep. 2001.

[268] Y. Meron and K. Hirose, "Automatic Alignment of a Musical Score to Performed Music," *Acoustical Science & Technology*, vol. 22, no. 3, pp. 189–198, 2001. DOI: 10.1250/ast.22.189.

[269] V. Arifi, "Algorithmen zur Synchronisation von Musikdateien im Partitur-, MIDI- und PCM-Format," Dissertation, Rheinische Friedrich-Wilhelms-Universität, Bonn, 2002.

[270] V. Arifi, M. Clausen, F. Kurth, and M. Müller, "Score-PCM Music Synchronization Based on Extracted Score Parameters," in *Proceedings of the 2nd International Symposium on Computer Music Modeling and Retrieval (CMMR)*, Trondheim, Jun. 2004. DOI: 10.1007/978-3-540-31807-1_15.

[271] R. B. Dannenberg and N. Hu, "Polyphonic Audio Matching for Score Following and Intelligent Audio Editors," in *Proceedings of the International Computer Music Conference (ICMC)*, Singapore, Sep. 2003.

[272] S. Shalev-Shwartz, J. Keshet, and Y. Singer, "Learning to Align Polyphonic Music," in *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Oct. 2004.

[273] M. Müller, F. Kurth, and T. Röder, "Towards an Efficient Algorithm for Automatic Score-to-Audio Synchronization," in *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, Barcelona, Oct. 2004.

[274] A. Lerch, *Software-Based Extraction of Objective Parameters from Music Performances*. München: GRIN, 2009, ISBN: 978-3-640-29496-1.

[275] J. Saunders, "Real-Time Discrimination of Broadcast Speech/Music," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Atlanta: Institute of Electrical and Electronics Engineers (IEEE), May 1996. DOI: 10.1109/ICASSP.1996.543290.

[276] E. Scheirer and M. Slaney, "Construction and Evaluation of a Robust Multifeature Speech/Music Discriminator," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Munich: Institute of Electrical and Electronics Engineers (IEEE), Apr. 1997. DOI: 10.1109/ICASSP.1997.596192.

[277] G. Williams and D. Ellis, "Speech/Music Discrimination Based on Posterior Probability Features," in *Proceedings of the 6th European Conference on Speech Communication and Technology (EUROSPEECH)*, Budapest, Sep. 1999.

[278] M. Carey, E. Parris, and H. Lloyd-Thomas, "A Comparison of Features for Speech, Music Discrimination," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Phoenix: Institute of Electrical and Electronics Engineers (IEEE), Mar. 1999. DOI: 10.1109/ICASSP.1999.758084.

[279] K. El-Maleh, M. Klein, G. Petrucci, and P. Kabal, "Speech/Music Discrimination for Multimedia Applications," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Istanbul: Institute of Electrical and Electronics Engineers (IEEE), Jun. 2000. DOI: 10.1109/ICASSP.2000.859336.

[280] E. Wold, T. Blum, D. Keislar, and J. Wheaton, "Content-Based Classification, Search, and Retrieval of Audio," *IEEE Multimedia Magazine*, vol. 3, no. 3, pp. 27–36, 1996. DOI: 10.1109/93.556537.

[281] J. Foote, "A Similarity Measure for Automatic Audio Classification," in *Proceedings of the 14th National Conference on Artificial Intelligence*, Providence, Jul. 1997.

[282] N. Scaringella, G. Zoia, and D. Mlynek, "Automatic Genre Classification of Music Content: A Survey," *Signal Processing Magazine*, vol. 23, no. 2, pp. 133–141, 2006. DOI: 10.1109/MSP.2006.1598089.

[283] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "A Survey of Audio-Based Music Classification and Annotation," *Transactions on Multimedia*, vol. 13, no. 2, pp. 303–319, Apr. 2011, ISSN: 1520-9210. DOI: 10.1109/TMM.2010.2098858.

[284] F. Pachet and D. Cazaly, "A Taxonomy of Musical Genres," in *Proceedings of the Conference on Content-Based Multimedia Information Access*, Paris, Apr. 2000.

[285] H. Soltau, "Erkennung von Musikstilen," Diploma Thesis, Universität Karlsruhe, Karlsruhe, 1997.

[286] S. Lippens, J.-P. Martens, T. Mulder, and G. Tzanetakis, "A Comparison of Human and Automatic Musical Genre Classification," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Montreal: Institute of Electrical and Electronics Engineers (IEEE), May 2004. DOI: 10.1109/ICASSP.2004.1326806.

[287] T Lambrou, P Kudumakis, R Speller, M. B. Sandler, and A Linney, "Classification of Audio Signals Using Statistical Features on Time and Wavelet Transform Domains," *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 6, no. Mdc, pp. 3621–3624, 1998. DOI: 10.1109/ICASSP.1998.679665.

[288] Z. Liu, Y. Wang, and T. Chen, "Audio Feature Extraction and Analysis for Scene Classification," *Journal of VLSI Signal Processing*, pp. 343–348, 1998. DOI: 10.1109/MMSP.1997.602659.

[289] H. Soltau, T. Schulz, M. Westphal, and A. Waibel, "Recognition of Music Types," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Seattle: Institute of Electrical and Electronics Engineers (IEEE), May 1998. DOI: 10.1109/ICASSP.1998.675470.

[290] R. B. Dannenberg, B. Thom, and D. Watson, "A Machine Learning Approach to Musical Style Recognition," in *Proceedings of the International Computer Music Conference (ICMC)*, Thessaloniki: International Computer Music Association (ICMA), Sep. 1997.

[291] T. Zhang, "Hierarchical System for Content-Based Audio Classification and Retrieval," *Proceedings of SPIE*, pp. 398–409, 1998, ISSN: 0277786X. DOI: 10.1117/12.325832.

[292] S. Z. Li, "Content-Based Audio Classification and Retrieval Using the Nearest Feature Line Method," *Transactions on Speech and Audio Processing*, vol. 8, no. 5, pp. 619–625, 2000, ISSN: 10636676. DOI: 10.1109/89.861383.

[293] S. Dixon, E. Pampalk, and G. Widmer, "Classification of Dance Music by Periodicity Patterns," *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, pp. 1–7, Oct. 2003.

[294] G. Tzanetakis, R. Jones, and K. McNally, "Stereo Panning Features for Classifying Recording Production Style," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Vienna, 2007, pp. 441–444.

[295] P. Ahrendt, A. Meng, and J. Larsen, "Decision Time Horizon for Music Genre Classification Using Short Time Features," in *Proceedings of the 12th European Signal Processing Conference (EUSIPCO)*, Vienna, Sep. 2004.

[296] R. O. Gjerdingen and D. Perrott, "Scanning the Dial: The Rapid Recognition of Music Genres," *Journal of New Music Research*, vol. 37, no. 2, pp. 93–100, Jun. 2008, ISSN: 0929-8215. DOI: 10.1080/09298210802479268.

[297] G. Tzanetakis and P. Cook, "MARSYAS: A Framework for Audio Analysis," *Organised Sound*, vol. 4, no. 3, 2000. DOI: 10.1017/S1355771800003071.

[298] G. Guo and S. Z. Li, "Content-Based Audio Classification and Retrieval by Support Vector Machines.," *Transactions on Neural Networks*, vol. 14, no. 1, pp. 209–215, Jan. 2003, ISSN: 1045-9227. DOI: 10.1109/TNN.2002.806626.

[299] A. Flexer, "A Closer Look on Artist Filters for Musical Genre Classification," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Vienna, 2007.

[300] M. A. Pitt and C. B. Monahan, "The Perceived Similarity of Auditory Polyrhythms," *Perception & Psychophysics*, vol. 41, no. 6, pp. 534–546, Jun. 1987, ISSN: 0031-5117. DOI: 10.3758/BF03210488.

[301] E. Bigand, "Abstraction of Two Forms of Underlying Structure in a Tonal Melody," *Psychology of Music*, vol. 18, no. 1, pp. 45–59, Apr. 1990, ISSN: 0305-7356. DOI: 10.1177/0305735690181004.

[302] C. L. Krumhansl, "Memory for Musical Surface," *Memory & Cognition*, vol. 19, no. 4, pp. 401–411, Jul. 1991, ISSN: 0090-502X. DOI: 10.3758/BF03197145.

[303] J. C. Bartlett, "Scale Structure and Similarity of Melodies," *Music Perception*, vol. 5, no. 3, pp. 285–314, 1988.

[304] T. Eerola, T. Järvinen, J. Louhivuori, and P. Toiviainen, "Statistical Features and Perceived Similarity of Folk Melodies," *Music Perception*, vol. 18, no. 3, pp. 275–296, Mar. 2001, ISSN: 0730-7829. DOI: 10.1525/mp.2001.18.3.275.

[305] A. Lamont and N. Dibben, "Motivic Structure and the Perception of Similarity," *Music Perception*, vol. 18, no. 3, pp. 245–274, Mar. 2001, ISSN: 0730-7829. DOI: 10.1525/mp.2001.18.3.245.

[306] R. Timmers, "Predicting the Similarity between Expressive Performances of Music from Measurements of Tempo and Dynamics," *Journal of the Acoustical Society of America (JASA)*, vol. 117, no. 1, 2005. DOI: 10.1121/1.1835504.

[307] S. McAdams and D. Matzkin, "Similarity, Invariance, and Musical Variation," *Annals of the New York Academy of Sciences*, vol. 930, no. 1, pp. 62–76, Jan. 2001, ISSN: 00778923. DOI: 10.1111/j.1749-6632.2001.tb05725.x.

[308] A. Novello, M. F. Mckinney, and A. Kohlrausch, "Perceptual Evaluation of Music Similarity," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Victoria, 2007.

[309] B. Logan and A. Salomon, "A Music Similarity Function Based on Signal Analysis," in *Proceedings of the International Conference on Multimedia and Expo (ICME)*, Institute of Electrical and Electronics Engineers (IEEE), 2001, pp. 745–748, ISBN: 0-7695-1198-8. DOI: 10.1109/ICME.2001.1237829.

[310] J. Foote and M. Cooper, "Visualizing Musical Structure and Rhythm via Self-Similarity," in *Proceedings of the International Computer Music Conference (ICMC)*, Habana, Sep. 2001.

[311] J.-J. Aucouturier and F. Pachet, "Music Similarity Measures: What's the Use?," in *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, Paris, 2002.

[312] E. Pampalk, A. Rauber, and D. Merkl, "Content-Based Organization and Visualization of Music Archives," in *Proceedings of the 10th ACM International Conference on Multimedia*, New York: ACM Press, 2002, p. 570. DOI: 10.1145/641007.641121.

[313] T. Li and M. Ogihara, "Content-Based Music Similarity Search and Emotion Detection," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 5, Institute of Electrical and Electronics Engineers (IEEE), 2004, pp. 705–708, ISBN: 0-7803-8484-9. DOI: 10.1109/ICASSP.2004.1327208.

[314] D. Bogdanov, J. Serrà, N. Wack, and P. Herrera, "From Low-Level to High-Level: Comparative Study of Music Similarity Measures," *Proceedings of the 11th International Symposium on Multimedia*, pp. 453–458, 2009. DOI: 10.1109/ISM.2009.72.

[315] F. Vignoli, "Digital Music Interaction Concepts: A User Study," in *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, Barcelona, 2004.

[316] P. R. Kleinginna and A. M. Kleinginna, "A Categorized List of Emotion Definitions, with Suggestions for a Consensual Definition," *Motivation and Emotion*, vol. 5, no. 4, pp. 345–379, Dec. 1981, ISSN: 0146-7239. DOI: 10.1007/BF00992553.

[317] H. P. Weld, "An Experimental Study of Musical Enjoyment," *The American Journal of Psychology*, vol. 23, no. 2, pp. 245–308, 1912. DOI: 10.2307/1412844.

[318] K. R. Scherer, "What Are Emotions? And How Can They Be Measured?," *Social Science Information*, vol. 44, no. 4, pp. 695–729, Dec. 2005, ISSN: 0539-0184. DOI: 10.1177/0539018405058216.

[319] L. B. Meyer, *Emotion and Meaning in Music*. Chicago: University of Chicago Press, 1956.

[320] K. R. Scherer, "Why Music Does Not Produce Basic Emotions: Pleading for a New Approach to Measuring the Emotional Effects of Music," in *Proceedings of the Stockholm Music Acoustics Conference (SMAC)*, Stockholm, Aug. 2003.

[321] ——, "Which Emotions Can Be Induced by Music? What Are the Underlying Mechanisms? And How Can We Measure Them?," *Journal of New Music Research*, vol. 33, no. 3, pp. 239–251, Sep. 2004, ISSN: 0929-8215. DOI: 10.1080/0929821042000317822.

[322] M. Zentner, D. Grandjean, and K. R. Scherer, "Emotions Evoked by the Sound of Music: Characterization, Classification, and Measurement," *Emotion*, vol. 8, no. 4, pp. 494–521, Aug. 2008, ISSN: 1528-3542. DOI: 10.1037/1528-3542.8.4.494.

[323] P. Juslin, "Cue Utilization of Emotion in Music Performance: Relating Performance to Perception," *Journal of Experimental Psychology*, vol. 26, no. 6, pp. 1797–1813, 2000. DOI: 10.1037/0096-1523.26.6.1797.

[324] R. Dillon, "Extracting Audio Cues in Real Time to Understand Musical Expressiveness," in *Proceedings of the MOSART Workshop*, Barcelona, Nov. 2001.

[325] ——, "A Statistical Approach to Expressive Intention Recognition in Violin Performances," in *Proceedings of the Stockholm Music Acoustics Conference (SMAC)*, Stockholm, Aug. 2003.

[326] J. A. Russel, "A Circumplex Model of Affect," *Journal of Personality and Social Psychology*, vol. 39, no. 6, pp. 1161–1178, 1980. DOI: 10.1037/h0077714.

[327] M. Leman, V. Vermeulen, L. De Voogdt, D. Moelants, and M. Lesaffre, "Prediction of Musical Affect Using a Combination of Acoustic Structural Cues," *Journal of New Music Research*, vol. 34, no. 1, pp. 39–67, Mar. 2005, ISSN: 0929-8215. DOI: 10.1080/09298210500123978.

[328] K. F. MacDorman, S. Ough, and H. Chin-Chang, "Automatic Emotion Prediction of Song Excerpts: Index Construction, Algorithm Design, and Empirical Comparison," *Journal of New Music Research*, vol. 36, no. 4, pp. 281–299, Dec. 2007, ISSN: 0929-8215. DOI: 10.1080/09298210801927846.

[329] E. Schubert, "Update of the Hevner Adjective Checklist," *Perceptual and Motor Skills*, vol. 96, pp. 1117–1122, 2003. DOI: 10.2466/pms.2003.96.3c.1117.

[330] X. Hu and J. S. Downie, "Exploring Mood Metadata: Relationships with Genre, Artist and Usage Metadata," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, Vienna, 2007.

[331] J. Kantor-Martynuska, "Emotion-Relevant Characteristics of Temperament and the Perceived Magnitude of Tempo and Loudness of Music," in *Proceedings of the 9th International Conference on Music Perception and Cognition (ICMPC)*, Bologna, Aug. 2006.

[332] J. Sloboda and A. Lehmann, "Tracking Performance Correlates of Changes in Perceived Intensity of Emotion During Different Interpretation of a Chopin Piano Prelude," *Music Perception*, vol. 19, no. 1, pp. 87–120, 2001. DOI: 10.1525/mp.2001.19.1.87.

[333] E. Schubert, "Modeling Perceived Emotion with Continuous Musical Features," *Music Perception*, vol. 21, no. 4, pp. 561–585, 2004. DOI: 10.1525/mp.2004.21.4.561.

[334] R. Timmers, M. Marolt, A. Camurri, and G. Volpe, "Listeners' Emotional Engagement with Performances of a Scriabin Etude: An Explorative Case Study," *Psychology of Music*, vol. 34, no. 4, pp. 481–510, 2006. DOI: 10.1177/0305735606067165.

[335] G. Husain, W. F. Thompson, and E. Schellenberg, "Effects of Musical Tempo and Mode on Arousal, Mood, and Spatial Abilities," *Music Perception*, vol. 20, no. 2, pp. 151–171, 2002. DOI: 10.1525/mp.2002.20.2.151.

[336] L. Mion and G. De Poli, "Score-Independent Audio Features for Description of Music Expression," *Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 2, pp. 458–466, Feb. 2008, ISSN: 1558-7916. DOI: 10.1109/TASL.2007.913743.

[337] Y. Feng, Y. Zhuang, and Y. Pan, "Music Information Retrieval by Detecting Mood via Computational Media Aesthetics," in *Proceedings of the International Conference on Web Intelligence (WI)*, Institute of Electrical and Electronics Engineers (IEEE), 2003, pp. 235–241, ISBN: 0-7695-1932-6. DOI: 10.1109/WI.2003.1241199.

[338] L. Lu, D. Liu, and H.-J. Zhang, "Automatic Mood Detection and Tracking of Music Audio Signals," *Transactions on Audio, Speech and Language Processing*, vol. 14, no. 1, pp. 5–18, Jan. 2006, ISSN: 1558-7916. DOI: 10.1109/TSA.2005.860344.

[339] Y.-H. Yang, Y.-f. Su, Y.-c. Lin, and H. H. Chen, "Music Emotion Recognition: The Role of Individuality," in *Proceedings of the International Workshop on Human-Centered Multimedia (HCM)*, Augsburg: ACM, 2007. DOI: 10.1145/1290128.1290132.

[340] Y.-H. Yang, C.-C. Liu, and H. H. Chen, "Music Emotion Classification: A Fuzzy Approach," in *Proceedings of the 14th Annual ACM International Conference on Multimedia*, ACM, 2006. DOI: 10.1145/1180639.1180665.

[341] Y.-H. Yang, Y.-C. Lin, Y.-F. Su, and H. H. Chen, "A Regression Approach to Music Emotion Recognition," *Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 2, pp. 448–457, Feb. 2008, ISSN: 1558-7916. DOI: 10.1109/TASL.2007.911513.

[342] A. Huq, J. P. Bello, and R. Rowe, "Automated Music Emotion Recognition: A Systematic Evaluation," *Journal of New Music Research*, vol. 39, no. 3, pp. 227–244, Sep. 2010, ISSN: 0929-8215. DOI: 10.1080/09298215.2010.513733.

[343] K. Trohidis, G. Tsoumakas, G. Kalliris, and I. Vlahavas, "Multi-Label Classification of Music into Emotions," in *Proceedings of the 9th International Conference of Music Information Retrieval (ISMIR)*, Philadelphia, 2008.

[344] B.-j. Han, S. Rho, S. Jun, and E. Hwang, "Music Emotion Classification and Context-Based Music Recommendation," *Multimedia Tools and Applications*, vol. 47, no. 3, pp. 433–460, Aug. 2010, ISSN: 1380-7501. DOI: 10.1007/s11042-009-0332-6.

[345] Y. E. Kim, E. M. Schmidt, R. Migneco, B. G. Morton, P. Richardson, J. Scott, J. A. Speck, and D. Turnbull, "Music Emotion Recognition: A State of the Art Review," in *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, Utrecht, 2010.

[346] M. J. Kartomi, *On Concepts and Classifications of Musical Instruments*. Chicago: University of Chicago Press, 1990, ISBN: 978-0-226-42549-8.

[347] P. Herrera, X. Amatriain, E. Batlle, and X. Serra, "Towards Instrument Segmentation for Music Content Description: A Critical Review of Instrument Classification Techniques," in *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, Plymouth, 2000.

[348] I. Kaminskyj and A Materka, "Automatic Source Identification of Monophonic Musical Instrument Sounds," in *Proceedings of the International Conference on Neural Networks (ICNN)*, vol. 1, Institute of Electrical and Electronics Engineers (IEEE), 1995, pp. 189–194, ISBN: 0-7803-2768-3. DOI: 10.1109/ICNN.1995.488091.

[349] I. Kaminskyj and P Voumard, "Enhanced Automatic Source Identification of Monophonic Musical Instrument Sounds," in *Proceedings of the Australian New Zealand Conference on Intelligent Information Systems (ANZIIS)*, Adelaide: Institute of Electrical and Electronics Engineers (IEEE), Nov. 1996, pp. 76–79, ISBN: 0-7803-3667-4. DOI: 10.1109/ANZIIS.1996.573893.

[350] J. C. Brown, "Computer Identification of Musical Instruments Using Pattern Recognition with Cepstral Coefficients as Features," *Journal of the Acoustical Society of America (JASA)*, vol. 105, no. 3, pp. 1933–41, Mar. 1999, ISSN: 0001-4966. DOI: 10.1121/1.426728.

[351] J. Marques and P. J. Moreno, "A Study of Musical Instrument Classification Using Gaussian Mixture Models and Support Vector Machines," Cambridge Research Laboratory CRL 99/4, Tech. Rep. June, 1999.

[352] A. Eronen and A. P. Klapuri, "Musical Instrument Recognition Using Cepstral Coefficients and Temporal Features," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, Institute of Electrical and Electronics Engineers (IEEE), 2000, pp. II753–II756, ISBN: 0-7803-6293-4. DOI: 10.1109/ICASSP.2000.859069.

[353] B. Kostek and A. Czyzewski, "Representing Musical Instrument Sounds for Their Automatic Classification," *Journal of the Audio Engineering Society (JAES)*, vol. 49, no. 9, pp. 768–785, 2001.

[354] A. Eronen, "Comparison of Features for Musical Instrument Recognition," in *Proceedings of the Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, Oct. 2001. DOI: 10.1109/ASPAA.2001.969532.

[355] J. C. Brown, O. Houix, and S. McAdams, "Feature Dependence in the Automatic Identification of Musical Woodwind Instruments," *Journal of the Acoustical Society of America (JASA)*, vol. 109, no. 3, pp. 1064–1072, 2001, ISSN: 00014966. DOI: 10.1121/1.1342075.

[356] G. Agostini, M. Longari, and E. Pollastri, "Musical Instrument Timbres Classification with Spectral Features," *Journal on Applied Signal Processing*, vol. 1, pp. 5–14, 2003. DOI: 10.1155/S1110865703210118.

[357] A. G. Krishna and T. V. Sreenivas, "Music Instrument Recognition: From Isolated Notes to Solo Phrases," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, Institute of Electrical and Electronics Engineers (IEEE), 2003, pp. iv–265iv–268–, ISBN: 0-7803-8484-9. DOI: 10.1109/ICASSP.2004.1326814.

[358] A. A. Livshin and X. Rodet, "Musical Instrument Identification in Continuous Recordings," in *Proceedings of the 7th International Conference on Digital Audio Effects (DAFX)*, Naples, 2004.

[359] S. Essid, G. Richard, and B. David, "Musical Instrument Recognition on Solo Performance," in *Proceedings of the European Signal Processing Conference (EUSIPCO)*, 2004.

[360] ——, "Musical Instrument Recognition by Pairwise Classification Strategies," *Transactions on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1401–1412, Jul. 2006, ISSN: 1558-7916. DOI: 10.1109/TSA.2005.860842.

[361] J. D. Deng, C. Simmermacher, and S. Cranefield, "A Study on Feature Analysis for Musical Instrument Classification.," *Transactions on Systems, Man, and Cybernetics*, vol. 38, no. 2, pp. 429–38, Apr. 2008, ISSN: 1083-4419. DOI: 10.1109/TSMCB.2007.913394.

[362] C. Joder, S. Essid, and G. Richard, "Temporal Integration for Audio Classification with Application to Musical Instrument Classification," *Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 1, pp. 174–186, Jan. 2009, ISSN: 1558-7916. DOI: 10.1109/TASL.2008.2007613.

[363] J. Eggink and G. Brown, "A Missing Feature Approach to Instrument Identification in Polyphonic Music," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 5, Institute of Electrical and Electronics Engineers (IEEE), 2003, pp. V–5536–, ISBN: 0-7803-7663-3. DOI: 10.1109/ICASSP.2003.1200029.

[364] J. Eggink and G. J. Brown, "Application of Missing Feature Theory to the Recognition of Musical Instruments in Polyphonic Audio," in *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, 2003.

[365] ——, "Instrument Recognition in Accompanied Sonatas and Concertos," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, Institute of Electrical and Electronics Engineers (IEEE), 2004, pp. iv–217iv–220–, ISBN: 0-7803-8484-9. DOI: 10.1109/ICASSP.2004.1326802.

[366] T. Heittola, A. P. Klapuri, and T. Virtanen, "Musical Instrument Recognition in Polyphonic Audio Using Source-Filter Model for Sound Separation," in *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR)*, Kobe, 2009.

[367] P. Cano, E. Batlle, E. Gomez, L. D. C. T. Gomes, and M. Bonnet, "Audio Fingerprinting: Concepts and Applications," in *Computational Intelligence for Modelling and Prediction*, S. K. Halgamuge and L. Wang, Eds., vol. 245, Berlin: Springer, 2005, ch. 17, pp. 233–245. DOI: 10.1007/10966518_17.

[368] P. Cano, E. Batlle, T. Kalker, and J. Haitsma, "A Review of Audio Fingerprinting," *Journal of VLSI Signal Processing*, vol. 41, no. 3, pp. 271–284, 2005. DOI: 10.1007/s11265-005-4151-3.

[369] J. G. Lourens, "Detection and Logging Advertisements Using Its Sound," *Transactions on Broadcasting*, vol. 36, no. 3, pp. 231–233, 1990, ISSN: 00189316. DOI: 10.1109/11.59850.

[370] P. Cano, E. Batlle, H. Mayer, and H. Neuschmied, "Robust Sound Modeling for Song Detection in Broadcast Audio," in *Proceedings of the 112th Audio Engineering Society Convention, Preprint 5531*, Munich: Audio Engineering Society (AES), 2002.

[371] A. Ramalingam and S. Krishnan, "Gaussian Mixture Modeling of Short-Time Fourier Transform Features for Audio Fingerprinting," *Transactions on Information Forensics and Security*, vol. 1, no. 4, pp. 457–463, Dec. 2006, ISSN: 1556-6013. DOI: 10.1109/TIFS.2006.885036.

[372] O. Hellmuth, E. Allamanche, J. Herre, T. Kastner, M. Cremer, and W. Hirsch, "Advanced Audio Identification Using MPEG-7 Content Description," in *Proceedings of the 111th Audio Engineering Society Convention, Preprint 5463*, New York: Audio Engineering Society, 2001.

[373] J. S. Seo, M. Jin, S. Lee, D. Jang, S. Lee, and C. D. Yoo, "Audio Fingerprinting Based on Normalized Spectral Subband Centroids," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Institute of Electrical and Electronics Engineers (IEEE), 2005, pp. 213–216, ISBN: 0-7803-8874-7. DOI: 10.1109/ICASSP.2005.1415684.

[374] A. Kimura, K. Kashino, T. Kurozumi, and H. Murase, "Very Quick Audio Searching: Introducing Global Pruning to the Time-Series Active Search," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 3, Institute of Electrical and Electronics Engineers (IEEE), 2001, pp. 1429–1432, ISBN: 0-7803-7041-4. DOI: 10.1109/ICASSP.2001.941198.

[375] J. Haitsma and T. Kalker, "Speed-Change Resistant Audio Fingerprinting Using Auto-Correlation," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, Institute of Electrical and Electronics Engineers (IEEE), 2003, pp. IV–72831–, ISBN: 0-7803-7663-3. DOI: 10.1109/ICASSP.2003.1202746.

[376] C. Yang, "MACS: Music Audio Characteristic Sequence Indexing for Similarity Retrieval," in *Proceedings of the Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz: Institute of Electrical and Electronics Engineers (IEEE), Oct. 2001, pp. 123–126, ISBN: 0-7803-7126-7. DOI: 10.1109/ASPAA.2001.969558.

[377] A. Wang, "An Industrial Strength Audio Search Algorithm," in *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, Washington, 2003.

[378] S. Kim and C. D. Yoo, "Boosted Binary Audio Fingerprint Based on Spectral Subband Moments," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 1, Institute of Electrical and Electronics Engineers (IEEE), Apr. 2007, pp. I–241I–244–, ISBN: 1-4244-0727-3. DOI: 10.1109/ICASSP.2007.366661.

[379] S. Sukittanon and L. E. Atlas, "Modulation Frequency Features for Audio Fingerprinting," in *Proceedings of the International Conference on Acoustics Speech and Signal Processing (ICASSP)*, Institute of Electrical and Electronics Engineers (IEEE), 2002, pp. II–II, ISBN: 0-7803-0946-4. DOI: 10.1109/ICASSP.2002.1006107.

[380] D. Schonberg and D. Kirovski, "Fingerprinting and Forensic Analysis of Multimedia," in *Proceedings of the 12th ACM Multimedia Conference*, New York, 2004. DOI: 10.1145/1027527.1027712.

[381] J. I. Martínez, J. Vitola, A. Sanabria, and C. Pedraza, "Fast Parallel Audio Fingerprinting Implementation in Reconfigurable Hardware and GPUs," in *Proceedings of the Southern Conference on Programmable Logic (SPL)*, Institute of Electrical and Electronics Engineers (IEEE), Apr. 2011, pp. 245–250, ISBN: 978-1-4244-8847-6. DOI: 10.1109/SPL.2011.5782656.

[382] C. J. C. Burges, J. C. Platt, and S. Jana, "Extracting Noise-Robust Features from Audio Data," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Institute of Electrical and Electronics Engineers (IEEE), 2002, ISBN: 0-7803-0946-4. DOI: 10.1109/ICASSP.2002.1005916.

[383] G. Richly, L. Varga, F. Kovacs, and G. Hosszu, "Short-Term Sound Stream Characterization for Reliable, Real-Time Occurrence Monitoring of Given Sound-Prints," in *Proceedings of the 10th Mediterranean Electrotechnical Conference. Information Technology and Electrotechnology for the Mediterranean Countries (CMeleCon)*, vol. 2, Institute of Electrical and Electronics Engineers (IEEE), 2000, pp. 526–528, ISBN: 0-7803-6290-X. DOI: 10.1109/MELCON.2000.879986.

[384] J. Haitsma and T. Kalker, "A Highly Robust Audio Fingerprinting System," in *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR)*, Paris, 2002.

[385] E. Batlle, J. Masip, and E. Guaus, "Automatic Song Identification in Noisy Broadcast Audio," in *Proceedings of the International Conference on Signal and Image Processing (SIP)*, Kauai, 2002.

[386] T. Kastner, E. Allamanche, J. Herre, and O. Hellmuth, "MPEG-7 Scalable Robust Audio Fingerprinting," *Proceedings of the 112th Audio Engineering Society Convention, Preprint 5511*, 2002.

[387] J. Sloboda, *The Musical Mind — The Cognitive Psychology of Music*, ser. Oxford Psychology Series 5. Oxford: Oxford University Press, 1985, Reprinted 2004.

[388] R. Kendall and E. Carterette, "The Communication of Musical Expression," *Music Perception*, vol. 8, no. 2, pp. 129–164, 1990.

[389] F. Dorian, *The History of Music in Performance — The Art of Musical Interpretation from the Renaissance to Our Day*. New York: W W Norton, 1942.

[390] C. Palmer, "Music Performance," *Annual Review of Psychology*, vol. 48, pp. 115–138, 1997. DOI: 10.1146/annurev.psych.48.1.115.

[391] J. Beran and G. Mazzola, "Analyzing Musical Structure and Performance — A Statistical Approach," *Statistical Science*, vol. 14, no. 1, pp. 47–79, 1999. DOI: 10.1214/ss/1009211806.

[392] E. F. Clarke, "Expression and Communication in Musical Performance," in *Music, Language, Speech and Brain*, J. Sundberg, N. L, and R Carlson, Eds., London: Macmillan Press, 1991.

[393] J. Sloboda, "Music Performance," in *The Psychology of Music*, D. Deutsch, Ed., New York: Academic Press, 1982.

[394] R. Timmers and H. Honing, "On Music Performance, Theories, Measurement and Diversity," *Cognitive Processing*, vol. 1-2, M. A. Belardinelli, Ed., 2002.

[395] P. Walls, "Historical Performance and the Modern Performer," in *Musical Performance — A Guide to Understanding*, J. Rink, Ed., Fifth printing 2006, Cambridge: Cambridge University Press, 2002.

[396] E. Clarke, "Understanding the Psychology of Performance," in *Musical Performance — A Guide to Understanding*, J. Rink, Ed., 5th printing 2006, Cambridge: Cambridge University Press, 2002.

[397] ——, "Listening to Performance," in *Musical Performance — A Guide to Understanding*, J. Rink, Ed., Fifth printing 2006, Cambridge: Cambridge University Press, 2002.

[398] P. N. Juslin, "Five Myths about Expressivity in Music Performance and What to Do About Them," in *Proceedings of the International Conference on Arts and Humanities*, Honululu, Hawaii, Jan. 2003.

[399] ——, "Studies of Music Performance: A Theoretical Analysis of Empirical Findings," in *Proceedings of the Stockholm Music Acoustics Conference (SMAC)*, Stockholm, Aug. 2003.

[400] N. Todd, "Vestibular Feedback in Musical Performance," *Music Perception*, vol. 10, no. 3, pp. 379–382, 1993.

[401] P. Johnson, "The Legacy of Recordings," in *Musical Performance — A Guide to Understanding*, J. Rink, Ed., Fifth printing 2006, Cambridge: Cambridge University Press, 2002.

[402] S. Weinzierl and C. Franke, ""Lotte, ein Schwindel!" — Geschichte und Praxis des Musikschnitts am Beispiel von Beethovens 9. Symphonie," in *Proceedings of the VDT International Audio Convention (22. Tonmeistertagung)*, Hannover, Nov. 2002.

[403] H.-J. Maempel, S. Weinzierl, and P. Kaminski, "Audiobearbeitung," in *Handbuch der Audiotechnik*, S. Weinzierl, Ed., Berlin: Springer, 2008, pp. 719–784.

[404] H.-J. Maempel, "Musikaufnahmen als Datenquellen der Interpretationsanalyse," in *Gemessene Interpretation — Computergestützte Aufführungsanalyse im Kreuzverhör der Disziplinen*, ser. Klang und Begriff, H. von Lösch and S. Weinzierl, Eds., Mainz: Schott, 2011, pp. 157–171, ISBN: 978-3-7957-0771-2.

[405] R. W. Lundin, *An Objective Psychology of Music*. New York: Ronald Press, 1953.

[406] E. Clarke, "Empirical Methods in the Study of Performance," in *Empirical Musicology*, E. Clarke and N. Cook, Eds., Oxford: Oxford University Press, 2004.

[407] W. Goebl, S. Dixon, G. Poli, A. Friberg, R. Bresin, and G. Widmer, "Sense in Expressive Music Performance: Data Acquisition, Computational Studies, and Models," in *Sound to Sense, Sense to Sound: A State-of-the-Art*, M. Leman and D. Cirotteau, Eds., Logos Berlin, Nov. 2005.

[408] S. Weinzierl and H.-J. Maempel, "Zur Erklärbarkeit der Qualitäten musikalischer Interpretationen durch akustische Signalmaße," in *Gemessene Interpretation — Computergestützte Aufführungsanalyse im Kreuzverhör der Disziplinen*, ser. Klang und Begriff, H. von Lösch and S. Weinzierl, Eds., Mainz: Schott, 2011, pp. 213–236, ISBN: 978-3-7957-0771-2.

[409] S. Dixon and W. Goebl, "Pinpointing the Beat: Tapping to Expressive Performances," in *Proceedings of the 7th International Conference on Music Perception and Cognition (ICMPC)*, Sydney, Jul. 2002.

[410] J. Hong, "Motor Action in Performance — Rostropovich and Richter's Repeated Renditions of Prokofiev's Cello Sonata Op.119," in *Proceedings of the Digital Music Research Network Conference (DMRN)*, London, 2006.

[411] D. Povel, "Temporal Structure of Performed Music. Some Preliminary Observations," *Acta Psychologica*, vol. 41, pp. 309–320, 1977. DOI: 10.1016/0001-6918(77)90024-5.

[412] B. H. Repp, "Patterns of Expressive Timing in Performances of a Beethoven Minuet by Nineteen Famous Pianists," *Journal of the Acoustical Society of America (JASA)*, vol. 88, no. 2, pp. 622–641, 1990. DOI: 10.1121/1.399766.

[413] ——, "Expressive Timing in a Debussy Prelude: A Comparison of Student and Expert Pianists," *Musicae Scientiae*, vol. 1, no. 2, pp. 257–268, 1997.

[414] ——, "A Microcosm of Musical Expression. I. Quantitative Analysis of Pianists' Timing in the Initial Measures of Chopin's Etude in E Major," *Journal of the Acoustical Society of America (JASA)*, vol. 104, no. 2, pp. 1085–1100, 1998. DOI: 10.1121/1.423325.

[415] ——, "A Microcosm of Musical Expression. II. Quantitative Analysis of Pianists' Dynamics in the Initial Measures of Chopin's Etude in E Major," *Journal of the Acoustical Society of America (JASA)*, vol. 105, no. 3, pp. 1972–1988, 1999. DOI: 10.1121/1.426743.

[416] ——, "A Microcosm of Musical Expression. III. Contributions of Timing and Dynamics to the Aesthetic Impression of Pianists' Performances of the Initial Measures of Chopin's Etude in E Major," *Journal of the Acoustical Society of America (JASA)*, vol. 106, no. 1, pp. 469–478, 1999. DOI: 10.1121/1.427078.

[417] G. Widmer, "A Machine Learning Analysis of Expressive Timing in Pianists' Performances of Schumann's "Träumerei"," in *Proceedings of the Stockholm Symposium on Generative Grammars for Music Performance*, Stockholm, May 1995.

[418] ——, "Modeling the Rational Basis of Musical Expression," *Computer Music Journal*, vol. 19, no. 2, pp. 76–96, 1995.

[419] ——, "Applications of Machine Learning to Music Research: Empirical Investigations into the Phenomenon of Musical Expression," in *Machine Learning, Data Mining, and Knowledge Discovery: Methods and Applications*, R. S. Michalski, I. Bratko, and M. Kubat, Eds., Chichester: Wiley, 1998, pp. 269–293.

[420] ——, "In Search of the Horowitz Factor: Interim Report on a Musical Discovery Project," in *Proceedings of the 5th International Conference on Discovery Science (DS)*, R. S. Michalski, I. Bratko, and M. Kubat, Eds., London: Springer, 1998, pp. 13–21. DOI: 10.1007/3-540-36182-0_4.

[421] P. Zanon and G. Widmer, "Learning to Recognize Famous Pianists with Machine Learning Techniques," in *Proceedings of the Stockholm Music Acoustics Conference (SMAC)*, Stockholm, Aug. 2003.

[422] G. Widmer and P. Zanon, "Automatic Recognition of Famous Artists by Machine," Österreichisches Forschungsinstitut für Artificial Intelligence (ÖFAI), Vienna, Tech. Rep. TR-2004-04, 2004, Last retrieved on Dec. 12, 2011. [Online]. Available: http://www.ofai.at/cgi-bin/tr-online?number+2004-04.

[423] S. Dixon, W. Goebl, and E. Cambouropoulos, "Perceptual Smoothness of Tempo in Expressively Performed Music," *Music Perception*, vol. 23, no. 3, pp. 195–214, 2006. DOI: 10.1525/mp.2006.23.3.195.

[424] A. Hartmann, "Untersuchungen über das metrische Verhalten in musikalischen Interpretationsvarianten," *Archiv für die gesamte Psychologie*, vol. 84, pp. 103–192, 1932.

[425] M. Dovey, "Analysis of Rachmaninoff's Piano Performances Using Inductive Logic Programming (Extended Abstract)," in *Proceedings of the 8th European Conference on Machine Learning (ECML)*, Heraclion, Sep. 1995, pp. 279–282.

[426] C. Palmer, "Mapping Musical Thought to Musical Performance," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 15, no. 2, pp. 331–346, 1989. DOI: 10.1037/0096-1523.15.2.331.

[427] P. Desain and H. Honing, "Does Expressive Timing in Music Performance Scale Proportionally with Tempo?," *Psychological Research*, vol. 56, pp. 285–292, 1994. DOI: 10.1007/BF00419658.

[428] B. Repp, "Pedal Timing and Tempo in Expressive Piano Performance: A Preliminary Investigation," *Psychology of Music*, vol. 24, no. 2, pp. 199–221, 1996. DOI: 10.1177/0305735696242011.

[429] B. H. Repp, "The Dynamics of Expressive Piano Performance: Schumann's "Träumerei" Revisited," *Journal of the Acoustical Society of America (JASA)*, vol. 100, no. 1, pp. 641–650, 1996. DOI: 10.1121/1.415889.

[430] B. Repp, "The Art of Inaccuracy: Why Pianists' Errors Are Difficult to Hear," *Music Perception*, vol. 14, no. 2, pp. 161–184, 1996.

[431] ——, "The Effect of Tempo on Pedal Timing in Piano Performance," *Psychological Research*, vol. 60, no. 3, pp. 164–172, 1997.

[432] B. H. Repp, "Acoustics, Perception, and Production of Legato Articulation on a Computer-Controlled Grand Piano," *Journal of the Acoustical Society of America (JASA)*, vol. 102, no. 3, pp. 1878–1890, 1997. DOI: 10.1121/1.420110.

[433] R. Bresin, "Virtual Virtuosity — Studies in Automatic Music Performance," Dissertation, Royal Institute of Technology (KTH), Stockholm, 2000.

[434] W. Goebl, "Melody Lead in Piano Performance: Expressive Device or Artifact?," *Journal of the Acoustical Society of America (JASA)*, vol. 110, no. 1, pp. 563–572, 2001. DOI: 10.1121/1.1376133.

[435] W. Windsor, R. Aarts, P. Desain, H. Heijink, and R. Timmers, "The Timing of Grace Notes in Skilled Musical Performance at Different Tempi: A Preliminary Case Study," *Psychology of Music*, vol. 29, pp. 149–169, 2001. DOI: 10.1177/0305735601292005.

[436] E. Stamatatos, "A Computational Model for Discriminating Music Performers," in *Proceedings of the MOSART Workshop*, Barcelona, Nov. 2001.

[437] G. Widmer, "Machine Discoveries: A Few Simple, Robust Local Expression Principles," *Journal of New Music Research*, vol. 31, no. 1, pp. 37–50, 2002. DOI: 10.1076/jnmr.31.1.37.8103.

[438] G. Widmer and A. Tobudic, "Playing Mozart by Analogy: Learning Multi-level Timing and Dynamics Strategies," *Journal of New Music Research*, vol. 32, no. 3, pp. 259–268, 2003. DOI: 10.1076/jnmr.32.3.259.16860.

[439] C. Wöllner, "Expressive Timing and Intensity Profiles in Mental Performances," in *Proceedings of the 8th International Conference on Music Perception & Cognition (ICMPC)*, Evanston, Aug. 2004.

[440] W. Windsor, P. Desain, A. Penel, and M. Borkent, "A Structurally Guided Method for the Decomposition of Expression in Music Performance," *Journal of the Acoustical Society of America (JASA)*, vol. 119, no. 2, pp. 1182–1193, 2006. DOI: 10.1121/1.2146091.

[441] G Faraone, S Johansson, and P. Polotti, "The Influence of the Practice of Basso Continuo on the Intonation of a Professional Singer in the Time of Monteverdi," in *Proceedings of the Stockholm Music Acoustics Conference (SMAC)*, Stockholm, Aug. 2003.

[442] T. Walker, "Instrumental Difference in Characteristics of Expressive Musical Performance," Dissertation, The Ohio State University, Columbus, 2004.

[443] R. Bowman Macleod, "Influences of Dynamic Level and Pitch Height on the Vibrato Rates and Widths of Violin and Viola Players," Dissertation, Florida State University, College of Music, Tallahassee, 2006.

[444] E. Ornoy, "An Empirical Study of Intonation in Performances of J.S. Bach's Sarabandes: Temperament, "Melodic Charge" and "Melodic Intonation"," *Orbis MusicÆ*, vol. 14, D. Halperin, Ed., pp. 37–76, 2007.

[445] E. Rapoport, "The Marvels of the Human Voice: Poem-Melody-Vocal Performance," *Orbis MusicÆ*, vol. 14, D. Halperin, Ed., pp. 7–36, 2007.

[446] M. Molina-Solana, J. Arcos, and E. Gomez, "Using Expressive Trends for Identifying Violin Recordings," in *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, Philadelphia, Sep. 2008.

[447] R. Ramirez, A. Perez, and S. Kersten, "Performer Identification in Celtic Violin Recordings," in *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, Philadelphia, Sep. 2008.

[448] J. Jerkert, "Measurements and Models of Musical Articulation," Master's Thesis, DTH, Department of Speech, Music and Hearing, Oct. 2003.

[449] ——, "Music Articulation in the Organ," in *Proceedings of Joint Baltic-Nordic Acoustics Meeting*, Mariehamn, Jun. 2004.

[450] M. Clynes and J. Walker, "Music As Time's Measure," *Music Perception*, vol. 4, no. 1, pp. 85–120, 1986.

[451] S. Dahl, "The Playing of an Accent — Preliminary Observations from Temporal and Kinematic Analysis of Percussionists," *Journal of New Music Research*, vol. 29, no. 3, pp. 225–233, 2000. DOI: 10.1076/jnmr.29.3.225.3090.

[452] T. Hoshishiba, S. Horiguchi, and I. Fujinaga, "Study of Expression and Individuality in Music Performance Using Normative Data Derived from MIDI Recordings of Piano Music," in *Proceedings of the International Conference on Music Perception and Cognition (ICMPC)*, Montreal, Aug. 1996.

[453] J. Langner, R. Kopiez, C. Stoffel, and M. Wilz, "Realtime Analysis of Dynamic Shaping," in *Proceedings of the 6th International Conference on Music Perception and Cognition (ICMPC)*, Keele, Aug. 2000.

[454] R. Dillon, "On the Recognition of Expressive Intention in Music Playing: A Computational Approach with Experiments and Applications," Dissertation, University of Genoa, Faculty of Engineering, Genoa, 2004.

[455] B. Repp, "Effects of Auditory Feedback Deprivation on Expressive Piano Performance," *Music Perception*, vol. 16, no. 4, pp. 409–438, 1999.

[456] J. Geringer, "Continuous Loudness Judgments of Dynamics in Recorded Music Excerpts," *Journal of Research in Music Education (JRME)*, vol. 43, no. 1, pp. 22–35, 1995. DOI: 10.2307/3345789.

[457] H. von Lösch and F. Brinkmann, "Das Tempo in Beethovens Appassionata von Fereric Lamond (1927) bis András Schiff (2006)," in *Gemessene Interpretation — Computergestützte Aufführungsanalyse im Kreuzverhör der Disziplinen*, ser. Klang und Begriff, H. von Lösch and S. Weinzierl, Eds., Mainz: Schott, 2011, pp. 83–100, ISBN: 978-3-7957-0771-2.

[458] C. Krumhansl, "A Perceptual Analysis of Mozart's Piano Sonata K. 282: Segmentation, Tension and Musical Ideas," *Music Perception*, vol. 13, no. 3, pp. 401–432, 1996.

[459] R. Timmers, R. Ashley, P. Desain, and H. Heijink, "The Influence of Musical Context on Tempo Rubato," *Journal of New Music Research*, vol. 29, no. 2, 2000. DOI: 10.1076/jnmr.29.2.131.3095.

[460] B. Repp, "Quantitative Effects of Global Tempo on Expressive Timing in Music Performance: Some Perceptual Evidence," *Music Perception*, vol. 13, no. 3, pp. 39–57, 1995.

[461] S. Dalla Bella and C. Palmer, "Tempo and Dynamics in Piano Performance: The Role of Movement Amplitude," in *Proceedings of the 8th International Conference on Music Perception & Cognition (ICMPC)*, Evanston, Aug. 2004.

[462] J. Jerkert, "Measurements and Models of Musical Articulation," in *Proceedings of the Stockholm Music Acoustics Conference (SMAC)*, Stockholm, Aug. 2003.

[463] P. Pfordresher and C. Palmer, "Effects of Delayed Auditory Feedback on Timing of Music Performance," *Psychological Research*, vol. 16, pp. 71–79, 2002. DOI: 10.1007/s004260100075.

[464] S. Finney and C. Palmer, "Auditory Feedback and Memory for Music Performance: Some Evidence for an Encoding Effect," *Memory & Cognition*, vol. 31, no. 1, pp. 51–64, 2003. DOI: 10.3758/BF03196082.

[465] P. Pfordresher, "Auditory Feedback in Music Performance: The Role of Melodic Structure and Musical Skill," *Journal of Experimental Psychology*, vol. 31, no. 6, pp. 1331–1345, 2005. DOI: 10.1037/0096-1523.31.6.1331.

[466] W. Goebl and C. Palmer, "Tactile Feedback and Timing Accuracy in Piano Performance," *Experimental Brain Research*, vol. 186, no. 3, pp. 471–479, 2008. DOI: 10.1007/s00221-007-1252-1.

[467] C. Drake and C. Palmer, "Skill Acquisition in Music Performance: Relations between Planning and Temporal Control," *Cognition*, vol. 74, pp. 1–32, 2000. DOI: 10.1016/S0010-0277(99)00061-X.

[468] C. Palmer, "Conceptual and Motor Learning in Music Performance," *Psychological Science*, vol. 11, pp. 63–68, 2000. DOI: 10.1111/1467-9280.00216.

[469] R. Meyer and C. Palmer, "Temporal and Motor Transfer in Music Performance," *Music Perception*, vol. 21, no. 1, pp. 81–104, 2003. DOI: 10.1525/mp.2003.21.1.81.

[470] C. Palmer, "Nature of Memory for Music Performance Skills," in *Music, Motor Control and the Brain*, E. Altenmüller, M. Wiesendanger, and J. Kesselring, Eds., Oxford: Oxford University Press, 2006, pp. 39–53.

[471] R. Timmers, "Context-Sensitive Evaluation of Expression," in *Proceedings of the Workshop on Current Research Directions in Computer Music*, Barcelona, Nov. 2001.

[472] S. Thompson, A. Williamson, and E. Valentine, "Time-Dependent Characteristics of Performance Evaluation," *Music Perception*, vol. 25, no. 1, pp. 13–29, 2007. DOI: 10.1525/mp.2007.25.1.13.

[473] R. Timmers, "Communication of (E)motion through Performance: Two Case Studies," *Orbis MusicÆ*, vol. 14, D. Halperin, Ed., pp. 116–140, 2007.

[474] ——, "Perception of Music Performance on Historical and Modern Commercial Recordings," *Journal of the Acoustical Society of America (JASA)*, vol. 122, no. 5, 2007. DOI: 10.1121/1.2783987.

[475] E. Lapidaki, "Stability of Tempo Perception in Music Listening," *Music Education Research*, vol. 2, no. 1, pp. 25–44, 2000. DOI: 10.1080/14613800050004413.

[476] B. Repp, "Detecting Deviations from Metronomic Timing in Music: Effects of Perceptual Structure on the Mental Timekeeper," *Perception & Psychophysics*, vol. 61, no. 3, pp. 529–548, 1999. DOI: 10.3758/BF03211971.

[477] B. Aarden, "How the Timing Between Notes Can Impact Musical Meaning," in *Proceedings of the 9th International Conference on Music Perception and Cognition (ICMPC)*, Bologna, Aug. 2006.

[478] W. G. Gardner, "Efficient Convolution without Input-Output Delay," *Journal of the Audio Engineering Society (JAES)*, vol. 43, no. 3, pp. 127–136, 1995.

[479] F. Harris, "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, 1978. DOI: 10.1109/PROC.1978.10837.

[480] J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," *Mathematics of Computation*, vol. 19, no. 90, p. 297, Apr. 1965, ISSN: 00255718. DOI: 10.2307/2003354.

[481] W. T. Cochran, J. W. Cooley, D. L. Favin, H. D. Helms, R. A. Kaenel, W. W. Lang, G. C. Maling, D. E. Nelson, C. M. Rader, and P. D. Welch, "What Is the Fast Fourier Transform?," *Transactions on Audio and Electroacoustics*, vol. 15, no. 2, pp. 45–55, Jun. 1967, ISSN: 0018-9278. DOI: 10.1109/TAU.1967.1161899.

[482] ISO/IEC JTC1/SC22 14882:2003, "Programming Languages — C++," ISO/IEC, Standard, 2003.

[483] G. Tzanetakis, "MARSYAS-0.2: A Case Study in Implementing Music Information Retrieval Systems," in *Intelligent Music Information Systems: Tools and Methodologies*, J. Shen, J. Shepherd, B. Cui, and L. Liu, Eds., IGI Global, 2008, pp. 31–49. DOI: 10.4018/978-1-59904-663-1.ch002.

[484] X. Amatriain, P. Arumí, and M. Ramírez, "CLAM, Yet Another Library for Audio and Music Processing?," in *Proceedings of the 17th ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, New York: ACM Press, Nov. 2002, p. 46, ISBN: 1-58113-626-9. DOI: 10.1145/985072.985097.

[485] X. Amatriain, M. de Boer, E. Robledo, and D. Garcia, "CLAM: An OO Framework for Developing Audio and Music Applications," in *Proceedings of the 17th ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, New York: ACM Press, Nov. 2002, p. 22, ISBN: 1-58113-626-9. DOI: 10.1145/985072.985084.

[486] X. Amatriain, "CLAM: A Framework for Audio and Music Application Development," *IEEE Software*, vol. 24, no. 1, pp. 82–85, Jan. 2007, ISSN: 0740-7459. DOI: 10.1109/MS.2007.8.

[487] C. Mckay, J. A. Burgoyne, and I. Fujinaga, "jMIR and ACE XML: Tools for Performing and Sharing Research in Automatic Music Classification," in *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries Workshop on Integrating Digital Library Content with Computational Tools and Services*, 2009.

[488] C. Mckay and I. Fujinaga, "jMIR: Tools for Automatic Music Classification," in *Proceedings of the International Computer Music Conference (ICMC)*, ICMA, 2009.

[489] M. Schedl, "The CoMIRVA Toolkit for Visualizing Music-Related Data," Johannes Kepler University Linz, Tech. Rep., 2006, Last retrieved on Jan. 31, 2012. [Online]. Available: http://www.cp.jku.at/people/schedl/Research/Development/CoMIRVA/webpage/CoMIRVA.html.

[490] C. Cannam, C. Landone, and M. Sandler, "Sonic Visualiser: An Open Source Application for Viewing, Analysing, and Annotating Music Audio Files," in *Proceedings of the International Conference on Multimedia (MM)*, New York: ACM Press, 2010, pp. 1467–1468, ISBN: 978-1-60558-933-6. DOI: 10.1145/1873951.1874248.

[491] O. Lartillot and P. Toiviainen, "A Matlab Toolbox for Musical Feature Extraction from Audio," in *Proceedings of the 10th International Conference on Digital Audio Effects (DAFX)*, Bordeaux, 2007.

[492] J. Bullock, "LIBXTRACT: A Lightweight Library for Audio Feature Extraction," in *Proceedings of the International Computer Music Conference (ICMC)*, Copenhagen, Aug. 2007, pp. 3–6.

[493] C. Sanden, C. R. Befus, and J. Z. Zhang, "CAMEL: A Lightweight Framework for Content-Based Audio and Music Analysis," in *Proceedings of the 5th Audio Mostly Conference on Interaction with Sound (AM)*, New York: ACM Press, 2010. DOI: 10.1145/1859799.1859821.

[494] B. Mathieu, S. Essid, T. Fillon, J. Prado, and G. Richard, "YAAFE, an Easy to Use and Efficient Audio Feature Extraction Software," in *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, Utrecht, 2010, pp. 441–446.

[495] G. Peeters, B. L. Giordano, P. Susini, N. Misdariis, and S. McAdams, "The Timbre Toolbox: Extracting Audio Descriptors from Musical Signals," *Journal of the Acoustical Society of America (JASA)*, vol. 130, no. 5, pp. 2902–2916, 2011. DOI: 10.1121/1.3642604.

[496] N. Collins, "SCMIR: A SuperCollider Music Information Retrieval Library," in *Proceedings of the International Computer Music Conference (ICMC)*, Huddersfield: ICMA, 2011.

[497] S. Pfeiffer and C. Parker, "bewdy, Maaate!," in *Presentation at the Australian Linux Conference*, Sydney, Jan. 2001.

[498] F. Eyben, M. Wöllmer, and B. Schuller, "OpenSMILE — The Munich Versatile and Fast Open-Source Audio Feature Extractor," in *Proceedings of the International Conference on Multimedia (MM)*, New York: ACM Press, 2010, pp. 1459–1462, ISBN: 978-1-60558-933-6. DOI: 10.1145/1873951.1874246.

[499] A. Lerch, G. Eisenberg, and K. Tanghe, "FEAPI: A Low Level Feature Extraction Plugin API," in *Proceedings of 8th International Conference on Digital Audio Effects (DAFX)*, Madrid, Sep. 2005.

[500] C. Cannam, C. Landone, M. Sandler, and J. Bello, "The Sonic Visualiser: A Visualisation Platform for Semantic Descriptors from Musical Signals," in *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, Victoria, Oct. 2006.

[501] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA Data Mining Software: An Update," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 10–18, Nov. 2009, ISSN: 19310145. DOI: 10.1145/1656274.1656278.

[502] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A Matlab-like Environment for Machine Learning," in *Proceedings of the NIPS Workshop on Parallel and Large-Scale Machine Learning*, Sierra Nevada, 2011.

[503] C.-c. Chang and C.-j. Lin, "LIBSVM: A Library for Support Vector Machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, 2011. DOI: 10.1145/1961189.1961199.

Index