

DIPLOMA THESIS

# Applications of a Constant-Q Transform for Time- and Pitch-Scale Modifications

---

Christian Schörkhuber

---

Graz, December 2011

Supervisor: Dr. Alois Sontacchi

External Supervisor: Dr. Anssi Klapuri

Assessor: Prof. Robert Höldrich

Institute of Electronic Music and Acoustics

University of Music and Performing Arts Graz, Austria

Graz University of Technology, Austria





## **Abstract**

Modifications of the time- and pitch-scale of polyphonic music are usually performed by manipulating the time-frequency representation of an audio signal. Most of the approaches proposed in the past are thereby based on the short-time Fourier transform (STFT) although its linear frequency bin spacing is known to be inadequate to some degree for analyzing and processing music signals. For this signal class the constant-Q transform (CQT) is superior to the STFT as it features a geometrical bin spacing and high Q-factors (typically equivalent to 12-96 bins per octave). In music processing applications, however, the CQT has been playing only a minor role due to its computational complexity and the lack of an inverse transform. Recently, solutions to these problems have been proposed, thus rendering the CQT a feasible alternative to the STFT for music processing applications. In this thesis time- and pitch-scaling algorithms based on the CQT representation of music signals are proposed that improve upon the quality achieved by simple STFT based approaches.



## Kurzfassung

Algorithmen zur Zeit- und Frequenzskalierung polyphoner Musiksignale beruhen überwiegend auf einer Manipulation der Zeit-Frequenz-Repräsentation des Eingangssignals. Dabei wird beim Großteil der Ansätze die in der Vergangenheit vorgestellt wurden auf die Fouriertransformation zurückgegriffen. Es ist allerdings bekannt, dass diese Transformation durch die lineare Verteilung der Frequenzbins keine optimale Darstellung für Musiksignale liefert. Die geometrische Verteilung der Frequenzbins bei gleichzeitig hohen Q-Faktoren (üblicherweise entsprechen diese einer Auflösung von 12-96 Bins pro Oktave) die durch die Constant-Q Transform (CQT) erreicht wird, liefert für diese Signalklasse eine wesentlich geeignetere Darstellung. Im Bereich der Musiksignalverarbeitung wurde die CQT in der Vergangenheit allerdings aufgrund des größeren Rechenaufwands und vorallem aufgrund der fehlenden Rücktransformation nur wenig Beachtung geschenkt. Für beide Probleme wurden kürzlich Lösungen vorgeschlagen, wodurch die CQT mittlerweile eine brauchbare Alternative zur Kurzzeit-Fouriertransformation (STFT) darstellt. In dieser Diplomarbeit werden CQT basierte Algorithmen zur Zeit- und Frequenzskalierung von polyphonen Musiksignalen präsentiert, die bezüglich der Signalqualität einfache STFT basierte Ansätze übertreffen.



## Acknowledgements

This work is dedicated to my parents, who supported me and who encouraged me to study what I was interested in.

I want to thank my advisor, Dr. Alois Sontacchi, for his support during this work and my entire studies. Special thanks to Dr. Anssi Klapuri for inducting me into this field of research and for his enduring support.

Christian Schörkhuber  
Graz, Austria, December 2011





# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Constant-Q Transform . . . . .	2
1.1.1	A Note on the Inverse CQT . . . . .	4
1.2	Scope of Work . . . . .	5
1.2.1	Time-Scale Modifications . . . . .	5
1.2.2	Pitch Transpositions . . . . .	6
1.3	Thesis Layout . . . . .	7
<b>2</b>	<b>The Constant-Q Transform</b>	<b>9</b>
2.1	Time vs. Frequency . . . . .	9
2.2	Fourier Transform: A Gabor Approach . . . . .	10
2.3	A Musical Ear vs. the Fourier Transform . . . . .	12
2.4	The Notion of a Constant-Q Transform . . . . .	15
2.5	A Faster Algorithm . . . . .	20
2.5.1	Algorithm by Brown and Puckette . . . . .	20
2.5.2	Processing One Octave at a Time . . . . .	22
2.6	Computational Complexity . . . . .	27
2.7	Reconstruction from the CQT Coefficients . . . . .	28
2.8	Kernel Design . . . . .	31
2.9	Redundancy . . . . .	34
2.10	Data Structure . . . . .	36
2.11	Quality of Reconstruction . . . . .	38
<b>3</b>	<b>Time-Scale Modification of Audio Signals</b>	<b>43</b>

3.1	The Phase Vocoder . . . . .	46
3.1.1	Phase Coherence . . . . .	48
3.1.2	Phase Locking . . . . .	49
3.1.3	Shape Invariance . . . . .	51
3.1.4	Transient Processing . . . . .	52
3.1.5	Resolution Issues . . . . .	53
3.2	A CQT Phase Vocoder . . . . .	54
3.2.1	Applying the Concepts . . . . .	54
3.2.2	CQT vs. DFT Phase Vocoder . . . . .	71
3.3	Conclusion . . . . .	76
<b>4</b>	<b>Pitch Transposition in the CQT Domain</b>	<b>79</b>
4.1	Transposing the Entire Representation . . . . .	80
4.1.1	Implementation . . . . .	81
4.1.2	Audio Examples . . . . .	84
4.1.3	Transients . . . . .	85
4.1.4	Sources of Artifacts . . . . .	88
4.2	A Sinusoidal Modeling Approach . . . . .	90
4.3	Formant Preservation . . . . .	92
4.4	Note Selective Transposition . . . . .	94
4.5	Conclusion . . . . .	98
<b>5</b>	<b>Summary and Future Work</b>	<b>101</b>
5.1	Contributions . . . . .	102
5.2	Future Work . . . . .	104
	<b>Bibliography</b>	<b>107</b>

# Chapter 1

## Introduction

Time-frequency representations are ubiquitous tools in today's digital audio signal processing applications. However, the accuracy with which the time and frequency evolution of a signal can be determined simultaneously is limited as a frequency can never be precisely localized in time - a fact that is closely related to Heisenberg's uncertainty principle [Mal09]. Due to this inevitable trade-off between time- and frequency-resolution, one and the same time-domain signal can be specified by an infinite number of different time-frequency representations. This inherent ambiguity has led to a great diversity of definitions of time-frequency transformations [HA08] proposed in the last decades. Amongst these numerous transformations, arguably the short-time Fourier transform (STFT) is the most prominent one being *the* standard tool when it comes to time-frequency processing of digital (audio) signals. The main reasons for the supremacy of the STFT are probably its very efficient implementation, its perfect reconstruction property and the ease of interpreting the produced data.

In the field of music processing, however, the shortcomings of the STFT representation are well known. The frequency resolution of the STFT is linear, that is the frequency difference between the sampling points in frequency (bins) is constant. Contrary, the fundamental frequencies (F0s) of

the tones in Western music are geometrically spaced: in the standard 12-tone equal temperament, for example, the F0s obey  $F_k = 440\text{Hz} \times 2^{k/12}$ , where  $k \in [-50, 40]$  is an integer. Thus, in order to resolve single tones at lower frequencies, a very high frequency resolution is desired. In higher frequency regions on the other hand, rapid temporal changes call for a high temporal resolution. Hence, with its linear frequency bin spacing stemming from the fixed window size, the STFT cannot fully meet the needs when it comes to analyzing musical signals.

## 1.1 Constant-Q Transform

The Q-factor of a band-pass filter is defined as the ratio of the center frequency to its bandwidth. Considering the STFT's frequency bins as being band-pass filters, their Q-factors increase from low to high frequencies as their absolute bandwidths are constant. From an auditory perspective, this is in contrast with the frequency resolution of the peripheral hearing system of humans which is approximately constant-Q over a wide range from 20kHz down to approximately 500Hz, below which the Q-factors get progressively smaller [Moo95]. Hence, a constant-Q time-frequency representation providing a high frequency resolution for low frequencies and a high temporal resolution for high frequencies is well motivated from both musical and perceptual viewpoints.

The first formal definition of a constant-Q transform (CQT) for audio signal processing was proposed by Judith Brown [Bro91] in 1991. An efficient algorithm for the calculation of the CQT has been proposed in 1992 [BP92]. Here the CQT is a generalization of the STFT concerning the window lengths: in order to keep the Q-factor constant, windows get progressively smaller as frequency indices increase. Hence, the CQT is essentially a wavelet transform, but here the term CQT is preferred since it underlines the fact that transforms with relatively high Q-factors, equivalent to 12-96 bins per octave are considered. This renders many of the conventional wavelet transform tech-

niques inadequate; for example methods based on iterated filterbanks would require filtering the input signal hundreds of times.

Although the logarithmic frequency resolution of the CQT is clearly advantageous for broadband music signals, it has not replaced the STFT in audio signal processing. There are at least three reasons for this: Firstly, the algorithm is still computationally intensive when broadband music signals are considered. Secondly, the CQT proposed in [BP92] lacks an inverse transform that would allow reconstruction of the original signal from its transform coefficients. Thirdly, the CQT produces a data structure that is more difficult to work with than the time-frequency matrix (spectrogram) obtained by the STFT. The last problem is due to the fact that in CQT, the time resolution varies for different frequency bins, in effect meaning that the ‘sampling’ of different frequency bins is not synchronized. In chapter 2 solutions to these three problems are proposed.

As already mentioned above, constant-Q transform can be viewed as a wavelet transform. The wavelet literature is well-matured (see e.g. [Mal09]) and constant-Q (wavelet) transforms have been proposed that lead to perfect reconstruction. However, most of the work has focused on critically-sampled dyadic wavelet transforms where the frequency resolution is only one bin per octave – this is clearly insufficient for music signal analysis. Recently, perfect reconstruction wavelet transforms have been proposed that have rational dilation factors, meaning that the center frequencies of the bins are spaced by  $p/q$ , where  $p$  and  $q$  are integers [BS09] [KV93] [Sel11]. However, these are based on iterated filter banks and are therefore less attractive computationally when high Q-factors, such as 12–96 bins per octave, are required. Another interesting direction of research has been the application of frequency warping on a time-domain signal in such a way that the DFT of the warped signal is related to the DFT of the original signal via a frequency warping function [HKS<sup>+</sup>00, BS06]. A problem with these is that the warping filters have infinite impulse responses which makes it hard to design an inverse transform.

Brown and Puckette proposed a computationally efficient technique for computing constant-Q transforms with high Q-factors based on the fast Fourier transform (FFT) and a frequency-domain kernel [Bro91] [BP92]. A drawback of this CQT implementation is that there is no inverse transform for it. Recently, FitzGerald has shown that a good quality approximate inverse transform can be obtained if the signal to be inverted has a sparse representation in the discrete Fourier transform domain [FCC06]. However, this is not true for music signals in general.

In chapter 2 specific solutions to the three problems of the CQT mentioned above are proposed. The solution to the problem of computational efficiency is based on the technique proposed by Brown and Puckette in [BP92] which is extended to improve further its computational efficiency. Secondly, we propose to structure the transform kernel in such a way that reasonable-quality inverse transform (approximately 55dB signal-to-noise ratio) is obtained using the conjugate transpose of the CQT transform kernel. The reconstruction is achieved introducing only a moderate amount of redundancy (by factor four or five) to the transform (here redundancy refers to the number of elements in the transform compared to the samples in the original time-domain signal). Thirdly, interface tools for the data structure that facilitate working with the signal in the transform domain are proposed. A reference implementation of the proposed methods is provided as a Matlab toolbox at <http://www.elec.qmul.ac.uk/people/anssik/cqt/>. In [SK10] the approach outlined in chapter 2 is given in a more compact form.

### 1.1.1 A Note on the Inverse CQT

In [SK10] we showed that the original input signal can be efficiently reconstructed from its CQT coefficients with reasonable quality using the same kernel for forward and inverse transform. Very recently another approach to invertible constant-Q transforms featuring high Q-factors has been proposed, yielding even perfect reconstruction [VHDG11]. The suggested transform is based on frame theory [KC07] and utilizes nonstationary Gabor frames

[BDJ<sup>+</sup>11] to achieve geometrically spaced frequency bins and the property of invertibility. Loosely speaking, a frame is an overcomplete set of basis functions (think kernel matrix  $\Theta$  consisting of time-frequency atoms  $a_{n,k}$ ) allowing for redundant signal representations. Signal reconstruction from the redundant representation is then obtained by applying a dual frame (think inverse kernel matrix  $\tilde{\Theta}$  consisting of dual atoms  $b_{n,k}$ ) such that  $x = \Theta\tilde{\Theta}x$  for an input signal  $x$ . A computational efficient implementation of the constant-Q transform is proposed in [VHDG11] where atoms  $a_{n,k}$  are designed in the frequency domain and subsequently applied to the Fourier transform of the input signal.

Although the CQT proposed in [VHDG11] is very much appreciated, it is not considered throughout this thesis since the level of artifacts in time- or pitch-scaled music signals is usually far beyond the signal-to-noise ratio achieved by the inverse CQT we proposed in [SK10].

## 1.2 Scope of Work

As outlined above, the logarithmic frequency resolution obtained by the CQT is more adequate to represent audio signals in the time-frequency domain than the linear frequency resolution of the standard STFT. With the tools at hand proposed in [SK10], providing both a computational efficient implementation of the CQT and a reasonable quality inverse transform, it is natural to exploit this beneficial time-frequency representation for existing music processing applications that suffer from STFT resolution issues. In a first effort the scope of this thesis is to investigate into the applicability of the CQT to time- and pitch-scale modifications of polyphonic music signals.

### 1.2.1 Time-Scale Modifications

The aim of altering the time-scale of an audio signal is to change the overall duration of the signal while retaining the perceived pitch and timbre of the

material. This can be approached either in time-domain or in frequency-domain. While techniques that operate in time-domain are often preferred when monophonic music signals or speech signals are considered, frequency domain or hybrid approaches usually yield better results when time-scale modifications of polyphonic music signals are desired. Amongst the latter phase vocoder based techniques ([AKZ<sup>+</sup>02], [DGBA00]) are the most prominent ones. Literature on time-scale modifications based on the phase vocoder is well matured and algorithms able to produce high quality output signals have been proposed [Puc95] [LD97] [LD99a] [Röb03] [Roe10]. However, these algorithms are all based on STFT representations of audio signals and artifacts stemming from the inadequate frequency resolution are still an issue. Hence the objective is to improve the quality of the time-scaled output signal by replacing the STFT in a standard phase vocoder implementation with the CQT.

### 1.2.2 Pitch Transpositions

Pitch-scale modifications of audio signals aim at keeping the duration the input signal constant while altering its frequency content. The term *pitch transposition* refers to pitch-scale modifications expressed in musical terms. That is, rather than implementing arbitrary scaling factors, the desired frequency shift is given in semitones (or cents). Since time- and pitch-scale modifications are dual operations, shifting the frequency content of a signal is often implemented by applying a time-scaling stage followed by a resampling stage (or vice versa). On the other hand, approaches to manipulate the frequency content directly in the STFT representation have been proposed [LD99b] [JSBA08]. Although these approaches are able to achieve reasonable results they suffer from the inadequate frequency resolution featured by the STFT. The objective of this thesis is to exploit the advantageous frequency bin spacing provided by the CQT for pitch-scale modifications in the frequency domain.



## 1.3 Thesis Layout

In chapter 2 the constant-Q transform is derived as a generalization of the windowed Fourier transform. The algorithm to compute the CQT coefficients as proposed in [Bro91] and [BP92] is outlined and an improved algorithm that reduces further the computational complexity is proposed. The novel approach to reconstruct the original input signal from its CQT coefficients we proposed in [SK10] is described in detail and the achieved results are presented.

Chapter 3 provides a short overview of existing time-scale modification techniques whereas the phase vocoder is discussed in more detail. A phase vocoder based on the CQT representation is derived and the quality of the produced time-scaled output signals is compared to the standard STFT based approach.

In chapter 4 the implementation of a frequency-domain pitch transposition algorithm that operates on the CQT representation of the input signal is outlined. The ease of manipulating the pitch of single notes within a polyphonic music signal in the CQT domain (note selective transpositions) rather than transposing the entire input signal is demonstrated.

Chapter 5 provides a summary of the main contributions of the thesis and suggests a number of possible routes for future work.



# Chapter 2

## The Constant-Q Transform

### 2.1 Time vs. Frequency

When it comes to analyzing time-invariant signals in the frequency domain, usually the Fourier transform is the tool of choice. As soon as signals change over time, however, it is desired to localize information not only in frequency but also in time so as to transform a one-dimensional time-domain signal into a two-dimensional time-frequency representation. This can be achieved by multiplying the basis functions of the Fourier transform with a function that tapers off to zero at its ends (window function), thus localizing the frequency information in time (windowed Fourier transform or short-time Fourier transform). Following the definition of Gabor in his 'Theory Of Communication' [Gab46] published in 1946, this new set of basis functions that are now localized both in frequency and time, are called time-frequency atoms. Unfortunately, due to the uncertainty principle, the energy spread of a function and its Fourier transform cannot be arbitrarily small at the same time [Mal09]. The basis functions of the Fourier transform, that is, which are infinitely concentrated in frequency (but infinitely spread in time), get spread in frequency while localizing them in time. This matter is best visualized by considering the basis functions (atoms) as rectangles (or ellipses) in the time-

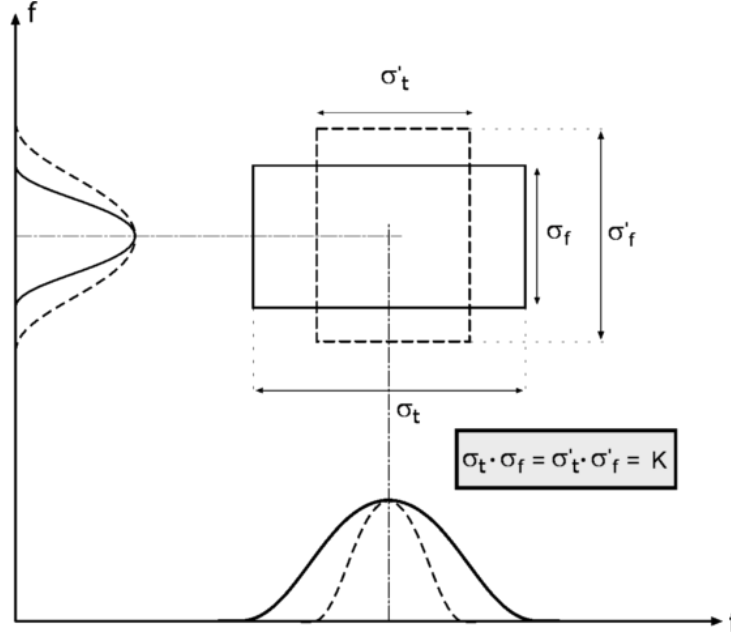


Figure 2.1: Area in the time-frequency plane that captured by a single time-frequency atom. The Area  $K$  is constant: As the atom is shrunk in time (shorter window length), it simultaneously gets dilated in frequency

frequency plane. Figure 2.1 illustrates, that the area of such a rectangles is fixed. As the atoms are dilated in time they are shrunk in frequency and vice versa. The exact value for the area  $K$  in figure 2.1 depends on the window function and reaches its minimum for a Gaussian function (since it is invariant under the Fourier transform), however, also other functions are used<sup>1</sup>.

## 2.2 Fourier Transform: A Gabor Approach

There are several ways to look at the short-time Fourier transform (STFT) or the windowed discrete Fourier transform (wDFT), respectively. Probably the most common viewpoint in engineering is to think of cutting a time-domain

<sup>1</sup>The Gaussian function is optimal in this theoretical sense. However, for several practical reasons (e.g. reduced frequency smearing), Hann, Hamming or Blackman windows are most frequently used.

signal into (overlapping) blocks, multiplying them with a window function to reduce leakage [OSB<sup>+</sup>89] and Fourier analyzing them. However, if the window function is thought to be applied to the basis functions rather than the signal, and the signal is thought of as a time-frequency plane rather than just a time-domain signal, we end up with the Gabor approach. Following this interpretation the windowed discrete Fourier transform  $X^{\text{DFT}}(k, n)$  of a discrete time-domain signal  $x(n)$  is defined by

$$X^{\text{DFT}}(k, n) = \sum_{m=n-\lfloor N/2 \rfloor}^{n+\lfloor \frac{N}{2} \rfloor} x(m) a^*(m - n + N/2) \quad (2.1)$$

where  $k = 0, 1, \dots, N - 1$  indexes the complex-valued transform coefficients (frequency bins) of the DFT,  $\lfloor \cdot \rfloor$  denotes rounding towards negative infinity and  $a^*(n)$  denotes the complex conjugate of  $a(n)$ . The basis functions  $a(n)$  are complex-valued waveforms, the time-frequency *atoms*, and are defined by

$$a(n) = w\left(\frac{n}{N}\right) \exp\left[i2\pi n \frac{f_k}{f_s}\right] \quad (2.2)$$

where  $f_k$  is the center frequency of bin  $k$ ,  $f_s$  denotes the sampling rate, and  $w(t)$ , is a continuous window function (for example Hann or Blackman window), sampled at points determined by  $t$ .  $N$  is the window length in samples<sup>2</sup>, the window function is zero outside the range  $t \in (0, 1)$ . Note that in (2.1) the windows are centered at sample  $n$  of the input signal.

The bins center frequencies  $f_k$  obey

---

<sup>2</sup>For simplicity it is assumed that the number of DFT coefficients (the *DFT size*) and the window length (the *window size*) is the same, i.e. the signal blocks are not zero-padded.

$$f_k = k \frac{f_s}{N} \quad (2.3)$$

and are thus linearly spaced along the frequency axis. In most applications  $X^{\text{DFT}}(k, n)$  is not computed for every  $n$  (unless a sliding DFT as described in [JL03] is applied), but for every  $H^{\text{th}}$  sample, with  $H$  being the *hop size*. Figure 2.2 illustrates how a signal is captured in the time-frequency plane by the wDFT. Each tile in figure 2.2 represents one time-frequency atom  $a(n)$  centered at  $f_k$  in frequency and  $n$  in time. Note that time-frequency atoms actually are not box-shaped: Figure 2.3(a)-(d) shows an example atom using a Hann window function in the time-domain, the frequency domain and the time-frequency plane, respectively. The atom is centered in time at sample  $n_c$  and in frequency at  $\omega_k = 2\pi f_k$ . In order to capture the signal uniformly over the entire time-frequency plane, the atoms need to overlap both in time and frequency such that the power sum is equal to unity in both domains (see section 2.7).

## 2.3 A Musical Ear vs. the Fourier Transform

The (w)DFT is *the* standard tool when digital signals are to be analyzed or processed in the frequency domain. And indeed it has quite a few properties that render the DFT a really neat transform: invertibility, fast computation via the FFT [BM67] and a data structure that is easily interpretable. When the class of signals to be analyzed is music, however, the DFT lacks the ability to capture all important features. The fundamental frequencies (F0s) of the tones in Western music are geometrically spaced: in the standard 12-tone equal temperament, for example, the F0s obey  $F_k = 440\text{Hz} \times 2^{k/12}$ , where  $k \in [-50, 40]$  is an integer. Thus, in order to resolve single tones in the spectrum in lower frequencies, a very high frequency resolution is desired. For higher frequencies on the other hand, the frequency difference between

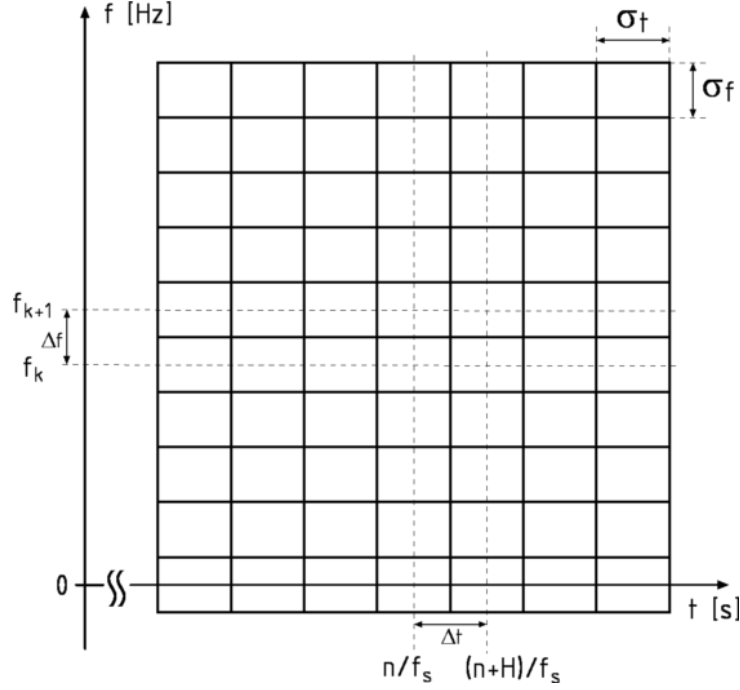


Figure 2.2: Spacing of time-frequency atoms for the wDFT:  $\Delta f = f_s/N$  and  $\Delta t = H/f_s$ .

tones is much higher but rapid time changes call for a high temporal resolution. As depicted in figure 2.2, the atoms are uniformly distributed across the time-frequency plane, hence, the absolute time/frequency resolution is constant over the entire frequency range. The parameter for this trade-off between time and frequency resolution is the window length  $N$  in (2.1) and (2.2), respectively. Since  $\sigma_f \propto 1/N$  and  $\sigma_t \propto N$  in figure 2.2, a large  $N$  accounts for a high frequency resolution but a low time resolution and vice versa. The rigidity of the DFT concerning this trade-off is also apparent when looking at the frequency responses of the atoms  $a(n)$  in figure 2.4 for a 8-point DFT (Gaussian window). Each atom forms a bandpass filter with a 3-dB bandwidth  $\delta f$  and a center frequency  $f_k$  (the *bin*-frequency). The Q-factor of a bandpass-filter is defined by

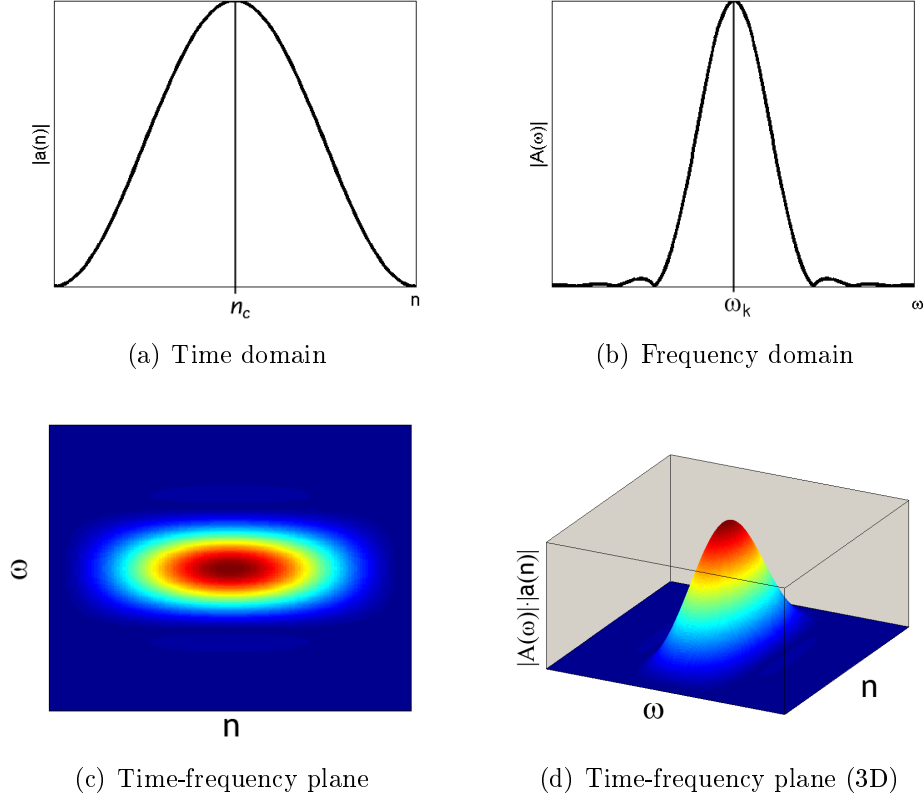


Figure 2.3: Exemplary time-frequency atom using a Hann window. (a) shows the atom  $a(n)$  in time domain as defined in (2.2), (b) shows the atom in the frequency domain whereas  $A(\omega) = \mathcal{F}\{a(n)\}$ , (c) shows the region in the time-frequency plane captured by the atom, and (d) shows a 3-dimensional model of the atom in the time-frequency plane. The z-axis in (d) corresponds to the amplitude product  $|a(n)| \cdot |A(\omega)|$ .

$$Q \stackrel{\text{def.}}{=} \frac{f_k}{\delta f} \quad (2.4)$$

that is, the Q-factor is the ratio of the center frequency of a filter to its bandwidth. As illustrated in figure 2.4 the Q-factor increases from low to high frequency bins since the absolute bandwidth  $\delta f$  is equal for all filters. This is in contrast to the peripheral hearing system of humans where the



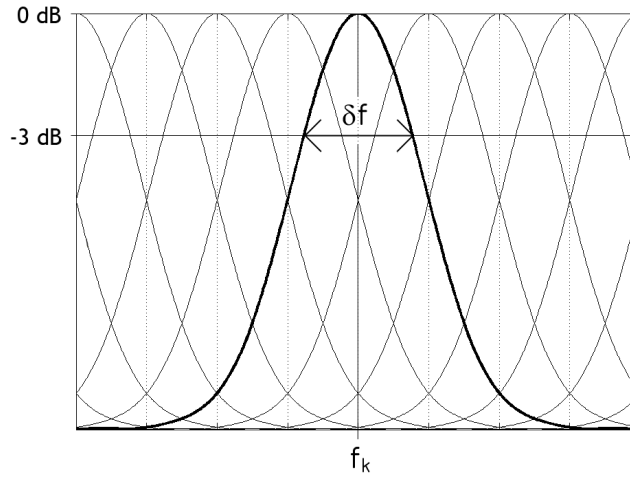


Figure 2.4: Frequency responses of the atoms (bandpass-filters) of a 8-point DFT using a Gaussian window. Each filter represents one DFT bin.

frequency resolution exhibits approximately a constant Q-factor over a wide range from 20kHz down to about 500Hz. Below this, the Q-values get progressively smaller as the bandwidths of the patterns of vibration on the basilar membrane in response to sinusoidal stimulation are approximately constant [Moo03]. So from both musical and perceptual viewpoints it seems obvious, that, as far as music signals are considered, the wDFT lacks the ability to adequately represent the signal over the entire frequency range.

## 2.4 The Notion of a Constant-Q Transform

There are two requirements that a time-frequency transform has to fulfill when the class of signals to be analyzed is music: the frequency bins have to be geometrically spaced (i.e., linear on a logarithmic frequency scale) and the bandwidths of the filters have to change with their center frequencies (i.e., the Q-factors of the filters should be constant rather than their bandwidths). Over the past few decades there have been several approaches to bypass the linear frequency resolution of the Fourier transform ([Har76], [YB78], [OJS71], [BO74], [Gam71], [Gam79], [KMR85]), however, in 1990 Judith

Brown was the first to propose a constant Q spectral transform [Bro91] that was specifically suitable for the analysis of musical signals. This constant-Q transform (CQT) is a generalization of the DFT: each time-frequency atom does not only have a distinct frequency but also a distinct length. Applying this notion to (2.1)-(2.2), the CQT transform  $X^{\text{CQ}}(k, n)$  of a discrete time-domain signal  $x(n)$  is defined by

$$X^{\text{CQ}}(k, n) = \sum_{m=n-\lfloor N_k/2 \rfloor}^{n+\lfloor N_k/2 \rfloor} x(m) a_k^*(m - n + N_k/2) \quad (2.5)$$

with its basis functions  $a_k(n)$  (time-frequency atoms, or in the sequel *kernel*) defined by

$$a_k(n) = \frac{1}{\sum_n w(n/N_k)} w\left(\frac{n}{N_k}\right) \exp\left[i2\pi n \frac{f_k}{f_s}\right], \quad (2.6)$$

whereas the window lengths  $N_k \in \mathbb{R}$  in (2.5)-(2.6) are real-valued and inversely proportional to  $f_k$  in order to have the same Q-factor for all bins  $k$ .

Since a bin spacing corresponding to the equal temperament is desired, the center frequencies  $f_k$  obey

$$f_k = f_1 2^{\frac{k-1}{B}} \quad (2.7)$$

where  $f_1$  is the center frequency of the lowest-frequency bin (due to the desired logarithmic frequency resolution there is no DC-bin), and  $B$  determines the number of bins per octave. For  $B = 12$  each CQT bin corresponds to one semitone, however, higher values may be chosen. In practice,  $B$  is the most important parameter of choice when using the CQT, because it determines the time-frequency resolution trade-off.

In (2.4) the Q-factor of a filter was defined as the ratio of the center frequency to its 3-dB bandwidth. Figure 2.5 shows an example of the frequency responses of the atoms for  $B = 12$  and one octave only. The Q-factor of bin

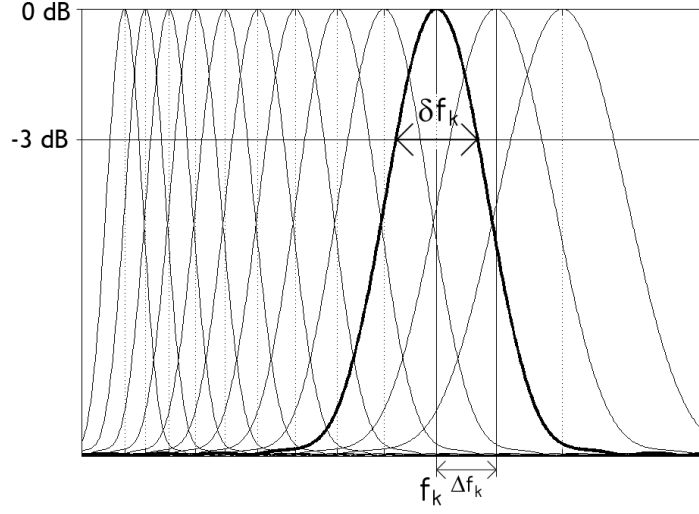


Figure 2.5: Frequency responses of the atoms (bandpass-filters) of a CQT using a Gaussian window with  $B = 12$  and for one octave only. Each filter represents one CQT bin.

$k$  in figure 2.5 is thus defined by

$$Q_k \stackrel{\text{def.}}{=} \frac{f_k}{\delta f_k} \quad (2.8)$$

where  $\delta f_k$  denotes the 3-dB bandwidth of the frequency response of the atom  $a_k(n)$  and  $f_k$  its center frequency. The bandwidth is given by

$$\delta f_k = \delta\omega \frac{f_s}{N_k} = \delta\omega \Delta f_k \quad (2.9)$$

where  $\Delta f_k$  is the frequency distance between to adjacent bins (the *bin spacing*) and  $\delta\omega$  is the 3-dB bandwidth of the main lobe of the used window function in DFT bins. In [Har78]  $\delta\omega$  is tabulated for several window functions, being  $\delta\omega \approx 1.30$  for Hamming and  $\delta\omega \approx 1.68$  for Blackman windows, for example. The bin spacing  $\Delta f_k$  (see figure 2.5) for the CQT is defined by

$$\Delta f_k \stackrel{!}{=} f_{k+1} - f_k = f_k \left( 2^{\frac{1}{B}} - 1 \right), \quad (2.10)$$

hence, after substituting (2.10) in (2.9) and (2.8), the Q-factors are given by

$$Q = \frac{q}{\delta\omega(2^{\frac{1}{B}} - 1)} \quad (2.11)$$

where the subscript of  $Q_k$  has been omitted since the Q-factors are by definition all equal and  $0 < q \lesssim 1$  is an additional scaling factor, and typically  $q = 1$ . Values of  $q$  smaller than 1 can be used to improve the time resolution at the cost of degrading the frequency resolution. Important to note is that setting for example  $q = 0.5$  and  $B = 48$  leads to exactly the same time-frequency resolution trade-off as setting  $q = 1$  and  $B = 24$ , but the former contains twice more frequency bins per octave. In this sense, values  $q < 1$  can be seen to implement *oversampling* of the frequency axis, analogously to the use of zero padding when calculating the DFT. For example  $q = 0.5$  corresponds to oversampling factor of 2: the effective frequency resolution is equivalent to  $B/2$  bins per octave, although  $B$  bins per octave are computed.

After substituting (2.9) in (2.8) and (2.11) and solving for  $N_k$ , the window size of the atom  $a_k(n)$  is given by

$$N_k = \frac{q f_s}{f_k(2^{\frac{1}{B}} - 1)} \quad (2.12)$$

where it can be observed, that the dependency on  $\delta\omega$  has disappeared.

In the process of computing the transform basis of the CQT, for each desired frequency bin at  $f_k$  a window function with length  $N_k$  is used. In figure 2.6(a)-(d) CQT atoms are compared to DFT atoms. It can be observed that, in contrast to the DFT atoms, the length of the CQT atoms decreases as their center frequency increases. Hence, the number of cycles  $\Theta$  in the CQT atoms is constant and given by

$$\Theta = \frac{f_k}{\Delta f_k} = \delta\omega Q = \frac{q}{2^{\frac{1}{B}} - 1} \quad (2.13)$$

being only dependent on the number of bins per Octave  $B$  and the scaling factor  $q$ . Since all atoms  $a_k(n)$  are only dilated and scaled versions of  $a_1(n)$ ,

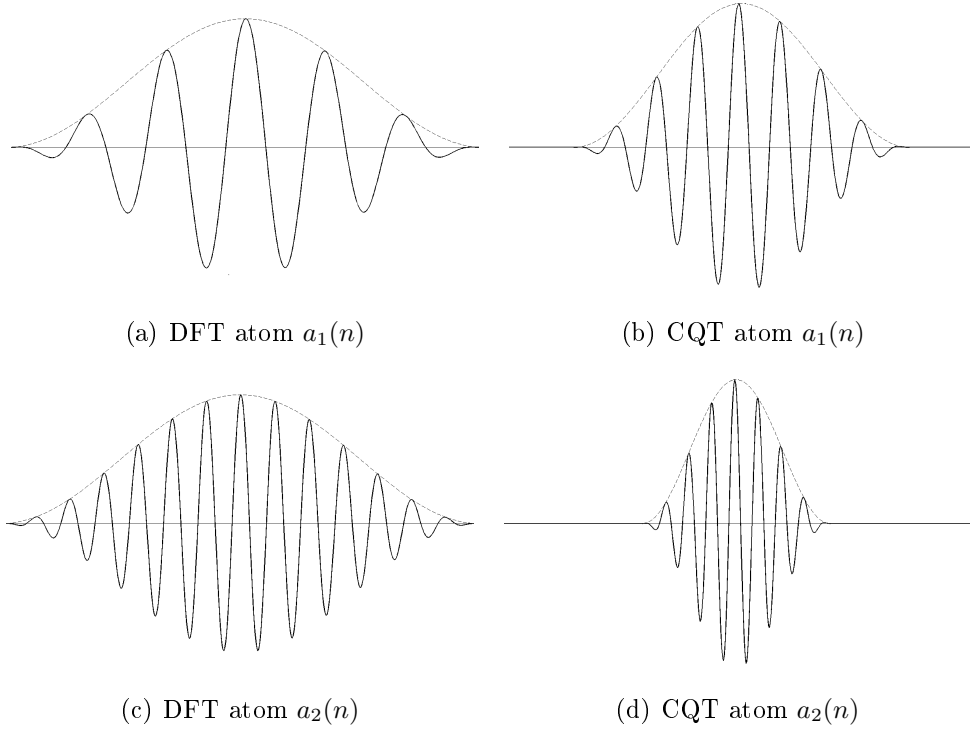


Figure 2.6: Real part of temporal atoms using a Hann window. (a) and (c) are DFT atoms for two different frequencies, (b) and (d) are CQT atoms for two different frequencies.

the CQT in fact is a wavelet transform [Mal09].

It is not computationally reasonable to calculate the coefficients  $X^{\text{CQ}}(k, n)$  at time instants  $n$  of the input signal. To enable signal reconstruction from the CQT coefficients, successive atoms can be placed  $H_k$  samples apart, where  $H_k$  is the hop size between atoms corresponding to frequency bin  $k$ . In order to analyze all parts of the signal properly and to achieve reasonable signal reconstruction (see section 2.7), values  $0 < H_k \lesssim \frac{1}{2}N_k$  are useful.

## 2.5 A Faster Algorithm

The computationally-efficient forward CQT proposed here is based on the principles proposed by Brown and Puckette in [BP92]. Therefore, the technique proposed in [BP92] is explained first, extensions are described in subsection 2.5.2.

### 2.5.1 Algorithm by Brown and Puckette

Calculating the CQT transform coefficients  $X^{\text{CQ}}(k, n)$ , the direct evaluation of (2.5) at one time instant  $n$  for an input signal  $x(n)$  obviously requires the computation of the inner products of the input signal with each of the transform bases. Figure 2.7(a) illustrates the real part of the transform bases  $a_k(n)$ , assuming here for simplicity  $B = 12$  bins per octave and a frequency range of only two octaves.

In [BP92] Brown and Puckette proposed a computationally more efficient way to compute the transform coefficients by utilizing the identity

$$\sum_{n=0}^{N-1} x(n)a^*(n) = \frac{1}{N} \sum_{j=0}^{N-1} X(j)A^*(j) \quad (2.14)$$

where  $X(j)$  denotes the DFT of  $x(n)$  and  $A(j)$  denotes the DFT of  $a(n)$ . Equation (2.14) holds for any discrete signals  $x(n)$  and  $a(n)$  and stems from Parseval's theorem [Mal09]. Note that in the sequel the factor  $\frac{1}{N}$  is omitted as this normalization can be applied to  $A(j)$  prior to summation.

Using (2.14), the CQT transform in (2.5) can be written as

$$X^{\text{CQ}}(k, N/2) = \sum_{j=0}^N X(j)A_k^*(j) \quad (2.15)$$

where  $A_k(j)$  is the complex-valued  $N$ -point DFT of the transform basis  $a_k(n)$  so that the bases  $a_k(n)$  are centered at the point  $N/2$  within the transform

frame. To be consistent with the terminology in [BP92],  $A_k(j)$  and  $a_k(n)$  will be referred to as the spectral and the temporal kernels, respectively.

Figure 2.7(b) illustrates the absolute values of the spectral kernels  $A_k(j)$  corresponding to the temporal kernels  $a_k(n)$  in figure 2.7(a). As observed by Brown and Puckette, the spectral kernels  $A_k(j)$  are sparse: most of the values being near zero because they are Fourier transforms of modulated sinusoids.

Therefore, the summation in (2.15) can be limited to values near the peak in the spectral kernel to achieve sufficient numerical accuracy. Near-zero values in  $A_k(j)$  can be omitted by setting values  $A_k(j) < \epsilon$  to zero, where  $\epsilon$  is a threshold that has to be chosen such that the error is negligible. This is the main idea of the efficient CQT transform proposed in [BP92]. It is also easy to see that the summing has to be carried out for positive frequencies only, followed by a multiplication by two.

For convenience, the spectral kernels  $A_k(j)$  are stored as columns in matrix  $\mathbf{A}$ . The transform in (2.15) can then be written in matrix form as

$$\mathbf{X}^{\text{CQ}} = \mathbf{A}^* \mathbf{X} \quad (2.16)$$

where  $\mathbf{A}^*$  denotes the conjugate transpose of  $\mathbf{A}$ . Matrices  $\mathbf{X}$  and  $\mathbf{X}^{\text{CQ}}$  have only one column each, containing the DFT values  $X(j)$  and the corresponding CQT coefficients, respectively.

Figure 2.8 illustrates how the CQT atoms capture the time-frequency plane. The representation thus computed produces minimum redundancy while enabling reasonable quality signal reconstruction (see section 2.7) if the window functions overlap by 75% for each bin in time domain, for example. However, the calculation of the CQT using (2.15) and (2.16), respectively, implies that the points in time  $n$  (where the atoms are centered on the time-frequency plane), can no longer be arbitrarily chosen for each bin  $k$ . Thus, the hop size of the kernels is the same for all frequency bins. This increases the redundancy and puts special demands on the window function (see section 2.8) in order to capture the signal uniformly across the time-frequency plane. On

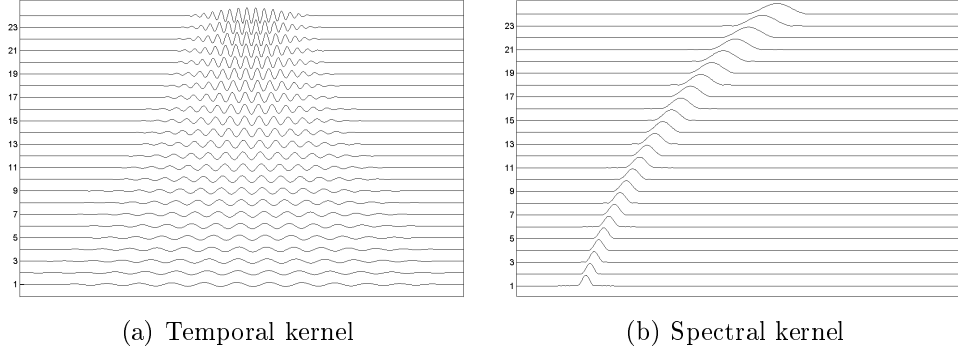


Figure 2.7: (a) illustrates the real part of the transform bases (temporal kernel) that can be used to calculate the CQT of two octaves, with 12 bins per octave. (b) shows the absolute values of the corresponding spectral kernel.

the plus side, using the same hop size for all bins increases the usability of the produced data structure.

### 2.5.2 Processing One Octave at a Time

Although increasing the computational efficiency, the method outlined in the previous subsection still exhibits two problems when broadband music signals, for example eight octaves from 60Hz to 16kHz, are considered. For the atom  $a_1(n)$  corresponding to the lowest frequency bin ( $f_1 = 60$  Hz), the window size  $N_1 \approx 25000$  for the sampling rate  $f_s = 44100$  and  $B = 24$ , hence the DFT transform blocks need to be at least  $N_1$  samples long<sup>3</sup>. On the other hand, the window size of the temporal kernel  $a_K(n)$  corresponding to the highest frequency bin  $f_K$  is  $N_K \approx 94$ . Two problems arise from this big range of window sizes: firstly, due to their small window sizes and the long DFT transform blocks, the spectral kernels for higher frequencies will have many non-zero coefficients and will no longer be very sparse. Secondly, in order to analyze all parts of the input signal adequately, the CQT transform for the highest frequency bins has to be calculated at least every  $N_K/2$  samples apart. Both of these factors reduce the computational efficiency of the

<sup>3</sup>If a standard FFT algorithm is used, the block length would be  $N_{\text{DFT}} = 2^{\lceil \log_2(N_1) \rceil}$ .



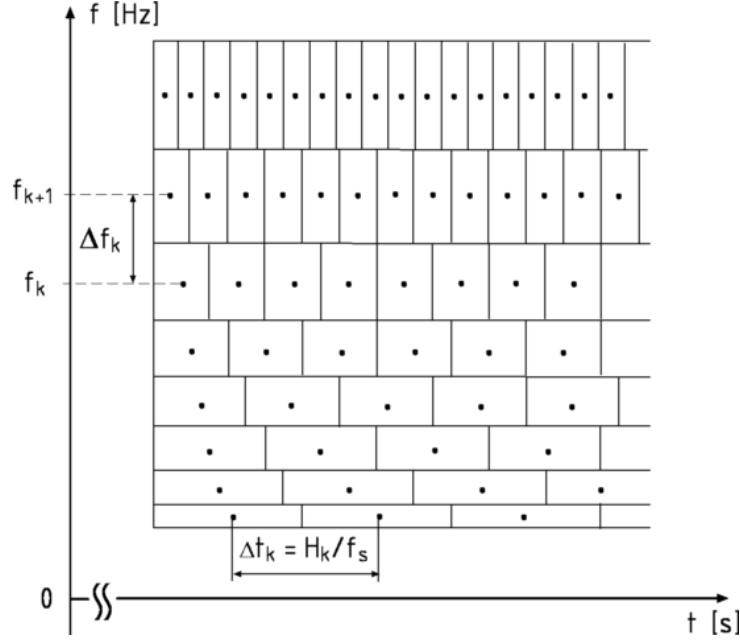


Figure 2.8: Spacing of time-frequency atoms for the CQT. The time resolution increases for higher frequencies and the frequency resolution increases for lower frequencies.

method.

To address these problems, two extensions are proposed here. The first is processing by octaves<sup>4</sup>. The spectral kernel matrix  $\mathbf{A}$  is chosen, such that it produces the CQT for the highest octave of interest only. After computing the CQT bins for the highest octave for the entire signal, the input signal is low-pass filtered and down-sampled by factor two. The same process is repeated to calculate the CQT bins for the next octave, using exactly the same DFT block size and the same spectral kernel  $\mathbf{A}$  (due to the down sampling of the input signal, the lengths of the atoms  $a_k(n)$  effectively has been doubled). This process is repeated iteratively until the desired number of octaves has been covered. Figure 2.9 illustrates this process.

Since the spectral kernel  $\mathbf{A}$  now represents frequency bins that are at maxi-

<sup>4</sup>J. Brown mentions this possibility already in [Bro91], although octave-by-octave processing was not implemented in [Bro91, BP92]

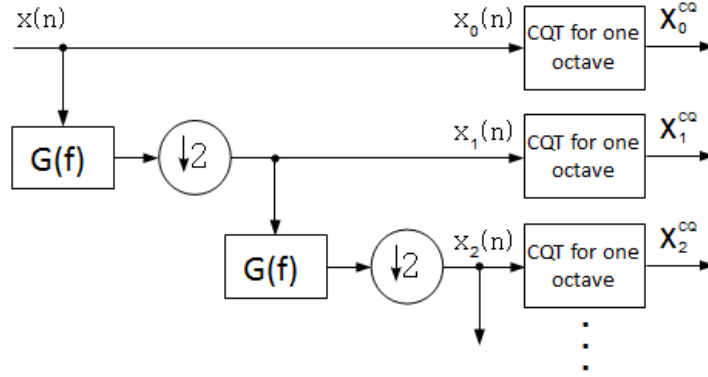


Figure 2.9: An overview of computing the CQT one octave at the time. Here  $G(f)$  is a low-pass filter and  $\downarrow 2$  denotes down-sampling by factor two.

imum one octave apart, the length of the DFT block can be made quite short (according to  $N_k$  of the lowest CQT bin) and the matrix  $\mathbf{A}$  is very sparse even for the highest-frequency bins.

Another computational efficiency improvement is obtained by using several temporally translated versions of the transform bases  $a_k(n)$  within the same spectral kernel matrix  $\mathbf{A}$ . In other words, successive columns of  $\mathbf{A}$  contain the DFTs of  $a_k(n)$  that have been temporally shifted to different locations. As a result, DFT transforms of the input signal  $x(n)$  to obtain the DFT spectrum  $X(j)$  in (2.15) need to be computed less often: if there are  $P$  successive atoms within the same spectral kernel, the DFTs need to be computed  $P$  times less often. Note, that this method also enables the use of a different number of atoms for different frequency bins. Figure 2.10(a)-(b) illustrates the general structure of the kernel matrix applied in this thesis. In the shown example, the number of bins per octave  $B = 12$ . By looking closely, it can be seen that the highest four kernel functions have the same center frequency, but correspond to four different temporal locations. The detailed structure of the spectral kernel will be discussed in Sec. 2.8; here it suffices to say that the kernel structure is crucial for high-quality reconstruction (inverse CQT) of the input signal  $x(n)$  from the CQT coefficients.

The transform for a single octave (indicated by "CQT for one octave" in

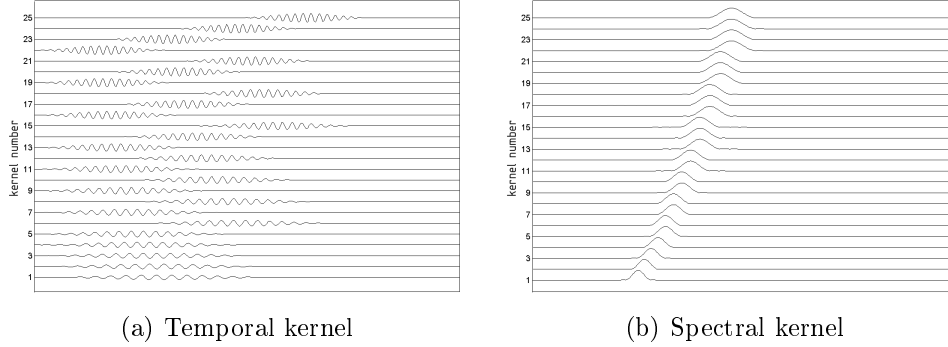


Figure 2.10: Illustration of the general structure of the kernel matrices that are used here. (a) shows the real part of the temporal kernel used to compute the CQT for one octave. (b) shows the absolute values of the corresponding spectral kernel.

figure 2.9) is defined as follows. Let  $x_d(n)$  denote a signal that is obtained by decimating the input signal  $d$  times by factor two. The sampling rate of  $x_d(n)$  is therefore  $f_s/2^d$ . The signal  $x_d(n)$  is blocked into DFT transform frames of length  $N_{\text{DFT}}$  which are positioned  $H_{\text{DFT}}$  samples apart (i.e., successive frames overlap by  $N_{\text{DFT}} - H_{\text{DFT}}$  samples). Each frame is Fourier transformed using a rectangular window and the resulting spectrogram is stored in a matrix  $\mathbf{X}$ , where column  $m$  contains the complex-valued spectrum of frame  $m$  (positive frequencies only). Then the CQT transform  $\mathbf{X}_d^{\text{CQ}}$  for this octave  $d$  is calculated as

$$\mathbf{X}_d^{\text{CQ}} = \mathbf{A}^* \mathbf{X}_d \quad (2.17)$$

where  $\mathbf{A}^*$  is the conjugate transpose of the complex-valued spectral kernel matrix for one octave as described above. The column  $m$  of  $\mathbf{X}_d^{\text{CQ}}$  contains the CQT coefficients representing DFT block  $m$  and the different rows of  $\mathbf{X}_d^{\text{CQ}}$  correspond to the different spectral kernels that are stored in the different columns of matrix  $\mathbf{A}$ .

The above process is repeated for each successive octave, as illustrated in figure 2.9. Note that the kernel remains the same for all octaves. Also, the

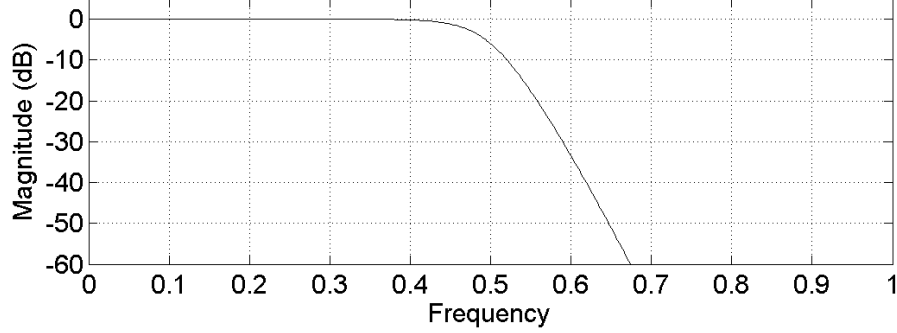


Figure 2.11: Magnitude response of the low-pass filter  $G(f)$ .

DFT length  $N_{\text{DFT}}$  (in samples) remains the same despite the decimations, therefore the effective FFT length (in seconds) doubles in each decimation. The first octave is computed using signal  $x_0(n)$ , which is identical to the input,  $x(n)$ .

The decimated signals  $x_d(n)$  are obtained from  $x_{d-1}(n)$  by low-pass filtering and down-sampling by factor two. For the low-pass filter  $G(f)$ , zero-phase forward-and-reverse filtering with a sixth-order Butterworth IIR filter that has a cut-off frequency  $f_s/4$  is used. Forward-and-reverse filtering means that after filtering in the forward direction, the filtered sequence is reversed and run back through the filter and the result of the second filtering is then reversed once more. The result has precisely zero phase distortion and magnitude modified by the square of the filter's magnitude response. Figure 2.11 shows the magnitude response of the low-pass filter  $G(f)$  (square of the magnitude response of sixth-order Butterworth filter). Down-sampling by factor two is then done simply by removing every second sample of the time-domain signal.

A final practical consideration is to deal with the beginning and end of the input signal  $x(n)$ . This problem is addressed by padding  $2^{D-1}N_1$  zeros at the beginning of the signal and  $2^{D-1}N_{\text{DFT}}$  zeros at the end of the signal, where  $N_1$  is the window length of the lowest-frequency bin within the one-octave kernel,  $D$  is the number of octaves calculated, and  $N_{\text{DFT}}$  is the length of the

DFT frame. The zero padding is done before any of the CQT computations, and the zeros are then removed at the inverse transform stage. Note that a smaller number of zeros is needed, if the zero padding is done separately for each octave, but this would make the implementation less clear and therefore it is assumed that the number of zeros padded is negligible in comparison to the length of the input signal  $x(n)$ .

## 2.6 Computational Complexity

Let  $L$  denote the length of the input signal  $x(n)$  after the zero padding at the beginning and the end. The number of DFT frames  $m$  to cover the entire signal before any decimation is  $\lfloor (L - N_{\text{DFT}})/H_{\text{DFT}} \rfloor + 1$ . For the next octave, the number of fast Fourier transforms (FFTs) is roughly twice smaller,  $\lfloor (L/2 - N_{\text{DFT}})/H_{\text{DFT}} \rfloor + 1$ , in fact a bit more than twice smaller. For the next octave, the number of DFT transforms is roughly four times smaller, and so forth. Since  $1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \dots \approx 2$ , the total number  $C$  of FFTs to compute is

$$C \leq 2 (\lfloor (L - N_{\text{DFT}})/H_{\text{DFT}} \rfloor + 1) \quad (2.18)$$

*regardless* of the number of octaves computed.

For each of the  $C$  DFT frames, the complex-valued DFT spectrum (a column vector) has to be multiplied by the conjugate transpose of the spectral kernel  $\mathbf{A}^*$  (see (2.17)). However, since  $\mathbf{A}$  is sparse, the number of multiplications is quite small. In our reference Matlab implementation, the matrix is implemented as a sparse matrix, therefore also the memory complexity of storing  $\mathbf{A}$  is quite low. The exact number of non-zero elements in  $\mathbf{A}$  depends on the kernel structure and the threshold below which the near-zero elements are rounded to zero (see 2.15). The number of low-pass filterings is proportional to the number of octaves  $D$  and causes a non-negligible computational load too.

To compare the complexity of the proposed method with that of the original

method by Brown and Puckette [BP92], consider a case where the CQT is computed over an eight-octave range. If atoms over all octaves are stored into a single kernel, the frequency kernels in the highest octave will have  $2^7 = 128$  times more non-zero elements than the corresponding atoms in the lowest octave, and the number of multiplications in (2.16) increases in the same proportion. The lengths of the DFT transform frames, in turn, have to be 128 times larger in order to accommodate the lowest-frequency atoms without decimation.

## 2.7 Reconstruction from the CQT Coefficients

The proposed methods for further reducing the computational complexity of the CQT extend the methods proposed in [BP92] and thus make the CQT more attractive to be applied to musical signal analysis tasks. However, if the signal is to be altered in the frequency domain, a method to reconstruct the time domain signal from its CQT coefficients is required. Mathematically speaking, invertibility of a transform is assured when the transform kernel is a (orthonormal) basis - implying that the transform is non-redundant. However, since the conditions to a bases are very restrictive and often very hard or even impossible to meet, frame theory<sup>5</sup> has proven to be a more flexible tool to enable invertibility for redundant transforms while not demanding orthonormality [Chr03][KC07]. The approach to signal reconstruction from its CQT coefficients as proposed in this thesis, however, is straight forward and not mathematically rigorous, while being strongly related to frame theory [Dör02]. Loosely speaking, a signal can be reconstructed from its transform coefficients, if the transform kernel is designed so that it captures the time-frequency plane uniformly (in the range of interest). In section 2.8 the kernel design proposed here is described. For now it is assumed that the kernel is properly designed, hence the reconstruction process is straight forward.

Figure 2.12 shows an overview of the inverse CQT transform (ICQT), where

---

<sup>5</sup>It is generally acknowledged, that frames were first mentioned in [DS52].

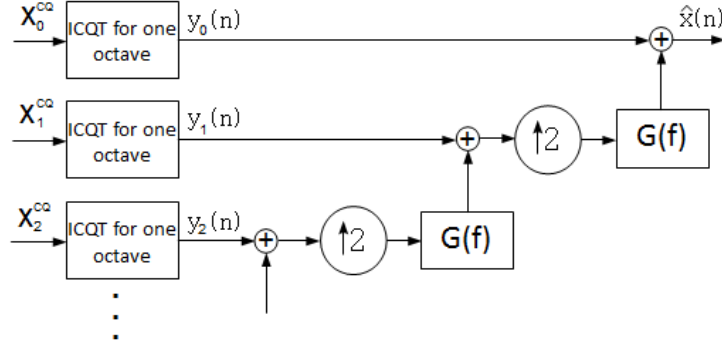


Figure 2.12: An overview of the inverse CQT transform (ICQT), where an approximation  $\hat{x}(n)$  of the input signal  $x(n)$  is reconstructed from the octave-wise CQT coefficient matrices  $\mathbf{X}_d^{\text{CQ}}$ .

an approximation  $\hat{x}(n)$  of the input signal  $x(n)$  is reconstructed from the octave-wise CQT coefficient matrices  $\mathbf{X}_d^{\text{CQ}}$ . The process is analogous to the forward transform, except that all is done in reverse order.

The block indicated by "ICQT for one octave" in figure 2.12 corresponds to the reconstruction of a time-domain signal  $y_d(n)$  that represents only one-octave range of the input signal  $x(n)$ . The signal  $y_d(n)$  is obtained as follows. First, an inverse spectral kernel  $\mathbf{V}$  is applied to reconstruct the complex-valued DFT bins within this single octave:

$$\mathbf{Y}_d = \mathbf{V}^* \mathbf{X}_d^{\text{CQ}} \quad (2.19)$$

where the column  $m$  of  $\mathbf{Y}_d$  contains the complex-valued DFT approximating the column  $m$  of  $\mathbf{X}_d$  in (2.17), but only over the frequency bins that belong to this octave - outside this octave,  $\mathbf{Y}_d$  is zero. The structure of the inverse spectral kernel  $\mathbf{V}$  will be described in section 2.8: here kernels for which  $\mathbf{V} = \mathbf{A}^*$  are used, meaning that the inverse kernel is a conjugate transpose of the forward transform kernel.<sup>6</sup>

Since each column of  $\mathbf{Y}_d$  only contains the DFT spectrum for the positive fre-

<sup>6</sup>Note that then  $\mathbf{Y}_d = \mathbf{V}^* \mathbf{X}_d^{\text{CQ}} = \mathbf{A} \mathbf{A}^* \mathbf{X}_d$ , where multiplication by  $\mathbf{A} \mathbf{A}^*$  actually implements a near-perfect one-octave bandpass filter.

quencies, each column is augmented with its complex conjugate spectrum to reconstruct the negative frequencies (for real-valued time-domain signals, the DFT spectrum is conjugate-symmetric). The resulting columns are inverse DFT transformed to obtain the time-domain signals within each DFT block, and successive DFT blocks are then overlap-added to construct the entire signal  $y_d(n)$  over time (note that time-domain windowing of the DFT-blocks at the synthesis stage is included in the inverse CQT kernel  $\mathbf{V}$ ).

The resulting time-domain signal  $y_d(n)$  contains a reconstruction of one octave of the original input signal  $x(n)$ . This signal is added to a signal that already contains a reconstruction of all the lower octaves ( $d+1, d+2, \dots, D-1$ ) in order to obtain a signal  $\hat{x}_d(n)$  that approximates the input signal for octaves  $d, d+1, \dots, D-1$ . The signal  $\hat{x}_d(n)$  is then up-sampled by factor two by inserting zeros between the original samples, multiplying the signal by two, and low-pass filtering using zero-phase forward-and-reverse filtering with a sixth-order Butterworth IIR filter having cut-off frequency  $f_s/4$  (the same that was used at the analysis stage).

The above process is repeated for each octave at a time, as illustrated in figure 2.12. After reconstructing all the octaves,  $d = 0, 1, \dots, D-1$ , the resulting signal  $\hat{x}_0(n) \equiv \hat{x}(n)$  is an approximate reconstruction of the input signal  $x(n)$ .

The computational complexity of the inverse transform is approximately the same as that of the forward transform: here, instead of FFTs, inverse FFTs are computed, and instead of the spectral kernel, the inverse kernel is applied. Since the used inverse kernel

$\mathbf{V} = \mathbf{A}^*$ , the inverse kernel is sparse too. The number of low-pass filtering operations is the same as that at the forward transform stage.



## 2.8 Kernel Design

To enable reconstruction of the input signal from its CQT coefficients, the transform atoms need to cover the time-frequency plane uniformly. Since the transform kernel, as mentioned in section 2.5.2, contains frequency bins that cover one octave, the requirement of uniform coverage considering the design of the kernel matrix is confined to this one-octave range, too. Splitting up this requirement this means, that in time domain, successive window functions  $w(n)$  for bin  $k$  should sum up to unity over the entire signal. In frequency domain, on the other hand, the spectral kernels (bins) should sum up to unity. In the case of an analysis-synthesis system (CQT followed by ICQT) using the same kernel for both analysis and synthesis, this requirement is directed at the product  $\mathbf{A}\mathbf{A}^*$ . Hence, in time domain, the *squares* of successive window functions  $[w(n)]^2$  have to sum to unity because the signal will be windowed twice at the time-frequency location of each atom: once when applying  $\mathbf{A}^*$  for the CQT, and second time when applying  $\mathbf{V}^* \equiv \mathbf{A}$  for the ICQT (thus audible artifacts are avoided if the signal is manipulated in the CQT transform domain).

Typically, then, the window function  $w(n)$  is defined to be the square root of one of the commonly used window functions (e.g. Hann or Blackman). For analysis-only applications, the square root can be omitted to improve the time-frequency localization properties of the window.

As described in 2.5.2, the kernel matrix contains several temporal translated atoms. Thus, the window functions  $w(n)$  for bin  $k$  have to sum up to unity within the kernel, as well as over successive, overlapping DFT blocks. Most window functions, e.g. the Hann window, sum to a constant value only when the distance between successive windows  $H_k = \frac{1}{z}N_k$ , where  $z \geq 2$  is an integer and  $N_k$  is the window size for atom  $k$ . For an individual frequency bin  $k$ , the DFT frame hop  $H_{\text{DFT}}$  can be chosen to be an integer multiple of  $\frac{1}{z}N_k$  so that exactly an integer number of time-shifted atoms would fit between the beginnings of frame  $m$  and  $m+1$ , and these time-shifted atoms would be stored in the kernel  $\mathbf{A}$ . However, this requirement for  $H_{\text{DFT}}$  cannot

be simultaneously satisfied for all frequency bins  $k$  with different  $N_k$ . A reasonable solution is obtained by using a relatively large DFT frame size, in which case  $H_{\text{DFT}}$  can be made large relative to atom sizes  $N_k$ , and thereby  $H_{\text{DFT}}$  approximately divisible by  $\frac{1}{z}N_k$  for all  $k$ . This approach leads to a reasonable quality results, however, there are some disadvantages. Firstly, the obtained data structure is inconvenient to work with, since there is a different number of CQT coefficients for every bin  $k$  and thus the bins are not temporally synchronized. Secondly, due to the large DFT frame size, the spectral kernel  $\mathbf{A}$  is less sparse and thus computationally less efficient.

Due to this drawbacks, the kernel structure proposed here uses atoms that are temporally synchronized *within each octave*. This means that the temporal atoms within the kernel are centered at the same, successive points in time for all bins. The spacing between those points in time is defined by the common hop size  $H_{\text{ATOM}}$ , hence the individual hop size factor and the relative hop size  $H_{\text{ATOM}}/N_k$ , respectively, is different for each bin, but only up to a factor of two since the kernel contains only atoms covering one octave.

In order to synchronize the atom positions in time also across octaves, the position of the first atom stack within one DFT frame is not arbitrary but has to be an integer multiple of the common atom hop size.

Figure 2.13 illustrates the error  $e_r$  due to time-domain ripple for different window functions over the overlap factor. The overlap factor is given by  $1 - H_{\text{ATOM}}/N_k$  and the error is defined by

$$e_r = \sqrt{\frac{\sum_L (\nu - \frac{1}{L} \sum_L \nu)^2}{\sum_L \nu^2}} \quad (2.20)$$

where  $\nu$  is obtained by summing successive window functions and truncating at its ends to eliminate edge effects and  $L$  is length of  $\nu$ .

From figure 2.13 it can be seen that, e.g. for the Hann window,  $e_r = 0$  only for overlap factors  $\frac{z}{z+1}N_k$ ,  $z \in \mathbb{N}$  - otherwise  $e_r \gg 0$ . Since the overlap factors vary across bins in the proposed kernel structure, using a Hanning window

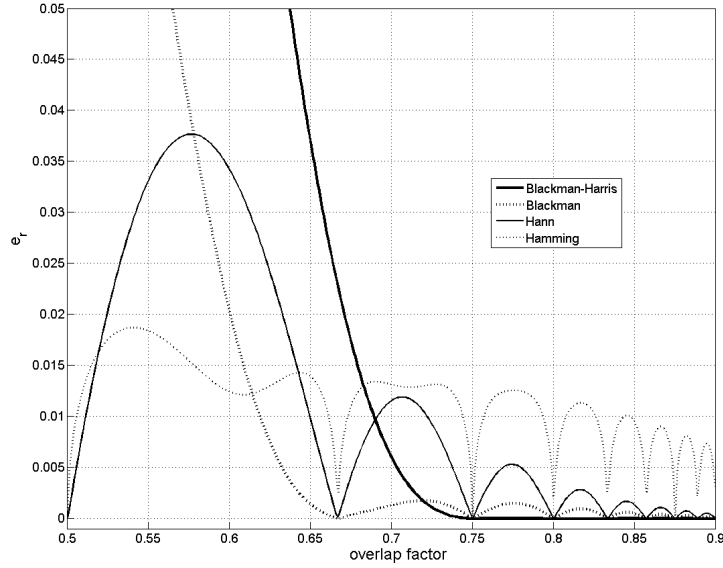


Figure 2.13: Error  $e_r$  due to time domain ripple over the overlap factor for different window functions.

will therefore result in a low quality of the reconstructed signal. This is also true for the Hamming window as depicted in figure 2.13. The Blackman window produces a lower error for overlap factors greater than 66%, however, the best choice for the given kernel structure is the Blackman-Harris window, since the error is almost negligible for a wide range of overlap factors, as long as the overlap factor  $\geq 75\%$  (from here the error still gets progressively smaller). Thus, using the Blackman-Harris window, successive windows will sum up to approximately a constant value which can be normalized to unity.

The parameter  $H_{\text{ATOM}}$  determines a trade-off between reconstruction quality (SNR of the signal reconstructed by ICQT) and redundancy of the representation. Having small value of  $H_{\text{ATOM}}$  leads to high quality, but also more redundancy, that is, larger number of CQT coefficients in proportion to the number of samples in the input signal. However, a default value for  $H_{\text{ATOM}}$  is easy to calculate: it is recommended to use  $H_{\text{ATOM}}/N_K \approx \frac{1}{4}$ , where  $N_K$  denotes the length of the shortest atom within the one-octave kernel. This leads to redundancy factors around five and reasonable quality of the reconstructed signal (see section 2.11).

The requirement of summation of the atoms (bins) to unity in the frequency domain is fulfilled, by choosing  $B \geq 12$ . Figure 2.14(a)-(d) shows the impulse responses of  $\mathbf{A}\mathbf{A}^*$  for different values for  $B$ . It can be observed, that as  $B$  increases, the impulse response gets increasingly flat. This behavior is best understood by looking at one individual bin and its right and left neighbor: if  $B$  is large, the distance between the bin and its two neighbors is very similar - as well as their bandwidths, since their lengths  $N_k$  are only slightly different. Thus, locally, the CQT bins behave very much like DFT bins and therefore sum up to approximately unity. This approximation gets increasingly accurate with increasing values for  $B$ .

The quality of the reconstruction can thus be increased, by increasing the number of bins per octave ( $B$ ) and by decreasing the hop size. Detailed figures and plots concerning the quality of reconstruction for different kernel parameters are provided in section 2.11.

## 2.9 Redundancy

The redundancy factor  $R$  of the proposed CQT transform is given by

$$R = \frac{2C_{\text{CQT}}}{C_{\text{IN}}} \quad (2.21)$$

where  $C_{\text{CQT}}$  and  $C_{\text{IN}}$  denote the amount of CQT coefficients and the amount of samples in the input signal, respectively. The factor 2 is due to the fact that CQT coefficients are complex-valued.

The amount of CQT coefficients produced by processing the highest octave is given by

$$C_{\text{OCT}} = \frac{C_{\text{IN}}B}{hN_K} \quad (2.22)$$

where  $h = H_{\text{ATOM}}/N_K$  is the atom hop size relative to the length  $N_K$  of the shortest atom (highest-frequency bin).

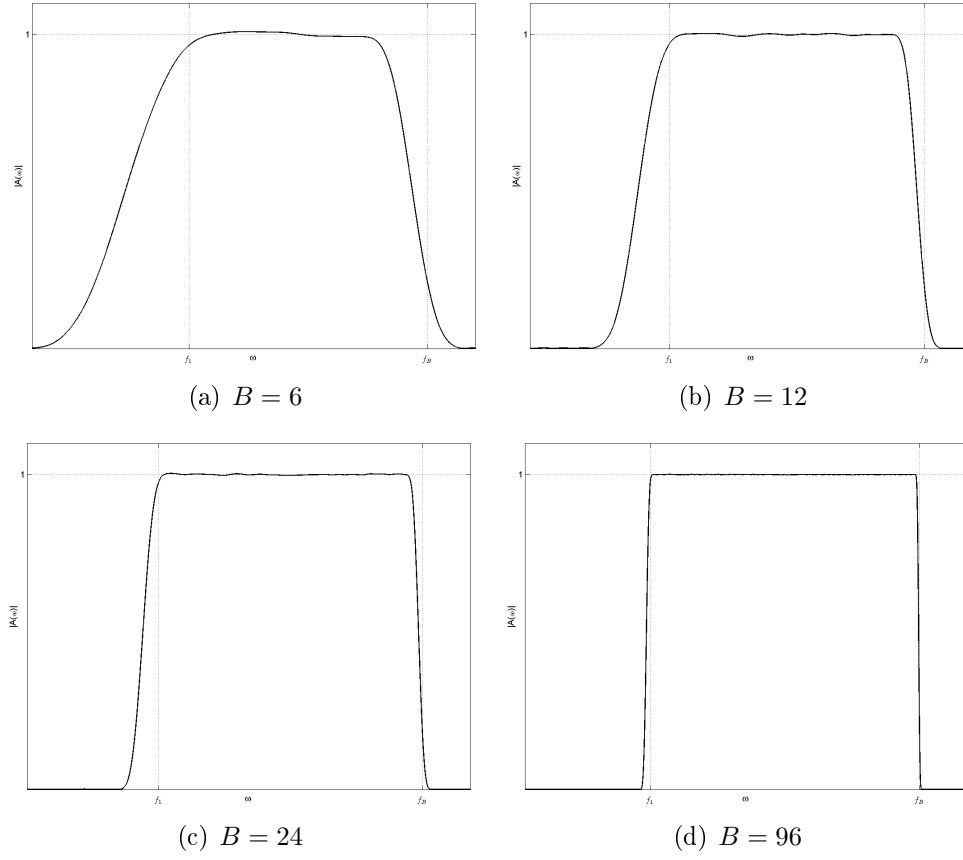


Figure 2.14: Transfer functions of  $\mathbf{AA}^*$  for different values of bins per octave.

Substituting the length  $N_K$  from (2.12)

$$C_{\text{OCT}} = \frac{C_{\text{IN}} f_K B (2^{\frac{1}{B}} - 1)}{h q f_s} \approx \frac{0.7 C_{\text{IN}} f_K}{h q f_s} \quad (2.23)$$

where the latter approximation is obtained by noting that  $B(2^{\frac{1}{B}} - 1) \approx 0.7$  when  $B \geq 12$ . Here it is assumed that the number of bins per octave  $B \geq 12$  (however, this is also a reasonable approximation for  $B < 12$ ).

Since the length of the input signal decreases by the factor of two at each decimation, it is easy to see from (2.23) that the number of CQT coefficients decreases by the same factor for each octave down. Therefore, the overall amount of data for a large number of octaves is  $C_{\text{OCT}}(1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \dots) \approx$

$2C_{\text{OCT}}$ . Substituting this to (2.21), the overall redundancy of the CQT transform is

$$R = \frac{2 \times 2 \times C_{\text{OCT}}}{C_{\text{IN}}} = \frac{2.8f_K}{h q f_s}. \quad (2.24)$$

where the redundancy is proportional to the highest frequency analyzed,  $f_K$ , and inversely proportional to the relative atom hop size  $h$  and the Q-value scaling factor  $q$  (see 2.11).

## 2.10 Data Structure

Due to the proposed kernel structure, the data that is produced by the CQT for one octave is of matrix form. However, since the input signal is down sampled by factor 2 to calculate the CQT of the next octave, the number of points where  $X^{\text{CQ}}(k, n)$  is evaluated decreases by factor 2 as well. Figure 2.15 depicts the produced data points of the CQT proposed here in the time-frequency plane. Note, that even though the number of coefficients for one octave is equal for all frequency bins (the absolute hop size is constant), the time resolution decreases from high to low frequencies since the atoms lengths vary. The default representation of the CQT coefficients in the proposed toolbox, is in the form of a sparse matrix, where for the second to highest octave one zeros is inserted after every calculated coefficient, for the third to highest octave three zeros are inserted after every calculated coefficients, and so on.

In order to allow the user an easy access to the information without mind-ing the inherent time sampling technique, the reference implementation of the toolbox in Matlab contains interface tools to access the CQT data in a representation that is regularly sampled in time. This 'rasterized' CQT data structure is achieved by interpolating the magnitudes between the time points  $X^{\text{CQ}}(k, n)$  that have been computed by the CQT (the sampling frequency of the rasterized CQT representation is set according to the atom hop size of the highest octave). With the interface tools, the user can obtain the entire CQT matrix representing the input data, or access only extracts of it.

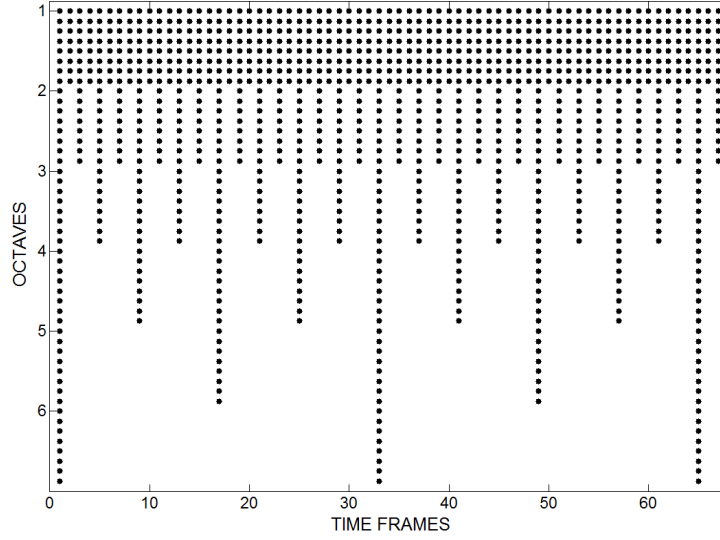


Figure 2.15: Points in the time-frequency plain where  $X^{\text{CQ}}(k, n)$  is evaluated.

It is also possible to access only a certain time slice  $n$  of the CQT transform  $X^{\text{CQ}}(k, n)$  or all the CQT coefficients of a certain frequency bin over time.

Another important tool is a function for plotting the magnitude of the CQT transform  $X^{\text{CQ}}(k, n)$  in a form similar to the DFT spectrogram using the described interpolation technique. Figure 2.17 shows the CQT of a signal containing bass, guitar, synthesizer sounds and a singing voice thus generated. The decrease of temporal resolution due to the increase of frequency resolution towards low frequencies is demonstrated in figure 2.16. The signal depicted contains pure sinusoids with their frequencies being a major triad apart from each other.

If a rasterized CQT representation is desired not only for the magnitudes, but the complex coefficients, the toolbox provides functionality to actually *calculate* all desired data points with reasonable computational efficiency. With this tool, several time shifted versions of the spectral kernel are applied to the DFT blocks, that is, the number of FFTs as well as the number of filter operations (down-sampling filters) is unchanged.

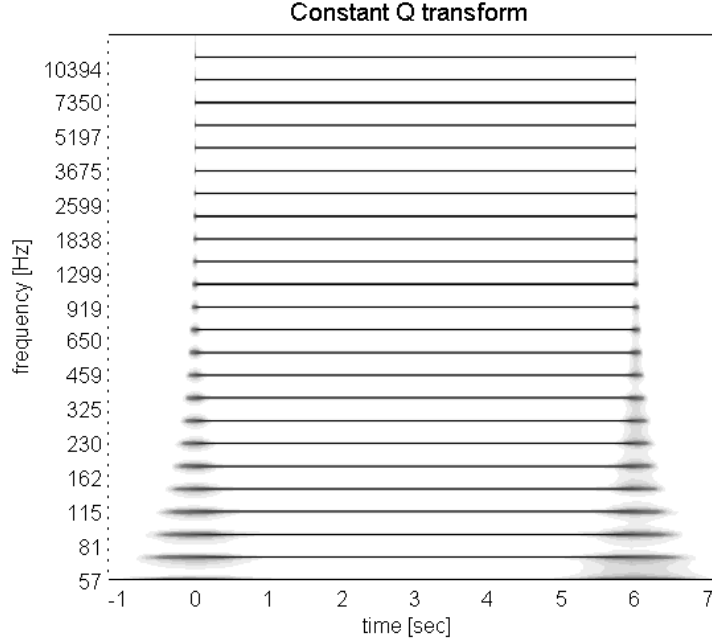


Figure 2.16: CQT of a six-second signal containing sinusoids with frequencies four semitones apart from each other.

## 2.11 Quality of Reconstruction

Figure 2.18 shows the quality of the reconstructed time-domain signal  $\hat{x}(n)$  as a function of the redundancy  $R$  (see (2.21)) and different window functions  $w(n)$ . Here the number of bins per octave was  $B = 48$ . In this plot, the redundancy was increased by decreasing the relative hop-size  $h$  of the shortest atom from 0.6 to 0.1. A constant  $Q$  scaling factor  $q = 1$  has been used, which means that only time-domain redundancy has been added. Using  $q = 0.5$  (frequency-domain oversampling) would improve the quality further by  $\approx 3\text{dB}$  but also increase the redundancy by factor two, therefore results are shown only for  $q = 1$ .

The input signal was Gaussian random noise, bandpass filtered to contain only frequency components within the range being analyzed: we used  $f_K = f_s/3 = 14.7\text{kHz}$  for the highest CQT bin, and analyzed eight octaves down to 57Hz. Random noise represents a “worst case”: for music signals, the



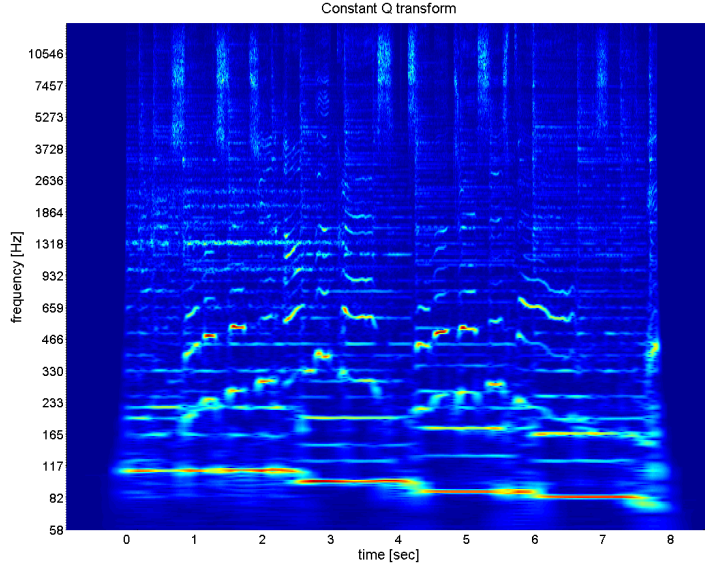


Figure 2.17: CQT transform of a music excerpt containing singing, acoustic guitar, bass, and synthesizer sounds.

reconstruction quality is typically a few decibels better. Redundancy factors were calculated by substituting  $f_K = 14.7\text{kHz}$  and  $f_s = 2 \times 14.7\text{kHz}$  into (2.24), where the latter is the sampling rate required to represent the time domain signal up to  $14.7\text{kHz}$ .<sup>7</sup>

Signal-to-noise ratios (SNRs) were calculated by comparing the reconstructed signal  $\hat{x}(n)$  after inverse CQT with the input signal  $x(n)$ :

$$\text{SNR} = 10 \log_{10} \frac{\sum_n [x(n)]^2}{\sum_n [\hat{x}(n) - x(n)]^2} \quad (2.25)$$

It can be observed that the choice of the window function has crucial influence on the quality of the reconstruction. For a very low redundancy, corresponding to a large atom hop size, the highest SNR values are achieved using a Hann window. For the redundancy range from 3 to 4.5 the Blackman window performs best, whereas for  $R > 4.5$  the Blackman-Harris window

<sup>7</sup>Note that if an input signal is to be analyzed up to the Nyquist frequency ( $f_K = f_s/2$ ), the input signal has to be slightly up-sampled (say,  $f'_s = \frac{4}{3}f_s$ ) before applying the proposed method, since the low-pass filter  $G(f)$  in Fig. 2.11 is not ideal.

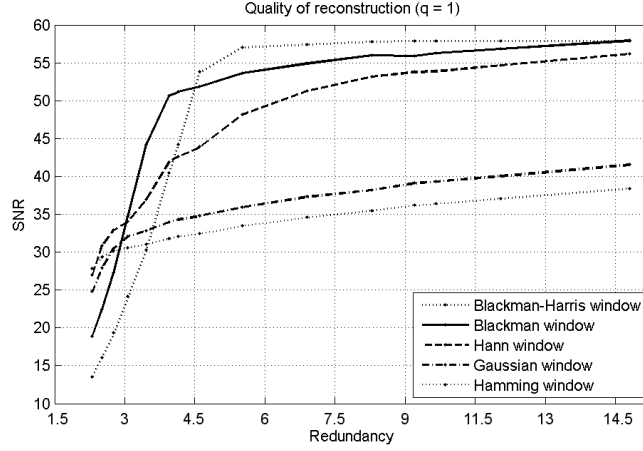


Figure 2.18: Quality of the reconstructed signal as a function of the window function  $w(n)$  and the redundancy  $R$ .

achieves the highest SNR values. This result can be explained by considering the time ripple of the different window functions for varying hop sizes. The Blackman-Harris window shows large time ripple with small overlap values, but for overlap values greater than 75%, consecutive windows sum up to unity almost perfectly. The Blackman window has similar properties but converges slower to a low level of ripple. Figure 2.18 shows that using Blackman-Harris window, SNR values of about 55dB are achieved with  $R \approx 5$ .

Fig. 2.19 shows the quality of the reconstructed time-domain signal  $\hat{x}(n)$  as a function of the redundancy  $R$  (see (2.21)) using a Blackman-Harris window and different values for  $B$  (bins per octave). It can be observed that the quality of the reconstructed signal improves by increasing the number of bins per octave, achieving up to 60 dB SNR using  $B = 96$ . The property of the Blackman-Harris window obtaining the highest SNR values already for low redundancy values is independent of the number of bins per octave.

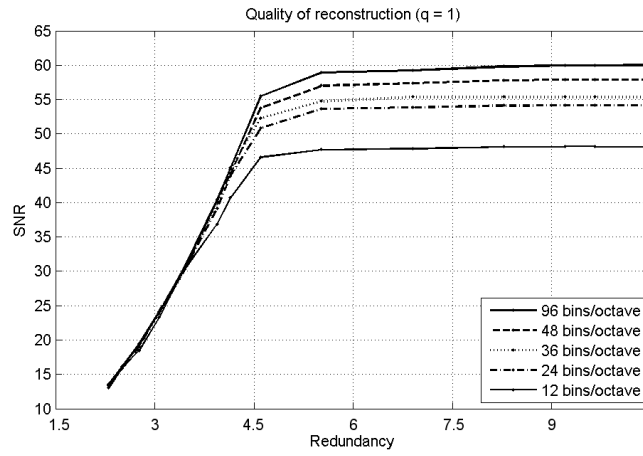


Figure 2.19: Quality of the reconstructed signal as a function of the number of bins per octave,  $B$ , and redundancy  $R$ .



## Chapter 3

# Time-Scale Modification of Audio Signals

The aim of altering the time-scale of an audio signal, is to change the overall duration of the signal while retaining the perceived pitch and timbre of the material. The ultimate goal is to obtain an audio signal that sounds as if the piece of music was played at a different tempo or the speaker has spoken at different rate, respectively. This very common digital audio effect is a standard tool in todays music production environments and has a long history of research leading to steadily improving algorithms, able to produce high quality results. However, the mitigation of artifacts in the modified audio signal is still an issue.

The problem of time-scale modification can either be tackled in the time domain or in the time-frequency domain (more generally, in a transform domain). The principle of early approaches that operated directly in the (analog) time domain, such as [Gar53] and [FEJ54], was to simply segment the signal into non-overlapping frames and discard or repeat every  $k^{th}$  frame to achieve the desired compression ratio. In [Lee72] a digital implementation of this process is presented. Obviously, this simple approach leads to artifacts caused by the discontinuities at the boundaries of the non-overlapping

segments and pitch distortions.

A first attempt to mitigate these errors, is to divide the input signal into overlapping segments of a fixed size using an input hop size  $R$  and overlap-add (OLA) the segments using the output hop size  $\alpha R$ , whereas overlapping segments are cross-faded to produce a smooth output signal ([ABLS02]). In [RW85], this methodology has been extended towards what is called the Synchronous Overlap and Add (SOLA). Here the cross correlation of the overlapping segments is computed and the output hop size is adjusted according to the time lag where the cross correlation has its maximum. A variation of the SOLA algorithm is the Pitch Synchronous Overlap and Add (PSOLA) proposed in [MC90] [HMC89], especially aiming at voice processing. In this approach the segmentation of the input signal is based upon the pitch periods, which are obtained by first analyzing the input signal. There is number of related algorithms, such as the Adaptive Overlap and Add (AOLA) [LF99] or the Waveform Similarity Overlap and Add (WSOLA) technique [VR93], that are all operating directly on the unprocessed time-domain input signal. The benefit of such approaches is the low computational complexity, however, they are restricted to speech or monophonic audio signals (or polyphonic music for small time-scaling factors), since they are based on the existence of quasi-periodic signals. For these classes of signals, time-domain approaches are able to produce high quality output signals.

In order to apply time-domain techniques to polyphonic audio signals, in [SVW94] a sub-band approach to time-scale modification has been proposed. Here the complex (i.e. polyphonic) input signal is first split into several less complex sub-bands using perfect reconstruction filter banks. Since the requirement of the existence of quasi-periodic signals is likely to be met in these sub-bands, time-domain techniques discussed above can be applied. After time-scaling the individual sub-bands they are added together to form the output signal. In [SVW94] it is stated, that the main source of error in this approach is the problem of sub-band synchronization, an issue that, for example, has been addressed in [DL04]. For a comprehensive comparison of different time domain time-scale modification algorithms, the interested

reader is referred to [CDL06].

Since the scope of this thesis is the assessment of the applicability of the CQT (plus inverse) to digital audio signal processing, frequency domain techniques for time-scale modifications are discussed more specifically in the sequel. Amongst these approaches, phase vocoder techniques are the most prominent ones and seem especially suitable for replacing the commonly used DFT with the CQT due to its logarithmic frequency resolution. Thus, the standard DFT based phase vocoder will be described in the next section. For the sake of completeness, however, two other frequency domain approaches to time-scale modification only be mentioned and referenced here.

In [GL84] an iterative method is proposed to obtain a signal estimation from a modified STFT representation or only from the modified STFT magnitudes, respectively. The method in [GL84] is based on the fact, that not every STFT representation is valid in the sense that there is no signal that has this particular STFT representation. Time-scale modification is then performed by modifying the magnitudes of the original STFT representation so that the desired scaling factor is achieved. Using an initial estimate for the phase values, the output signal is estimated iteratively.

Sound modifications, such as time-scaling, can also be achieved by representing an audio signal as a sum of slowly varying sinusoidal trajectories. This sinusoidal modeling approach was proposed in [MQ86] and [SS87] and comprises three analysis stages: spectral peak detection, estimation of exact frequency, amplitude and phase values for spectral peaks and partial tracking, that is linking spectral peaks to several sinusoidal trajectories. Using these extracted and potentially modified parameters, the audio signal can be resynthesized by synthesizing and summing the sinusoidal trajectories. The synthesis step is usually performed by an oscillator bank in time domain (additive synthesis). Since the sinusoidal model, however, is not able to represent noisy signal parts, in [SS90] and [Ser89] the model has been extended towards a sinusoidal (deterministic) plus noise (stochastic) representation. This technique is commonly referred to as *spectral modeling synthesis*

(SMS) and aims at modeling time-varying spectra as a collection of sinusoids controlled through time by piecewise linear amplitude and frequency envelopes (deterministic part) and a time-varying filtered noise component (stochastic part). In the analysis stage, the identified sinusoidal trajectories are removed by spectral subtraction and the remaining residual (noise) is modeled as white noise through a time-varying filter. The system proposed in [SS90] is designed to model speech and single instrument signals whereas an ideal time/frequency resolution trade-off is achieved by employing pitch-synchronous DFT window lengths. Since this approach cannot be applied to polyphonic music signals, a multiresolution approach to sinusoidal modeling has been proposed in [LVS98]. Furthermore, as transient signal components are very expensive to be modeled as a sum of sinusoids and sharp attacks can not be modeled by filtered noise either, in [LI98] a sinusoidal + transient + noise model for audio representation has been proposed. The transition from a multiresolution DFT system to a spectral modeling approach based on the CQT seems natural, however, the application of the CQT for spectral modeling purposes is beyond the scope of this thesis.

### 3.1 The Phase Vocoder

The first form of the phase vocoder was proposed in [Fla66] and has been translated into its modern FFT based form in [Por76]. In [Moo78] [Dol86] the phase vocoder was further investigated in a musical context. Today the phase vocoder is a standard tool in digital audio signal processing as a means of time- and pitch/frequency-scaling, respectively. The theoretical background of the standard phase vocoder approach is very well documented, e.g. in [AKZ<sup>+</sup>02], [DGBA00] and [KLB06], and thus will only be briefly described here.

The algorithm basically comprises three stages: analyses, transformation and synthesis. In the analyses stage, the STFT representation of the input signal is calculated according to (2.1). Since the hop size that is used to compute



the STFT is a crucial parameter in the phase vocoder, (2.1) can be altered to incorporate the analysis hop size  $R_a$  as follows:

$$X^{\text{DFT}}(v, k) = \sum_{m=0}^{N-1} w\left(\frac{m}{N}\right) x\left(m + vR_a - \frac{N}{2}\right) e^{-i\Omega_k m} \quad (3.1)$$

with the STFT frame number  $v$  and the bins normalized center frequency  $\Omega_k = \frac{2\pi k}{N}$  and  $k = 0, 1, \dots, N-1$ .

In order to obtain a time-scaled output signal, the STFT representation is modified in the transformation stage. In the standard phase vocoder approach time-scaling is achieved by altering the hop size, that is, the synthesis hop size  $R_s \neq R_a$  (unlike the approach proposed by Bonada in [Bon00] where the hop size is kept constant and STFT frames are repeated or omitted to obtain a time-scaled output signal). Thus the magnitudes of the STFT representation are kept unchanged so that

$$|Y^{\text{DFT}}(v, k)| = |X^{\text{DFT}}(v, k)| \quad (3.2)$$

with  $Y^{\text{DFT}}(v, k)$  being the modified STFT representation, while the phases have to be updated in order to be consistent with the synthesis hop size  $R_s$ . This phase update process is based on an estimation of a partials instantaneous frequency at bin  $k$ . In order to estimate the instantaneous frequency, the unwrapped phase difference at bin  $k$  between two consecutive STFT frames has to be determined by

$$\Delta\varphi_a(v, k) = [\varphi_a(v, k) - R_a\Omega_k - \varphi_a(v-1, k)]_{\pm\pi} \quad (3.3)$$

where  $\varphi_a(v, k) = \angle X^{\text{DFT}}(v, k)$  and  $[\cdot]_{\pm\pi}$  denotes the principle argument (between  $\pm\pi$ ). The estimated instantaneous frequency at frame  $v$  and bin  $k$  is given by

$$\hat{\omega}(v, k) = \Omega_k + \frac{\Delta\varphi_a(v, k)}{R_a} \quad (3.4)$$

The synthesis phases are set according to a *phase-propagation* formula, which, in its simplest form, is given by

$$\varphi_s(v, k) = \varphi_s(v - 1, k) + R_s \hat{\omega}(v, k) \quad (3.5)$$

where  $\varphi_s(v, k) = \angle Y^{\text{DFT}}(v, k)$ .

The time-scaled output signal is obtained by transforming the modified DFT frames back into the time domain using the inverse FFT and by overlap adding consecutive frames. To reduce artifacts, usually a window function is applied to the frames in the time-domain prior to the overlap-add process.

### 3.1.1 Phase Coherence

The phase update process described above (which is usually referred to as *phase unwrapping*), ensures that the phase values of overlapping DFT frames are consistent. Picturing a constant-frequency, constant-amplitude sinusoid that means, that the synthesized sinusoid in frame  $v$  picks up the phase of the synthesized sinusoid in frame  $v - 1$ , thus adding constructively in the region of overlap. This is what commonly is referred to as *horizontal phase coherence* [LD97]. If phase unwrapping is skipped or erroneous, the loss of horizontal phase coherence introduces amplitude and frequency modulations to the synthesized partials.

Since the window functions in (3.1) do not only overlap in time, but also in the frequency domain (bandpass filters), the phase values of the modified STFT representation also have to meet strong consistency conditions within one frame along the DFT bins (*vertical phase coherence*). Depending on the main lobe width of the window function that is used, a constant-frequency, constant-amplitude sinusoid causes several neighboring bins to have non-zero coefficients. The phases of these neighboring bins are identical if the window function is centered at the beginning of the frame and exhibit a  $\pm\pi$  alternation if the window function is centered at the STFT frame center, respectively (note that this is true only if the window function is periodic, that

is the continuous window function is sampled such that only one coefficient is zero). In order to obtain a valid STFT representation, these phase relations have to be retained during the phase update process. In general, the phase unwrapping process to ensure horizontal phase coherence destroys vertical phase coherence causing artifacts in the output signal that are perceived as 'phasiness' or added reverberation.

### 3.1.2 Phase Locking

A first attempt to force both horizontal and vertical phase coherence at the same time, was proposed in [Puc95]. In the original paper the proposed method is generally called *phase locking*, however, in order to discern this approach from approaches proposed later, today it is referred to as *loose phase-locking*. In [Puc95] a certain degree of vertical phase coherence is obtained by a simple post-processing step after the phase unwrapping. The idea is, to steer the phases of low magnitude bins towards the phase of a neighboring high magnitude bin. This is achieved by altering the phase of each complex coefficient  $Y(v, k)$  after phase unwrapping in the following way<sup>1</sup>:

$$\varphi_s(v, k) = \angle (Y(v, k) + Y(v, k - 1) + Y(v, k + 1)) \quad (3.6)$$

Applying (3.6) does not exactly establish vertical phase coherence since the phases of the bins around a peak are not forced to be exactly equal, however, the phases will be similar (hence the term *loose phase locking*). The advantage of this approach is its computational efficiency since the spectral peaks do not have to be identified (this makes it a very elegant approach). The downside is, that the reduction of artifacts in the time-scaled output signal is very signal-dependent and never dramatic [LD99a].

Although less elegant, due to the lack of quality improvement applying the concept of loose phase locking, it seems like analyzing the signal to a certain

---

<sup>1</sup>In the original paper the signs of this formula are different because the DFT windows were assumed to be centered at the frame center and thus neighboring bins are 180° apart

extent is inevitable. As a consequence, in [LD97] a method was proposed that introduces a stage where the spectral peaks in each frame are identified and considered to stem from a single sinusoidal component in the input signal. In this simple peak-picking stage a peak is defined as a bin whose magnitude is greater than the magnitudes of its four nearest neighbors. The spectrum is then divided into *regions of influence* whereas the region boundaries can either be set halfway between two peaks or at the bin with the lowest amplitude between two peaks. Since the spectral peaks are assumed to stem from sinusoidal components in the signal, the phase unwrapping can now be confined to the peak bins in each frame. The phases of all other bins in a peaks region of influence are then locked to the peak bins phase. In [LD99a] two different implementations of this *rigid phase locking* concept are proposed:

In the first implementation, that is referred to as *identity phase locking*, the phase differences between the peak bins and the bins in the corresponding regions of influence of the original STFT representation  $X(v, k)$  are applied to the modified STFT representation  $Y(v, k)$  after phase unwrapping, so that

$$\varphi_s(v, k) - \varphi_s(v, k_1) \stackrel{!}{=} \varphi_a(v, k) - \varphi_a(v, k_1) \quad (3.7)$$

where  $k_1$  denotes a peak bin and  $k$  covers all bins in its region of influence.

The second implementation proposed in [LD99a] is referred to as *identical phase locking* and applies two modifications to the identity phase locking technique. Firstly, (3.3) is modified to allow for the case, when a peak at bin  $k_0$  in frame  $v - 1$  migrates to bin  $k_1$  in frame  $v$ . Hence, the unwrapped phase difference for a peak bin  $\Delta\varphi_a(v, k_1)$  is given by

$$\Delta\varphi_a(v, k_1) = [\varphi_a(v, k_1) - R_a\Omega_k - \varphi_a(v - 1, k_0)]_{\pm\pi} \quad (3.8)$$

With the instantaneous frequency  $\hat{\omega}(v, k_1)$  at bin  $k_1$  given by (3.4) the phase-propagation formula (3.5) is now given by

$$\varphi_s(v, k_1) = \varphi_s(v - 1, k_0) + R_s\hat{\omega}(v, k_1) \quad (3.9)$$

Secondly, a slightly generalized form of (3.7) is used and now reads as

$$\varphi_s(v, k) = \varphi_s(v, k_1) + \beta [\varphi_a(v, k) - \varphi_a(v, k_1)] \quad (3.10)$$

where  $\beta$  is a scaling factor and  $\beta = 1$  performs identity phase locking. However, in [LD99a] it is stated, that empirical tests have shown that the audio quality of the output signal can be improved when  $1 < \beta < \frac{R_s}{R_a}$  is used, however, there is no mathematical explanation.

In terms of audio quality, the concept of rigid phase locking outperforms the loose phase locking concept at the cost of higher computational complexity. A high quality time-scaled output signal can be achieved for a variety of audio signals including polyphonic music.

### 3.1.3 Shape Invariance

The phase locking scheme outlined above aims at retaining vertical phase coherence for neighboring DFT bins. This is a local operation on spectral peaks and is applied independently to each sinusoidal component. Hence, inter-sinusoidal phase relations, e.g. phase relations between harmonics, are not retained by this method. If these inter-sinusoidal phase relations are altered, the waveform (shape) of the input signal is altered as well. However, in general the human ear is quite insensitive to phase relations, that is, that two waveforms that seem completely different can sound almost exactly the same. This is true for most music signals, however, if speech signals are considered the shape of the waveform is perceptually critical as the loss of inter-sinusoidal phase coherence affects the perception of the underlying glottal excitation pulses and leads to audible artifacts. In order to retain the phase relations between sinusoidal components for speech signals, shape invariant time-scaling methods has been proposed in [QM92] and more recently in [Roe10]. Since the scope of this thesis is to investigate the applicability of the CQT for time- and pitch-scale modifications for polyphonic music signals, however, the issue of shape invariance has not been considered throughout

this work.

### 3.1.4 Transient Processing

The two main groups of artifacts that remain in the output signal are *transient smearing* and *phasiness*. One major reason for both is the remaining lack of phase coherence of the modified STFT representation.

The obvious explanation for the loss of sharp attack transients is, that the described approaches do not take into account transients, since phase coherence is only maintained for spectral peaks, i.e. sinusoidal components. However, if phase coherence at an attack transient is lost, its energy is spread in time which causes the perception of smeared transients.

Several approaches to retain sharp attacks in the time-scaled output signal has been proposed in the past. An appealing way to deal with transients is to detect and separate them from the input signal prior to time-scaling [LI98]. The remaining transient part can then be time-scaled in time domain (time translation of transients). The main problem with this approach usually is the unreliability and/or the computational complexity of separation algorithms, however, improvements in that field have been made recently ([OMLR<sup>+</sup>08], [Fit10]).

Another approach to transient processing is to detect transient regions in the input signal and use a synthesis hop size unequal to the analysis hop size only for non-transient regions ([MB96], [Ham01], [Bon00], [DDS02]), thus leaving transients regions unchanged. It has been shown, that this significantly improves the characteristics of transients in the time-scaled output signal. In order to achieve the overall desired stretch-factor, however, the time-scale factor has to be higher in stationary regions of the signal, which can be a problem when the signal contains very dense transient patterns.

In [Röb03] an approach to transient preservation has been proposed where the time-scale factor is kept constant throughout the entire signal. The sharpness

of attack transients is retained by resetting the phase values to the original phases for frames where attack transients were detected near the center of the frame (in [Röb03] also a novel transient detection technique based on the frames *center of gravity* [Coh95] has been proposed). In frames prior to the detected transient, the amplitude and phase values of the preceding steady part are used to avoid pre-echoing, whereas the amplitude of the transient frame is multiplied with a constant factor to compensate for the loss of level.

### 3.1.5 Resolution Issues

DFT based phase vocoder implementations that maintain phase coherence by identifying spectral peaks as sinusoidal components (rigid phase locking) have proven to significantly improve the quality of the time-scaled output signal. Such methods mark a merging of phase vocoder techniques and sinusoidal modeling techniques - and thus are prone to the same errors. Since phase coherence, both horizontal and vertical, is only maintained for spectral peaks, it is crucial that all sinusoidal components in the audio signal are properly resolved. As discussed in section 2.3, the linear bin spacing of the DFT does not seem to be the best tool to capture the logarithmic spacing of the fundamental frequencies in music signals. While for low frequencies a high frequency resolution is needed to resolve sinusoidal components properly, for higher frequencies a better time resolution is desired to capture fast temporal changes. Oversampling in the frequency domain, i.e. zero-padding the frames, improves the ability to identify spectral peaks that otherwise would not be visible in the spectrum, however, a correct phase unwrapping requires the absence of interference between sinusoidal components. The use of zero-padding can not eliminate such interferences, thus the bandwidths of the bandpass-filters at lower frequencies actually have to be smaller than at higher frequencies. In [Bon00] this problem is addressed by applying three DFTs with different window sizes in parallel. The proposed window sizes are 93 ms, 46.5 ms and 35 ms, respectively, and the desired upper cutoff frequencies are 700Hz, 2400Hz and 2250Hz, whereas the cutoff frequencies are time

varying to prevent peaks to occur in two different bands simultaneously. It has been shown that this multiresolution approach improves the quality of the time-scaled output signal significantly. To avoid prominent partials in the transition bands, however, the exact cutoff frequencies of the filters need to be steadily adapted. This can be avoided when this *discrete* multiresolution approach based on the DFT is replaced with the CQT providing a somewhat *continuous* multiresolution.

## 3.2 A CQT Phase Vocoder

With the CQT toolbox proposed in chapter 2 at hand, providing an efficient implementation of the constant-Q transform and a reasonable quality reconstruction, it seems worthwhile to investigate its applicability to the phase vocoder technique for time-scaling audio signals. The reported benefits of the use of a multi-resolution DFT in [Bon00] suggest a further improvement of the quality of the time-scaled output signal when the three band DFT is replaced by the CQT. In the following, the applicability of the phase vocoder concepts discussed in section 3.1 to the CQT representation are investigated.

### 3.2.1 Applying the Concepts

In the previous sections the two main issues that determine the quality of the phase vocoder's time-scaled output signal have been identified as horizontal and vertical phase coherence. Horizontal phase coherence is maintained by the process referred to as *phase unwrapping* where the frequencies of the sinusoidal components present in the audio signal are estimated. Vertical phase coherence can (partly) be maintained by applying one of the two discussed rigid phase locking techniques. In the sequel the validity of this two techniques for CQT representations will be discussed.



**Horizontal Phase Coherence: Phase unwrapping**

In the absence of noise or other interfering signal components, the frequency of a constant-amplitude constant-frequency sinusoid  $x(n) = \sin(\frac{2\pi f_x}{f_s}n)$  can be precisely estimated. The only condition that has to be met, is, that the difference  $\Delta\tilde{\varphi}_a$  between the cumulated (unwrapped) phases of the  $x(n)$ , i.e.  $R_a \frac{2\pi f_x}{f_s}$ , and a bins center frequency, i.e.  $R_a \Omega_k$ , is smaller than  $\pi$ , that is

$$-\pi < \Delta\tilde{\varphi}_a < +\pi \quad (3.11)$$

where

$$\Delta\tilde{\varphi}_a = R_a \frac{2\pi f_x}{f_s} - R_a \Omega_k = R_a \frac{2\pi f_x}{f_s} - R_a \frac{2\pi f_k}{f_s} = 2\pi R_a \frac{\Delta f_{xk}}{f_s} \quad (3.12)$$

If condition (3.11) is not met, phase unwrapping is ambiguous and the use of (3.3) will result in an erroneous frequency estimation. In the DFT case, solving (3.11) for the absolute value of the frequency difference yields

$$|\Delta f_{xk}| = |\Delta b_{xk} \frac{f_s}{N}| < \frac{f_s}{2R_a} \quad (3.13)$$

where  $\Delta b_{xk}$  is the frequency deviation in DFT bins, assuming no frequency oversampling.

If phase unwrapping is only applied to the spectral peaks,  $|\Delta b_{xk}| \leq \frac{1}{2}$ , hence condition (3.11) simplifies to

$$\frac{R_a}{N} < 1 \quad (3.14)$$

meaning, that for all overlap factors greater than 0% horizontal phase coherence can be maintained for the DFT phase vocoder.

For the proposed CQT implementation, the overlap of successive atoms is not constant but at a minimum for the highest bin within the one octave kernel. Since the maximal possible frequency deviation between the bins center frequency and  $f_x$  reaches its maximum also for the highest bin in the

kernel, it suffices to check condition (3.11) for the highest bin only. Hence, for the CQT (3.13) reads as <sup>2</sup>

$$|\Delta f_{xk}| \approx f_K \left( 2^{\frac{1}{2B}} - 1 \right) < \frac{f_s}{2R_a} \quad (3.15)$$

where  $f_K$  is the center frequency of the highest-frequency bin in the kernel.

The atom hop size  $R_a$  (in chapter 2 this parameter was labeled  $H_{\text{ATOM}}$ ) is given by

$$R_a = hN_K \quad (3.16)$$

where  $h$  is the atom hop size relative to the length  $N_K$  of the shortest atom (highest-frequency bin). Substituting (2.12) in (3.16) and (3.15) yields

$$f_K \left( 2^{\frac{1}{2B}} - 1 \right) < (2^{1/B} - 1) \frac{f_K}{2hq} \quad (3.17)$$

$$hq < \frac{2^{\frac{1}{B}} - 1}{2 \left( 2^{\frac{1}{2B}} - 1 \right)} \quad (3.18)$$

and for arbitrary values for  $B$

$$hq < 1 \quad (3.19)$$

That is, assuming no frequency domain oversampling ( $q = 1$ ), for all overlap factors greater 0% horizontal phase coherence can be maintained also for the CQT phase vocoder.

---

<sup>2</sup>The approximation in (3.15) stems from the fact, that due to the logarithmic bin spacing and the slightly different bandwidths of neighboring bandpass filters, the frequency between bin  $k$  and bin  $k + 1$  where the magnitudes of both coefficients are equal is not exactly  $f_k \left( 2^{\frac{1}{2B}} - 1 \right)$ . The exact frequency position depends on  $B$  and the used window function, however, the this frequency deviation is negligible

### Vertical Phase Coherence: Phase Locking

The phase locking schemes to retain vertical phase coherence for sinusoidal components rely on predictable phase relations between neighboring bins. For the DFT case this relation is given by a  $\pm\pi$  phase alteration if the window is centered at the frame center. That is, an impulse in the frame center will exhibit a flat magnitude spectrum and a phase spectrum that alternates between  $\pi$  and 0. The reason for this can be seen in figure 3.1(a)-(b) where the first four harmonics of the complex basis function  $A_k(n) = e^{-i\Omega_k n}$  are depicted. At the frame center, the real part (i.e.  $\cos(i\Omega_k n)$ ) alternates between +1 and -1, whereas the imaginary part (i.e.  $-\sin(i\Omega_k n)$ ) is 0, what explains the  $180^\circ$  phase difference between neighboring bins for an impulse in the frame center.

The explanation why this  $\pm\pi$  alteration between neighboring bins not only occurs for an impulse, but also for a sinusoid when a frame centered window is used, is best approached in the frequency domain. In figure 3.2 the continuous magnitude and phase spectra of three adjacent windowed DFT bases functions  $w(n)e^{-i\omega_k n}$  with center frequencies  $\omega_{k-1}$ ,  $\omega_k$  and  $\omega_{k+1}$ , respectively, are sketched. Since the windows are (periodic) symmetric and centered within the DFT frame, it is easy to see that the group delay [Boa03] of each bandpass filter is constant and equal to  $N/2$ , where  $N$  is the DFT frame size. The group delay is defined as the negative derivative of the phase response with respect to frequency, whereas the frequency is defined on the interval  $\omega \in [0, 2\pi)$ . In the discrete-time discrete-frequency case of the DFT, the group delay  $\tau_d = -\Delta\phi(\omega)/\Delta\omega$ . Since  $\Delta\omega = 2\pi/N$ , a group delay of  $\tau_d = N/2$  corresponds to a phase slope  $\Delta\phi(\omega) = \angle A(\omega_{k-1}) - \angle A(\omega_k) = -\pi$  within the main lobe of the bandpass filters. In figure 3.2 it can be observed, that a sinusoidal component with frequency  $\omega_x$  (red line) 'samples' the main lobes of the three bandpass filters (bins) with exactly  $\pi$  phase difference between neighboring bins.

Usually it is preferred to obtain equal phases for all significant bins excited by a sinusoidal component. For the DFT representation, there are two im-

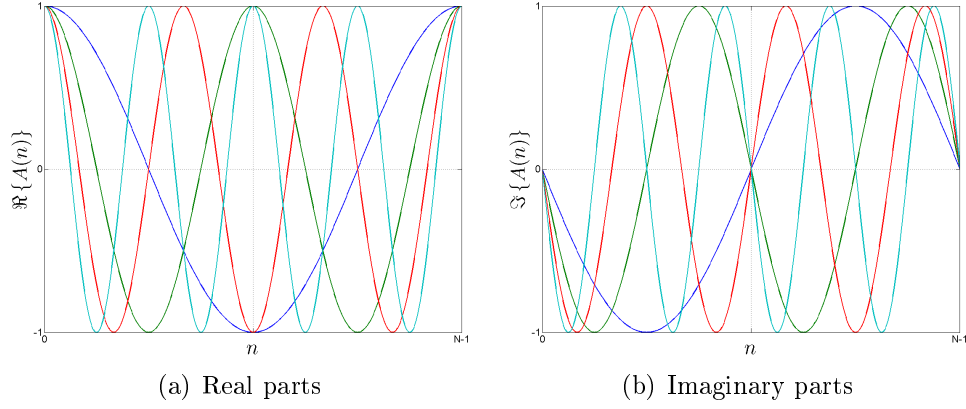


Figure 3.1: Real and imaginary parts of the first four harmonic basis functions within one DFT frame

plementations to achieve this: The first implementation is, to modify the DFT definition so that the point of zero phase for all basis functions is not at the beginning but at the center of the DFT frame. Thus, the modified STFT definition is given by

$$X^{\text{DFT}}(v, k) = \sum_{m=0}^{N-1} w\left(\frac{m}{N}\right) x\left(m + vR_a - \frac{N}{2}\right) e^{-i\Omega_k(m - \frac{N}{2})} \quad (3.20)$$

The real and imaginary parts of four of the phase shifted bases functions are depicted in figure 3.3(a)-(b). It can be observed, that the same result could have been achieved by simply inverting all odd harmonics, that is

$$X^{\text{DFT}}(v, k) = \sum_{m=0}^{N-1} w\left(\frac{m}{N}\right) x\left(m + vR_a - \frac{N}{2}\right) e^{-i\Omega_k m} (-1)^k \quad (3.21)$$

$$= (-1)^k \mathcal{F} \left\{ w\left(\frac{m}{N}\right) x\left(m + vR_a - \frac{N}{2}\right) \right\} \quad (3.22)$$

where the factor  $(-1)^k$  implements a circular shift by  $N/2$  samples in the frequency domain.<sup>3</sup>

<sup>3</sup>This is because a circular time shift by  $N_0$  samples can be implemented in the frequency domain by applying a phase shift of  $e^{i\frac{2\pi}{N}kN_0}$ . If the time shift  $N_0 = N/2$  the factor simplifies to  $e^{i\frac{2\pi}{N}k\frac{N}{2}} = e^{i\pi k} = \cos(\pi k) = (-1)^k$ .

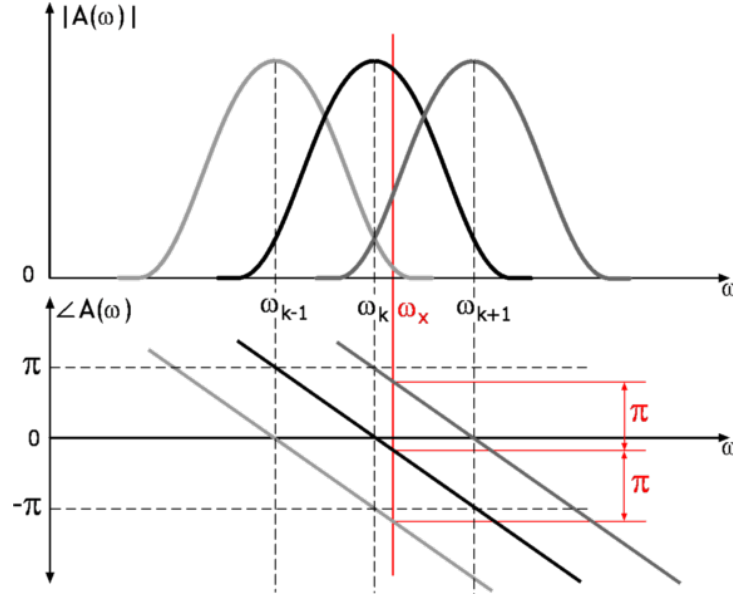


Figure 3.2: Continuous magnitude and phase spectra of three adjacent windowed DFT bases functions.

The second implementation to obtain the same result, is to apply this circular time shift not in the frequency domain but in the time domain. That is, a circular shift by  $N/2$  samples has to be applied to the windowed input signal what merely means to swap the first and the second half of the frame<sup>4</sup>. After applying this time shift, either in the frequency or in the time domain, the phase spectrum of the bandpass filters (bins) is zero (zero-phase filters) and thus neighboring bins excited by a single sinusoidal component will exhibit equal phases.

In the CQT definition given in chapter 2, the starting phases of the complex exponential in (2.6) are bounded to the beginning of the windows (just like in the standard STFT definition). In figure 3.4(a)-(b) the real and imaginary parts of four (unnormalized) CQT atoms are depicted. As discussed above, the bounding of the starting phase of the complex exponentials to the beginning of the window, causes a  $180^\circ$  phase jump between neighboring bins in the DFT representation. In the proposed CQT, atoms are centered

<sup>4</sup>In Matlab, for example, this operation is implemented in the function *fftshift*.

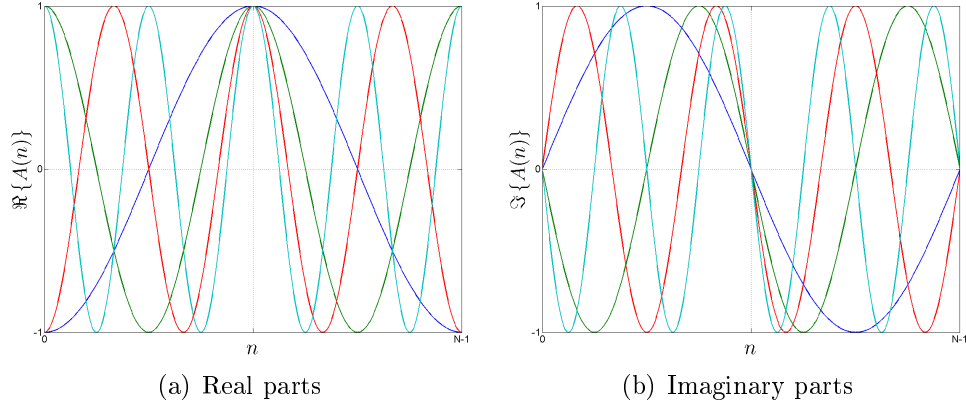


Figure 3.3: Real and imaginary parts of the first four harmonic basis functions within one DFT frame (zero-phase)

at the same position within the frame (*stacked atoms*) and the window size decreases as the center frequency increases. Hence, the phases of all atoms are equal at the atom center, that is, the phase difference between neighboring bin is exactly zero. However, this is only the case if all windows are exactly symmetric about the same point and  $N_k \propto \frac{1}{f_k}$ . For practical reasons, the window sizes  $N_k$  are rounded towards the nearest integer and standard window implementations are used in the first version of the toolbox proposed in [SK10], thus violating both conditions. In figure 3.5(a)-(b) it can be observed, that it is not possible to define a common center point for both even and odd length windows with standard window function implementations. Hence, the group delays of neighboring bins potentially differ by half a sample. That is, the phase slopes of neighboring bins are no longer parallel (unlike for the DFT case depicted in figure 3.2) and the phase differences between bins considering a constant-frequency sinusoidal input signal with frequency  $\omega_x$  are dependent on  $\omega_x$ . One way to ensure equal group delays for all bins would be to allow only even (or odd)  $N_k$ . This, however, will change the phase value at the window center, which is given by

$$\Phi_c = \frac{2\pi f_k}{f_s} \cdot \frac{N_k}{2} = \frac{\pi f_k N_k}{f_s} = \frac{q\pi}{2^{1/B} - 1} \quad (3.23)$$

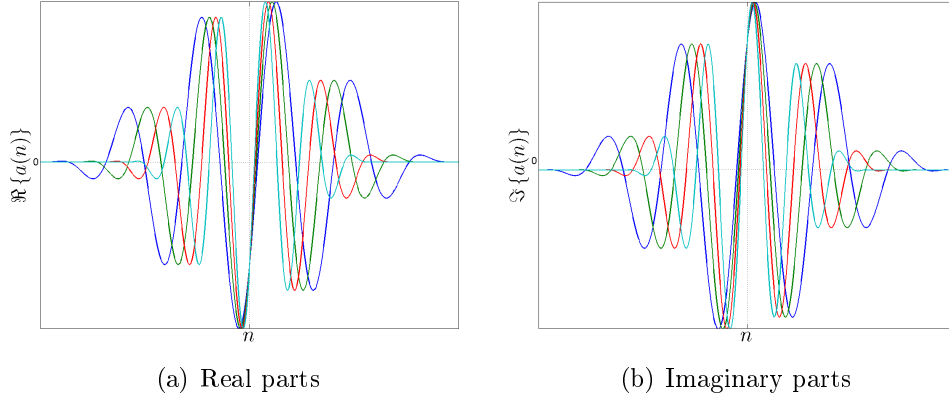


Figure 3.4: Real and imaginary parts of four CQT atoms

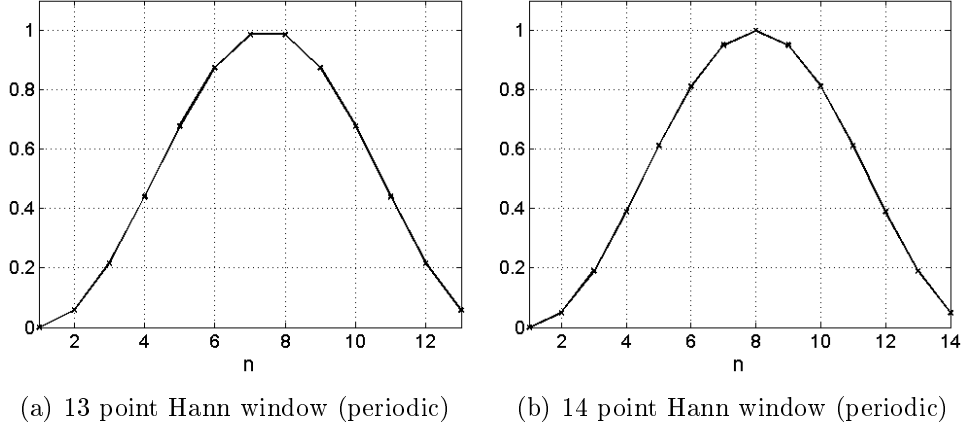


Figure 3.5: Standard implementation of the Hann window

Rounding  $N_k$  to the nearest even (or odd) integer  $\hat{N}_k$  adds a phase error term yielding the altered center phase

$$\hat{\Phi}_c = \Phi_c + \underbrace{\frac{\pi f_k}{f_s} \left( \frac{\hat{N}_k - N_k}{2} \right)}_{\Phi_e} \quad (3.24)$$

with the maximum value of the error term  $\Phi_{err}$  given by

$$|\Phi_e^{\max}(f_k)| = \frac{\pi f_k}{2f_s} \quad (3.25)$$

To avoid this phase error, modified window implementations which allow fractional window sizes are required. In figure 3.6(a)-(b) two modified Hann windows with fractional window sizes are depicted. In order to be able to define a common center point for different window sizes, the sample grid is shifted to capture the center of the window regardless of its length. Note that the number of samples returned by the modified Hann window implementation is always odd as the window function is sampled at its center and at  $m \in \mathbb{N}$  positions both right and left of the center.

Using modified window functions yields equal phases of neighboring bins, however, if zero phase bandpass filters are desired (so that an impulse in the center of an atom stack is zero phase), the phases of the complex exponential in the CQT definition have to be modified, yielding a CQT definition that reads

$$X^{\text{CQ}}(k, n) = \sum_{m=n-\lfloor N_k/2 \rfloor}^{n+\lfloor N_k/2 \rfloor} x(m) a_k^*(m - n + N_k/2) \quad (3.26)$$

with its basis functions  $a_k(n)$  defined by

$$a_k(n) = \frac{1}{\sum_n w(n/N_k)} w\left(\frac{n}{N_k}\right) \exp\left[i2\pi\left(n - \frac{N_k}{2}\right) \frac{f_k}{f_s}\right], \quad (3.27)$$

In figure 3.7(a)-(b) the real and imaginary parts of four zero-phase atoms obtained by this altered CQT definition are depicted.

### A note on the quality of reconstruction

In section 2.11 it was shown that the quality of the reconstructed signal from its CQT coefficients can be maximized when a Blackman-Harris window is used. Hence, instead using a modified Hann window the obvious thing to do would be to use modified Blackman-Harris windows that allow for fractional window sizes and guarantee center sampling. However, the Hann window is preferred for phase vocoder applications since its main lobe width



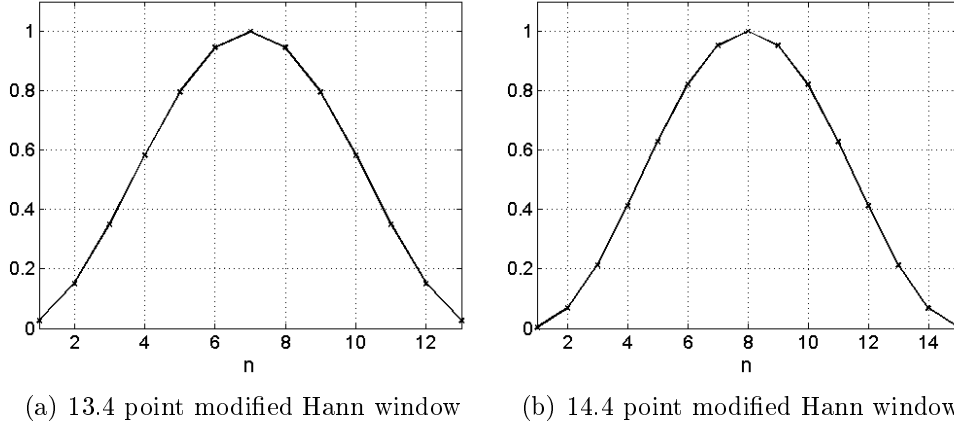


Figure 3.6: Modified Hann windows allow for fractional window sizes and always sample the center of the window.

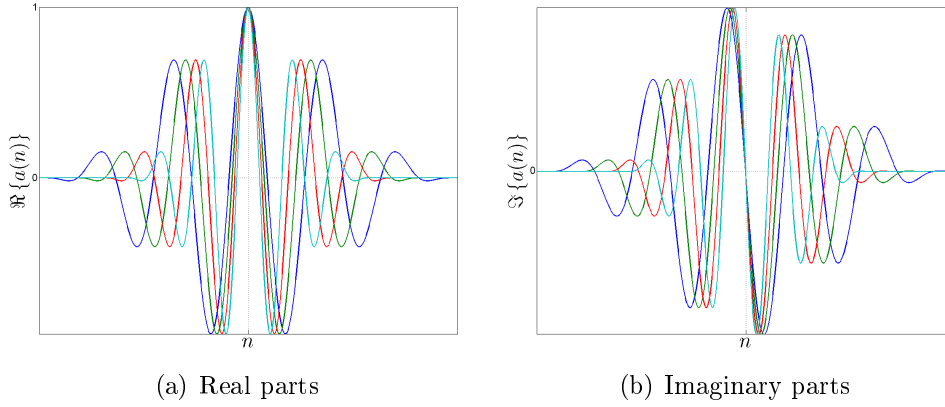


Figure 3.7: Real and imaginary parts of four zero-phase CQT atoms

is smaller than the main lobe width of a Blackman-Harris window and thus introduces less interference when closely spaced sinusoids need to be resolved. Fortunately it was found that if a squared modified Hann window is used<sup>5</sup> the reconstruction quality is similar or even higher compared to Blackman-Harris windows. In table 3.1 the quality of reconstruction for the Blackman-Harris window and the squared modified Hann window is compared.

---

<sup>5</sup>A squared modified Hann window since the window function is applied when the CQT is performed and again at the inverse CQT. For this reason, when a Blackman-Harris window is used, it's square root is applied.

R	B	Blackman-Harris window	squared mod. Hann window
4	24	39.0 dB SNR	44.9 dB SNR
4	48	40.5 dB SNR	47.2 dB SNR
5	24	55.8 dB SNR	53.0 dB SNR
5	48	57.8 dB SNR	54.5 dB SNR
6	24	56.9 dB SNR	57.6 dB SNR
6	48	58.6 dB SNR	57.9 dB SNR

Table 3.1: Quality of reconstruction for different settings and window functions. R is the redundancy factor (see section 2.9) and B is the CQT resolution (bins per octave). The test signal was white noise, what marks a worst case scenario. For real world music signals the achieved quality usually is a few dB higher.

### Transient Processing with the CQT Phase Vocoder

The problem of transient smearing discussed for the DFT phase vocoder is an issue also for the CQT phase vocoder. However, due to the improved temporal resolution at high frequencies (short window lengths), attack transients at higher frequencies do not require further processing since the onset softening is negligible. This is also true for percussive transients that exhibit most of their energy in the upper frequency regions (e.g. Hi-Hat, cymbals, etc.). Transient smearing for attack transients and percussive transients at low frequencies (e.g. bass notes, kick drum, etc.), where quite long windows are used, however, remains an issue.

To prevent transient smearing at lower frequencies, similar approaches as discussed in section 3.1.4 for the DFT phase vocoder could be used, however, not all of them are feasible for the CQT phase vocoder. In figure 3.8 the CQT representation of an impulse (Kronecker delta) is depicted. Due to the increasing window lengths towards low frequencies, the representation of the impulse gets widened. Trying to prevent transient smearing by using a local speed constraint, i.e. setting the scaling factor to 1 for transient regions, is not reasonable for the CQT phase vocoder since the number of time frames affected by a transient event depends on the frequency. Hence, a transient region cannot be isolated in time in the CQT representation. Additionally, a

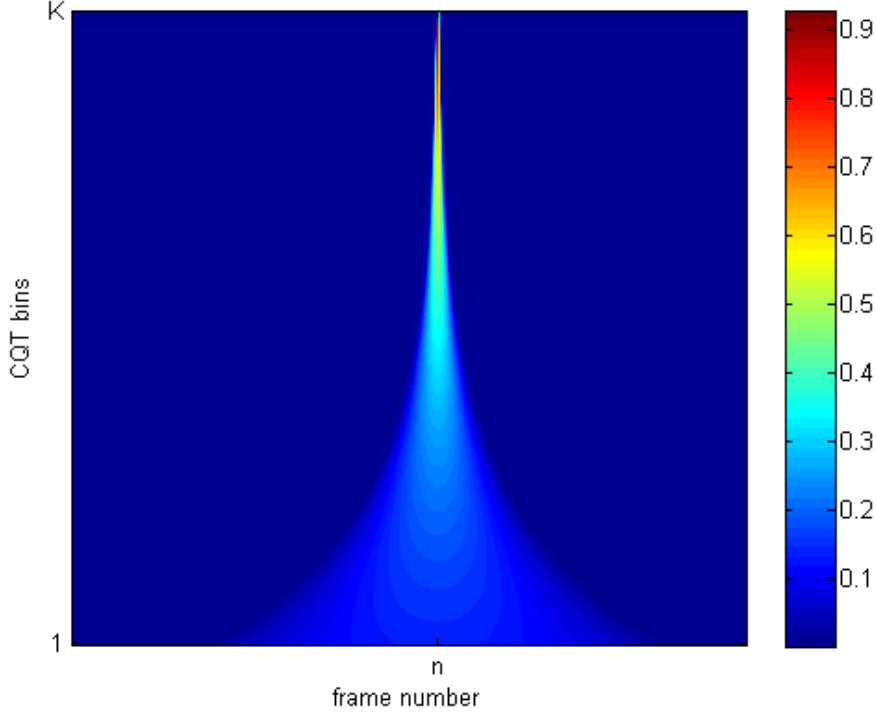


Figure 3.8: Kronecker delta: Magnitude values of the rasterized CQT representation

local speed constraint cannot be set in the sparse representation of the CQT but only in the less efficient full matrix (rasterized) representation.

An approach that obviously works for both the DFT and CQT phase vocoder, is to separate transients from the input signal prior to time-scaling. As mentioned in section 3.1.4, this method solely relies on the ability of the applied transient separation algorithm to subtract all transients from the input signal without affecting sinusoidal components. This approach has not been implemented for CQT phase vocoder, however, due to steadily improving separation algorithms, this could be an interesting topic for future research.

Following the notion of R  bel [R  b03], sharp attacks can also be retained by only adjusting the coefficients phases and amplitudes. In the sequel, the required phase/amplitude adjustments for transients in the CQT represen-

tation will be discussed by means of a single impulse input signal.

When the hop size of the CQT representation is altered, the main issue is to maintain phase coherence. For sinusoidal components, the horizontal phase progression needs to be adjusted so that no phase jumps between consecutive frames occur and vertical phases had to be locked to retain the spectral shape of the sinusoid to mitigate artifacts commonly referred to as 'phasiness'. For an impulse, however, vertical phase coherence is essential to keep the temporal shape of the signal within one frame. If vertical phase coherence is lost, a noise-like event will be generated during inverse transform rather than an impulse. Horizontal phase coherence, on the other hand, is essential to ensure that the impulses generated by successive frames add up at the same point in time. If horizontal phase coherence is lost, several closely spaced impulses will be generated causing the sensation of multiple attacks. The phase update process for an impulse thus comprises two stages:

- **Horizontal phase unwrapping:** An impulse is a broadband event, hence the phases need to be unwrapped not only for spectral peaks, but for the entire representation (or region around the impulse). In figure 3.9(a) the horizontal phase differences (phase differences between consecutive atoms for each bin) of the original input signal are depicted. Figure 3.9(b) shows the horizontal phase differences after phase unwrapping.
- **Vertical phase realignment:** For the case of sinusoidal components, vertical phase coherence can be retained by applying the phase differences from the original signal around a spectral peak to the modified representation (phase locking). This method can not be applied to impulses (or transients in general). In figure 3.10(a)-(b) the phase differences between adjacent bins within one time-slice (*vertical phase differences*) are depicted prior to, and after horizontal phase unwrapping, respectively. Since the phases just before the impulse depend on the entire signal content prior to the impulse, horizontal phase unwrapping causes a *scrambling* of the vertical phase differences. The idea of phase

realignment is to rotate all phase values so that frame  $M$ , for which the impulse is close to its center, is equal to frame  $M$  of the input signal. Thus, the realigned CQT representation  $Y(k, n)$  is given by

$$Y(k, n) = \tilde{X}(k, n)e^{\Phi_r(k)} \quad (3.28)$$

with CQT coefficients after phase unwrapping  $\tilde{X}(k, n)$  and

$$\Phi_r(k) = \angle X(k, M) - \angle \tilde{X}(k, M) \quad (3.29)$$

where  $X(k, M)$  is the CQT representation of the original input signal and  $M$  is the frame number where the impulse is close to the center of the frame. Note that this notation is only valid for the rasterized CQT representation, however, the concept can also be applied to the sparse representation.

The vertical phase differences thus realigned are depicted in figure 3.11. The same result can be obtained when the phases are unwrapped in two different directions. That is, starting at frame  $M$ , the phases are unwrapped to the left and to the right, hence, the phase realignment stage can be omitted.

Contrary to the case when steady sinusoidal components are considered, transients cannot be fully recovered with mere phase modifications. For example, consider an impulse that is centered at frame  $N$ : when the synthesis hop size is greater than the analysis hop size, the magnitude decay from frame  $N$  to frame  $N + 1$  should increase, and vice versa. When the phases are adjusted properly, the erroneous amplitude decay between frames only causes an amplification or attenuation of the impulse, respectively, that is, no artifacts are introduced and the original impulse can be recovered by rescaling the amplitudes. However, one error that actually changes the transient characteristic remains: not all atoms that captured the impulse during analysis still capture the impulse when the synthesis hop size is greater than the analysis

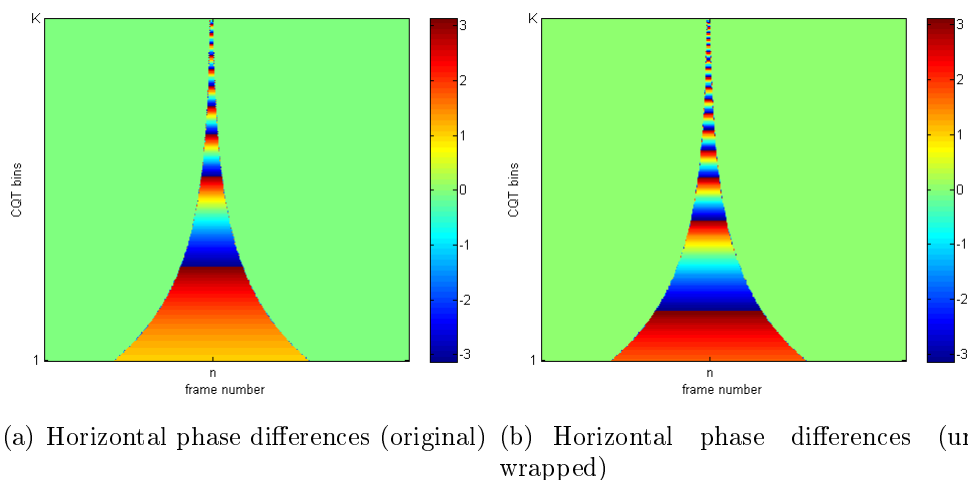


Figure 3.9: Kronecker delta: horizontal phase differences of the rasterized CQT representation.

hop size. In these cases the transient energy can no longer be shifted to the correct point in time but rather will be shifted in the wrong direction due to circular shifting. In figure 3.11 this effect can be observed causing a 'phase jump' from  $\pi$  to  $-\pi$  and from  $-\pi$  to  $\pi$ , respectively, in horizontal direction. This effect is mitigated by the fact that the amplitude values in these regions are quite small, however, for increasing scaling factors artifacts stemming from this error are no longer negligible. One possible solution to that problem would be to discard the coefficients in this *circular shifting region* and use amplitude and phase values from coefficients outside the transient region instead (similar to the approach proposed in [Röb03]).

To demonstrate that the phase realignment method does not only work for the theoretical case of a Kronecker delta but also for real world transient signals, a kick drum (the original waveform is depicted in figure 3.12(a)) is time-scaled applying a time-scaling factor of 1.7. Using a CQT resolution of 24 bins per octave, the phases for the entire signal are unwrapped, realigned and the amplitudes are rescaled. The time-scaled output signals with and without vertical phase realignment are depicted in figure 3.12(b) and 3.12(c), respectively. It can be observed that the shape of the waveform is heavily

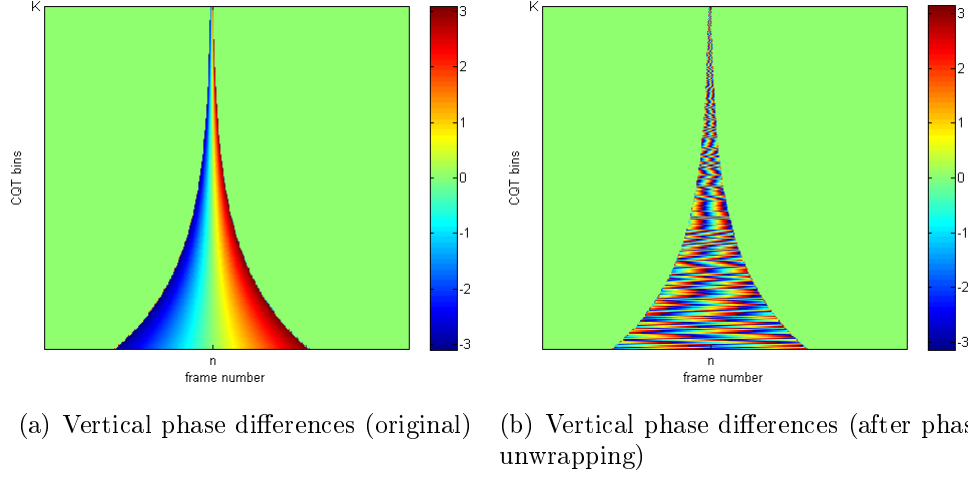


Figure 3.10: Kronecker delta: vertical phase differences of the rasterized CQT representation.

degraded when no phase realignment is applied. In 3.12(c) it can be observed that amplitude errors occur prior to and after the actual transient due to 'circular shifting' frames. The overall shape of the waveform, however, is preserved very well when phase realignment is applied. The corresponding audio examples are given in sample 1.

original | phase unwrapped | phases realigned |

---

**Sample 1:** kick drum (original and time-scaled by factor 1.7)

In order to incorporate the phase realignment method for transients in the CQT phase vocoder, the exact positions of transients need to be detected and the regions around these positions where phase realignment is applied have to be defined. This non-trivial stage has not been implemented in the the CQT phase vocoder in the course of this thesis and remains a task for future work.

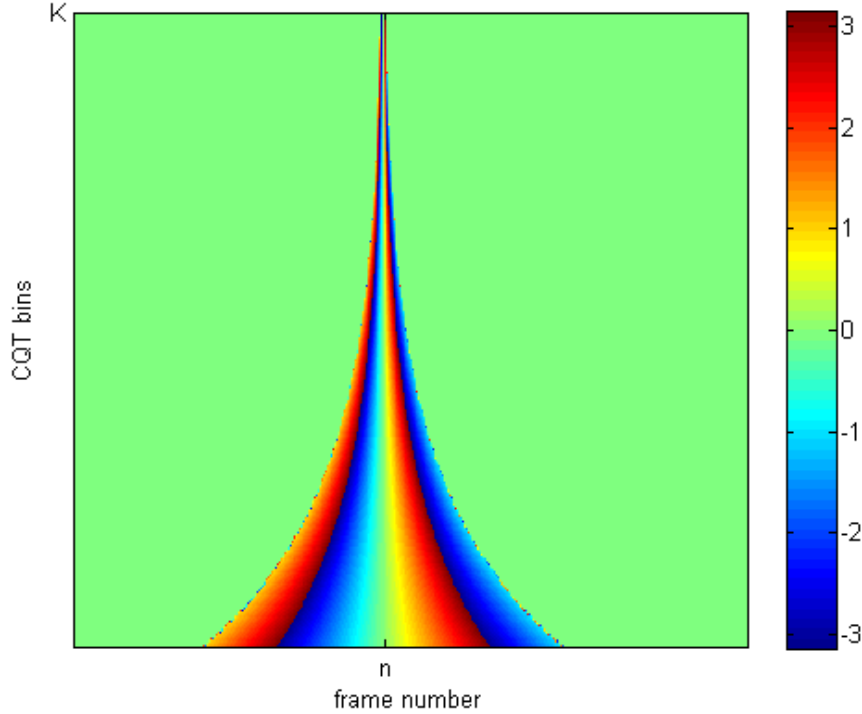


Figure 3.11: Kronecker delta: vertical phase differences of the rasterized CQT representation after phase realignment.

### Practical Considerations

A final practical consideration when applying the CQT to the phase vocoder concept, is the decreasing number of time-frequency sampling points for lower octaves. If a spectral peak is detected at the highest frequency bin in octave  $P-1$ , the phase of the lowest frequency bin of octave  $P$  needs to be locked to the peak's phase. However, since octave  $P$  has two times more coefficients than octave  $P-1$ , the missing coefficients in octave  $P-1$  have to be computed. To solve this problem, all octaves except for the highest can be time-domain oversampled by factor 2 by applying a second, time shifted kernel. The time-frequency sampling grid thus obtained is depicted in figure 3.13.

Additionally, when the kernel is populated by more than one atom per center frequency, a different kernel with altered window hop sizes has to be used in



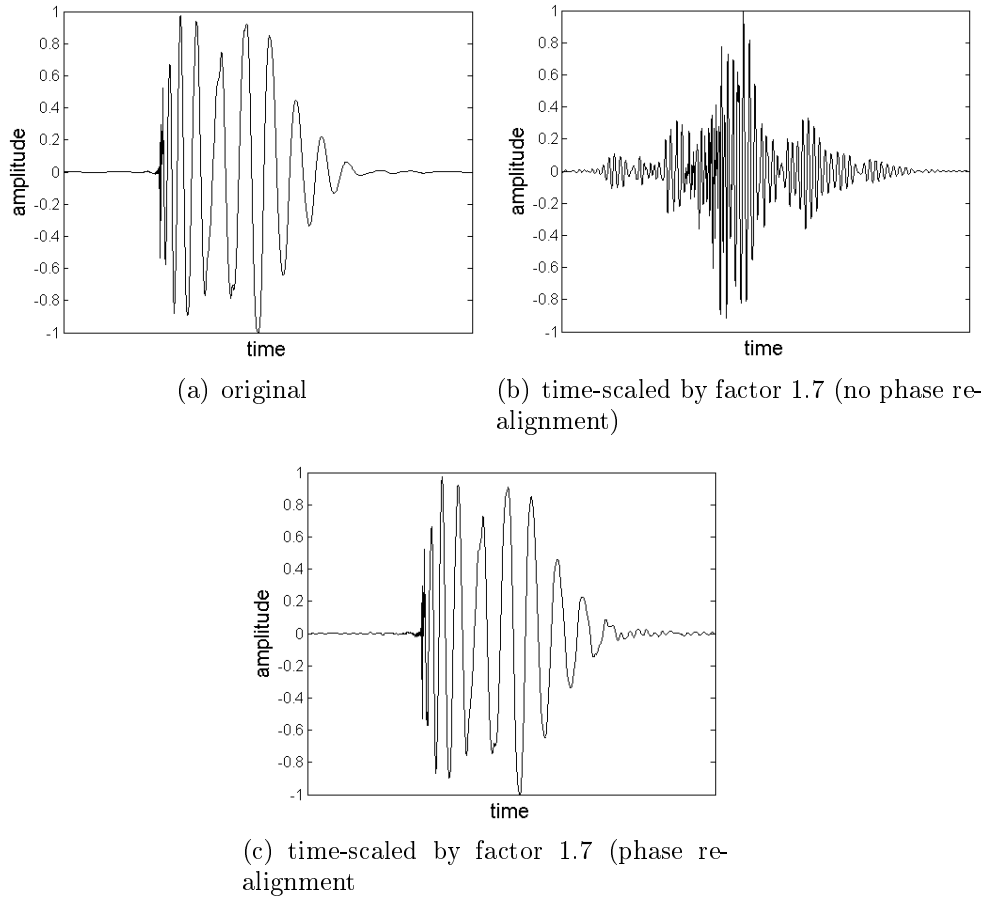


Figure 3.12: Waveforms of the original and the time-scaled versions of a kick drum sound.

the inverse CQT.

### 3.2.2 CQT vs. DFT Phase Vocoder

In the sections above, the theoretical considerations for the implementation of a CQT phase vocoder were outlined. In the following, the time-scaled output signals produced by a phase-locked phase vocoder based on the DFT and the CQT, respectively, are presented. The differences between the two approaches are discussed by means of analytical signals and real world music signals.

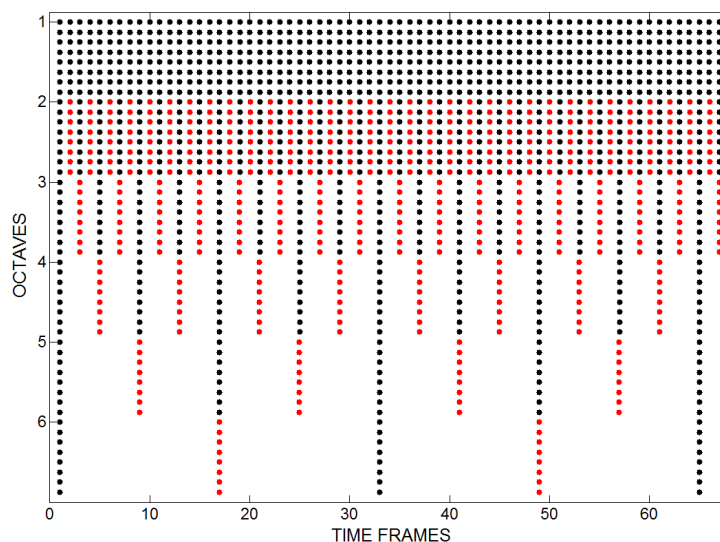


Figure 3.13: Increased density sample grid of the CQT representation by factor 2. The black dots correspond to the original sampling grid of the time-frequency plane, the red dots correspond to the additional sample points.

### Close Sinusoids

The expected benefits of using the logarithmic bin spacing provided by the CQT are the improved frequency resolution for low frequencies and the improved time resolution for high frequencies. To demonstrate the former, an input signal is composed of two pure sinusoids with respective frequencies of 98 Hz (G2) and 130.8 Hz (C3), that is, two low frequency sinusoids placed a perfect fourth apart from each other. In figure 3.14(a)-(b) the spectra of the time-scaled output signals for the two approaches are depicted, applying a time-scaling factor of 1.3. From figure 3.14(a) it can be appreciated, that the DFT based phase vocoder introduces artifacts stemming from the inadequate frequency resolution for the given input signal. In figure 3.14(b) it can be observed, that no artifacts are produced by the CQT phase vocoder.

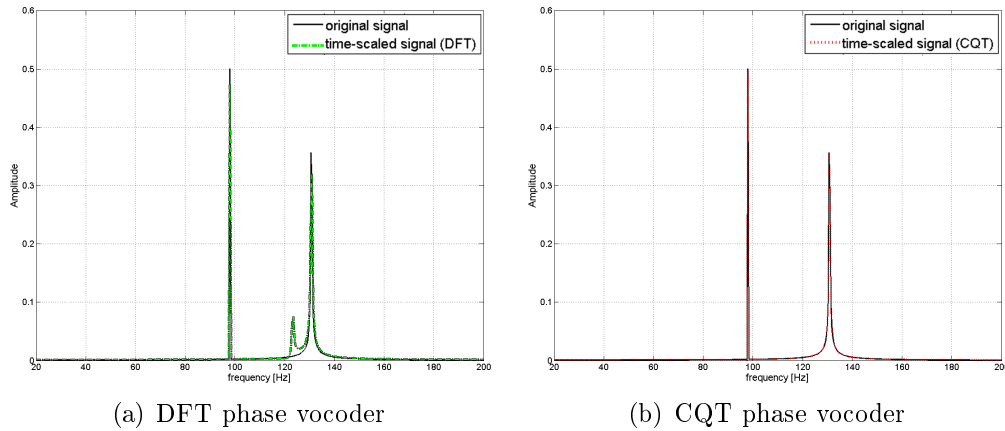


Figure 3.14: Spectra of time-scaled output signals stretched by factor 1.3. The input signal is a composition of two constant-frequency constant-amplitude sinusoids at 98 Hz (G2) and 130.8 Hz (C3), respectively. DFT phase vocoder: frame size = 4096, hop size = 512, FFT size = 4096; CQT phase vocoder: 24 bins/octave,  $q = 1$ , atom hop factor =  $1/8$

### Sinusoidal Bursts

The second exemplary signal is a short sinusoidal burst at 10 kHz with a duration of 10ms. In figure 3.15 the time scaled output signals as well as the ideal reference signal are depicted. It can be observed, that due to the lower time resolution for high frequencies, the output signal produced by the DFT phase vocoder is heavily degraded, whereas the CQT phase vocoder is able to maintain the general shape of the input signal quite well.

Note, that these two test signals are tailored to demonstrate the benefits of a constant Q frequency resolution. If the test signals would be a short low-frequency burst and two closely spaced high frequency sinusoids, respectively, the results would favor the DFT phase vocoder. However, it is assumed that in musical signals quick temporal changes mainly occur at higher frequencies whereas closely spaced sinusoidal components will occur at low frequencies due to the logarithmic frequency spacing of fundamental frequencies.

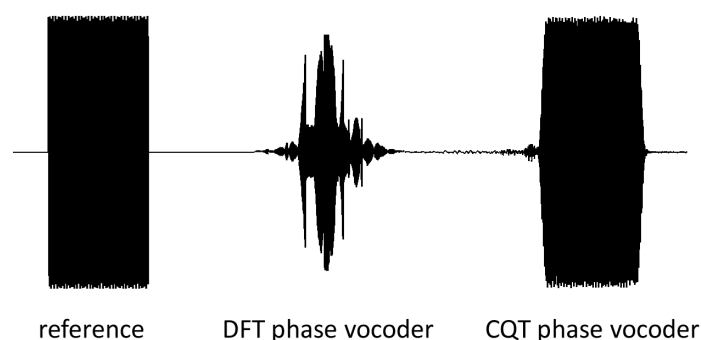




Figure 3.15: Time scaled output signals of a 10kHz sinusoidal burst (10ms) stretched by factor 1.7. DFT phase vocoder: frame size = 4096, hop size = 512, FFT size = 4096; CQT phase vocoder: 24 bins/octave,  $q = 1$ , atom hop factor =  $1/8$








### Music Signals

Having demonstrated the benefits of the CQT phase vocoder compared to the DFT based implementation for some analytical signals, the time-scaled output signals for several exemplary music signals are presented in the following. For the DFT based phase vocoder, the implementation provided by the Audio Research Group at the Dublin Institute of Technology (<http://www.audioresearchgroup.com/>) has been used. Both the CQT and the DFT based implementations use variable analysis hop sizes and apply phase unwrapping for spectral peaks and identity phase locking. The following examples were produced using a DFT frame size of 4096 samples and a CQT resolution of 48 bins per octave, respectively, and the applied stretch factor is 2.1.

	original   DFT phase vocoder   CQT phase vocoder
	<b>Sample 2:</b> strings, ride cymbal, electric piano

	original   DFT phase vocoder   CQT phase vocoder
	<b>Sample 3:</b> brass, reeds, harp

	original   DFT phase vocoder   CQT phase vocoder <b>Sample 4:</b> strings, bass, flugelhorn, piano
	original   DFT phase vocoder   CQT phase vocoder <b>Sample 5:</b> acoustic guitar
	original   DFT phase vocoder   CQT phase vocoder <b>Sample 6:</b> electric piano, percussion
	original   DFT phase vocoder   CQT phase vocoder <b>Sample 7:</b> drums, bass, guitar, vocals
	original   DFT phase vocoder   CQT phase vocoder <b>Sample 8:</b> ride cymbal, snare drum, upright bass, piano, trumpet
	original   DFT phase vocoder   CQT phase vocoder <b>Sample 9:</b> electric guitar, bass, organ, tambourine
	original   DFT phase vocoder   CQT phase vocoder <b>Sample 10:</b> female singer

**Subjective quality impressions:** In samples 2-4 (dense signals), the CQT phase vocoder seems to introduce less *roughness*<sup>6</sup> than the DFT phase vocoder. The ride cymbal in sample 2 sounds more natural using the CQT phase vocoder and the harp's attack transients in sample 3 seem to be better retained. In sample 4 the loss of sharp attacks at low frequencies using the

<sup>6</sup>According to [ZF99] *roughness* is the psychoacoustical sensation that occurs when a signal (or part of a signal) is amplitude-modulated by a modulation function with a spectrum between 15 and 300 Hz.

CQT phase vocoder can be recognized for the upright bass. In sample 5-6 attack transients for the acoustic guitar (sample 5) as well as for the hi-hat in sample 6 seem to be better preserved using the CQT phase vocoder. The need for transient processing at lower frequency in the CQT phase vocoder is obvious in sample 7 where the attacks of the kick drum and the bass are lost. The same is true for the upright bass in sample 8, however, less artifacts are introduced for the ride cymbal and the piano compared to the DFT phase vocoder<sup>7</sup>.

### 3.3 Conclusion

In this chapter the phase vocoder principle for time-scale modifications along with the basic concept of horizontal and vertical phase coherence has been outlined. Phase vocoder approaches are capable of producing a high quality output signal for both monophonic and polyphonic music input signals, however, the DFT based phase vocoder suffers from its rigid time/frequency resolution trade-off. It has been shown that for sinusoidal components and attack transients in higher frequencies artifacts that stem from these resolution issues can be significantly reduced with a CQT based phase vocoder approach. In order to implement a CQT phase vocoder the CQT definition given in chapter 2 has been modified slightly and a modified Hann window was introduced. The use of this window function in the CQT implementation eliminates phase errors that stem from rounding the window sizes and guarantee constant group delays for each atom stack. The application of modified Hann windows improves the ability to resolve partials compared to Blackman-Harris windows while not impairing the quality of signal reconstruction from its CQT coefficients.

While improving the quality of the time-scaled output signal for a wide range

---

<sup>7</sup>Note that this comparison between the STFT and the CQT based phase vocoder considering the perceived quality of the time-scaled output signals is solely based on the author's impressions. Neither objective listening tests nor psychoacoustical measurements were conducted in order to objectify these impressions.

of input signals, it has been demonstrated that due to very long windows for lower frequencies, low-pitched transients and attack transients can not be reproduced by the CQT phase vocoder without further processing. Addressing this issue, the principles of a phase realignment approach has been proposed, however, the incorporation of this approach into the CQT phase vocoder is an open issue. Another way of dealing with the problem of low frequency transients is to separate those parts prior to time-scaling, e.g. utilizing the separation algorithm proposed in [Fit10] and apply time-domain time-scaling for the transient part (the CQT implementation of this harmonic/percussion separation approach will be outlined in section 4.1.3). As a strict constant-Q resolution is not crucial for the phase vocoder, problems due to very long windows at low frequencies could also be partly solved by limiting the window size in the first place. This could be seen as a hybrid DFT/CQT phase vocoder, an approach that is yet to be implemented.

In general it can be appreciated that the CQT phase vocoder significantly improves the quality of the time-scaled output signal compared to the DFT based implementation. However, apart from a strictly constant Q resolution also other multi-resolution approaches (e.g. a multi-resolution DFT as proposed in [Bon00]) were shown to improve the performance of the phase vocoder. An application where a constant-Q resolution, however, outperforms both the single- and the multiresolution DFT concerning the ease of implementation is outlined in the next chapter.





## Chapter 4

# Pitch Transposition in the CQT Domain

Scaling the pitch of an audio signal is a dual operation to time-scaling. Hence, a very common approach to pitch-scaling is to time-scale the input signal, followed by a re-sampling stage retaining the signals original duration while having changed its frequency content. The order of the re-sampling stage and the time-scaling stage is interchangeable and thus can be chosen differently for up- and downscaling in order to minimize computational complexity. In [LD99b], however, a STFT-based approach was proposed to modify the frequency of sinusoidal components directly in the frequency domain without applying time-scaling. The idea is to identify sinusoidal components as peaks in the short time spectrum and shifting the region around this peak by a frequency  $\Delta\omega$ , applying a simple copy and paste operation. To retain horizontal phase coherence between consecutive frames, a very simple phase update step is outlined in [LD99b]: Given the frequency shift  $\Delta\omega$ , it suffices to rotate the phases by  $\Delta\omega R_a$ , where  $R_a$  is the STFT hop size, that is, there is no need to determine the exact frequency of the shifted sinusoidal component (no phase unwrapping). If the frequency shift does not correspond to an integer multiple of the DFT bin spacing (as is usually the case), frequency-domain interpolation is used. This simple implementation of the phase update pro-

cess, however, assumes, that the frequency shift  $\Delta\omega$  is independent of the true frequency of the sinusoidal component to be shifted, which is rarely the case since usually harmonic shifts are desired (i.e. a shift by factor  $\alpha$  rather than a linear frequency shift by a constant frequency  $\Delta\omega$ ). Nevertheless, the advantage of this method is, that single components of the signal can be shifted in the frequency domain, for example to change only certain notes in the audio signal while leaving others untouched. However, since each sinusoidal component needs to be shifted by a different frequency to retain the harmonic structure and frequency-domain interpolation is needed almost certainly, this method is limited to more exotic effects but can hardly be applied to polyphonic pitch shifting in general.

If the DFT in this approach, however, is replaced with the CQT, the problem of shifting each sinusoidal component by a different (usually fractional) number of DFT bins is circumvented, since frequency shifts by factor  $\alpha$  correspond to a constant shift of  $r$  CQT-bins with  $\alpha = 2^{r/B}$ , where  $r$  is independent of the frequency of the sinusoidal components. Furthermore, for the case of chromatic pitch transpositions (or transpositions by a fraction of a semitone, e.g.  $\frac{1}{8}$ -tones for a CQT with 48 bins per octave),  $r$  is an integer and no coefficient interpolation is needed. In the sequel chromatic pitch transpositions will be discussed, however, using simple interpolation arbitrary pitch-scaling factors can be implemented.

## 4.1 Transposing the Entire Representation

Pitch transpositions in the CQT domain provide the possibility to shift the frequency content of certain time- or frequency segments. That is, the entire signal can be transposed at a certain point in time or only one particular note within a chord can be transposed (more on this in section 4.4). However, if it is desired to scale the pitch of the entire input signal, the CQT-domain method exhibits computational benefits compared to the two-staged time-scaling/re-sampling approach. In contrast to the phase-vocoder method, the

analysis and synthesis hop sizes are equal in the CQT domain approach, thus, the window overlap can be chosen to produce minimum redundancy (e.g. 50% for a Hann window). Hence, for the phase vocoder method, applying a pitch-scaling factor of 2, for example, twice as many coefficients need to be calculated and subsequently processed compared to the CQT-domain method.

#### 4.1.1 Implementation

Assuming that the desired transposition is an integer multiple of the CQT resolution (e.g. an integer multiple of 25 cent for a resolution of 48 bins per octave), the transposition process can be implemented in four stages:

- **CQT:** As for the CQT phase vocoder, the CQT coefficients need to be oversampled in time by factor two in order to enable correct phase adjustment between adjacent octaves. With an oversampling factor of 2 pitch transpositions up to one octave can be performed. If a pitch transposition of more than one octave is required, the oversampling factor needs to be higher (an oversampling factor of 4 enables transpositions up to 2 octaves).
- **Shift CQT coefficients:** Due to the geometrical bin spacing of the CQT, a pitch-transposition can be performed simply by shifting the entire representation up- or downwards. In figure 4.1 both the original (oversampled) CQT coefficients and the CQT coefficients shifted by +5 bins are depicted.
- **Retain phase coherence:** Shifting a CQT coefficient  $X(k_1, n_1)$  from bin  $k_1$  at the time instant  $n_1$  to bin  $k_2$  at time instant  $n_1$  means, that when the signal is reconstructed, a time-frequency atom with frequency  $f_{k_2}$  will be generated at the same time instant with the magnitude and phase value of coefficient  $X(k_1, n_1)$ . However, the frequency difference  $\Delta f = f_{k_2} - f_{k_1}$  results in a phase jump between consecutive time instants  $n_1$  and  $n_2$ , thus horizontal phase coherence is lost. Hence,

phase coherence must be retained exactly like it is the case for the CQT phase vocoder and the same phase update approaches can be used.

Alternatively, the phase update approach based on the principles outlined in [LD99b] can be applied. As mentioned above, in this approach the applied phase rotation is solely based on the frequency difference between the old peak bin position and the new peak bin position (after transposition), thus circumventing the need of exact frequency estimations. For the linear bin spacing of the DFT, this phase update based on the frequency difference is valid as depicted in figure 4.2 . Here a sinusoidal component with frequency  $\omega_x$  and the corresponding peak bin at frequency  $\omega_k$  is shifted by  $S$  bins to the new frequency  $\omega_y$ . Since  $\omega_{k+S} - \omega_k = \omega_y - \omega_x$ , the phase update to retain horizontal phase coherence can be based on  $S$ . This is not true for the logarithmic bin spacing of the CQT in general, because  $\omega_{k+S} - \omega_k < \omega_y - \omega_x$ . Hence, if this phase update approach is applied to pitch transposition with the CQT, the slightly erroneous phase update will result in small amplitude and frequency modulations. However, informal listening test have shown, that these errors are hardly audible in the output signal when compared to the proper phase unwrapping technique. A possible explanation for this is, that according to [ZF99], the human hearing is insensitive to certain amounts of amplitude and frequency modulation. This fact has also been exploited in [DCL04] to reduce phasiness for a DFT phase vocoder.

- **Inverse CQT:** Reconstruct the pitch-scaled output signal from the modified CQT coefficients (all unnecessary oversampled coefficients are discarded).

This implementation yields a high quality of the transposed output signal, even though systematical errors at note onsets remain: as coefficients are transposed to higher frequencies, the lengths of the corresponding atoms decrease. Because the magnitude values are unchanged, this causes a softening

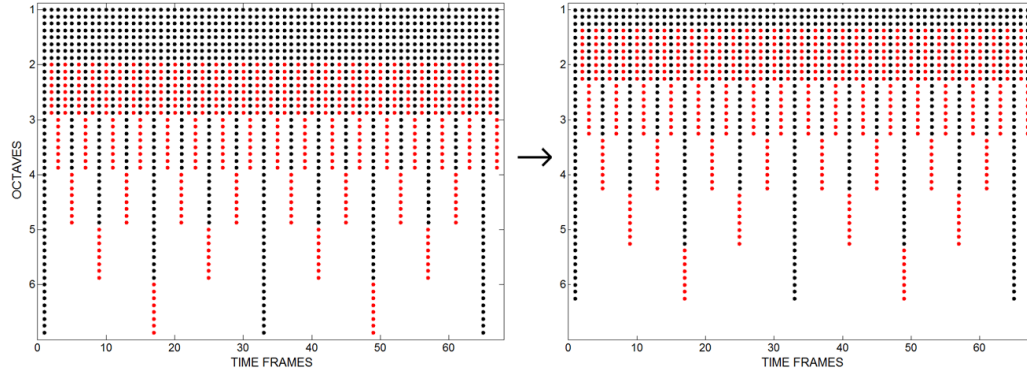


Figure 4.1: Shifting the CQT coefficients for pitch transposition. The left figure depicts the original positions of the CQT coefficients in the time-frequency plane, the right figure depicts the coefficients transposed by +5 bins. The red dots correspond to coefficients obtained by oversampling.

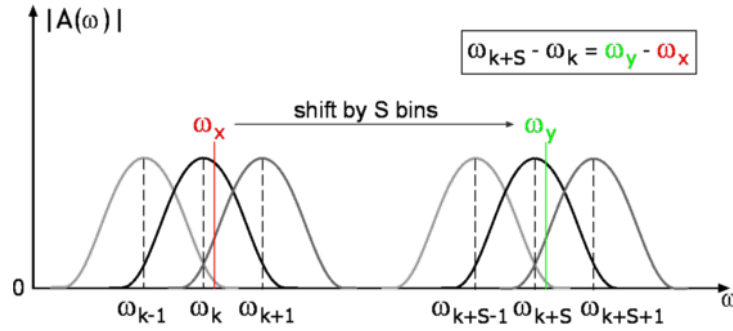



Figure 4.2: DFT: A sinusoidal component with frequency  $\omega_x$  is shifted by  $S$  bins to the new frequency  $\omega_y$ .

of attack transients similar to the CQT phase vocoder.

Note that due to the duality of time- and pitch-scaling, this efficient approach can also be used as a time-scale modification algorithm by re-sampling the transposed output signal. For large stretch factors, however, the input signal should be up-sampled since the outer frequency bins are lost as a consequence of the coefficient shift.

### 4.1.2 Audio Examples

As mentioned above, there are two ways to retain horizontal phase coherence. The transposed versions of the original signal in sample 11 (the amount of transposition is given in semitones) were generated applying proper phase unwrapping, that is phase adjustments based on true frequency estimations.




original | +2 | +6 | +12 | -2 | -6 | -12

---

**Sample 11:** orchestra 1, phase unwrapping based  
on true frequency estimation

The same audio excerpt is transposed in sample 12, but here the phase update is solely based on the frequency difference between the new and the old peak-bin (no frequency estimation). The frequency and amplitude modulations due to the slightly erroneous horizontal phase differences are hardly audible.




original | +2 | +6 | +12 | -2 | -6 | -12

---

**Sample 12:** orchestra 1, phase update without  
frequency estimation

From these samples it can be appreciated, that even for very dense audio signals, very little artifacts are produced by pitch transpositions in the CQT domain. Another example is given in sample 13. The CQT resolution for all samples in this section is set to 48 bins per octave.




original | +2 | +6 | +12 | -2 | -6 | -12

---

**Sample 13:** orchestra 2

The transposition of sharp attack transients yields a softening of attacks for positive transposition factors and over-accentuated attacks for negative transposition factors, respectively, if transients are not processed separately. The reason for this, as described above, is, that a transient in the CQT representation is wider at low frequencies and narrower at higher frequencies. This

can be appreciated in sample 14 where a piano piece is transposed. Despite this effect for attack transients, again very little artifacts are introduced. If large negative transposition factors are applied, the output signal sounds quite dull since there is a lack of high frequency harmonics. This problem, however, is inherent to pitch shifting in general and can only be addressed by generating artificial harmonics (*exciter*).




original | +2 | +6 | +12 | -2 | -6 | -12

---

**Sample 14:** piano

### 4.1.3 Transients

As discussed in 3.2.1, attack transients at higher frequencies are retained in the scaled output signal due to the small window sizes. Attack transients at low frequencies and transients exhibiting most of their energy in low frequencies, however, call for dedicated transient processing. In the samples given above these problems hardly occurred, however, if the input signal contains low bass attacks or even kick drums, the degradation of these signal components in the transposed output signal gets obvious. This degradation is demonstrated in sample 15.



original | +2 | +4 | +6 | +12 | -2 | -4 | -6 | -12

---

**Sample 15:** bass, drums, piano. Bass and kick  
drum attacks are degraded in the transposed output  
signals.

In order to prevent the degradation of transients and transient attacks, the methods discussed in section 3.2.1 for the CQT phase vocoder can also be applied for pitch transpositions in the CQT domain. In the context of pitch-scaling, the transient separation method is of particular interest as the transient part can be added to the transposed output signal without further processing. However, the requirements concerning the reliability of the sepa-

ration algorithm are even more rigorous since tonal components that remain in the transient part are not transposed and produce 'out of tune' artifacts. However, due to the computational efficiency of the transient separation approach proposed in [Fit10], this approach has been adapted for the CQT representation and yields promising results.

The percussive/harmonic separation approach proposed by FitzGerald in [Fit10] is based on the fact, that transients are represented by vertical ridges and partials are represented by horizontal ridges in the STFT spectrogram (the approach proposed in [OMLR<sup>+</sup>08] is based on the same notion). Hence, in order to suppress transients vertical ridges in the spectrogram need to be suppressed. To do so, FitzGerald regards percussive events as outliers across time for a given frequency bin and harmonic events as outliers in the frequency spectrum for a given time frame. From image processing we know that such outliers (i.e. speckle noise or salt and pepper noise) can be removed by applying a median filter. Contrary to image processing applications, two 1-dimensional median filters are applied to obtain a percussion-enhanced (= harmonic suppressed) spectrogram  $\mathbf{P}$  and a harmonic-enhanced spectrogram  $\mathbf{H}$  from the original magnitude spectrogram  $\mathbf{S}$ . Denoting the  $i$ th time frame of the original spectrogram as  $\mathbf{S}_i$  and the  $h$ th frequency slice as  $\mathbf{S}_h$  the desired spectrograms are obtained by

$$\mathbf{P}_i = \mathcal{M}(\mathbf{S}_i, l_p) \quad (4.1)$$

$$\mathbf{H}_h = \mathcal{M}(\mathbf{S}_h, l_h) \quad (4.2)$$

where  $\mathcal{M}$  denotes median filtering and  $l_p$  and  $l_h$  are the respective filter lengths. The spectrograms  $\mathbf{P}$  and  $\mathbf{H}$  are then used to generate masks which can be applied to the original spectrogram. Two families of masks were proposed in [Fit10]:

- **binary mask:** each coefficient in the original spectrogram is assumed



to belong to a percussion or a harmonic source, hence

$$\mathbf{M}_{\mathbf{H}_{h,i}} = \begin{cases} 1, & \text{if } \mathbf{H}_{h,i} > \mathbf{P}_{h,i} \\ 0, & \text{otherwise} \end{cases} \quad (4.3)$$

$$\mathbf{M}_{\mathbf{P}_{h,i}} = \begin{cases} 1, & \text{if } \mathbf{P}_{h,i} > \mathbf{H}_{h,i} \\ 0, & \text{otherwise} \end{cases} \quad (4.4)$$

where  $\mathbf{M}_{\mathbf{H}}$  is the mask to obtain a harmonic enhanced (percussion suppressed) output signal and  $\mathbf{M}_{\mathbf{P}}$  is used to obtain a percussion enhanced output signal.

- **soft mask:** these masks are based on Wiener Filtering and are defined as:

$$\mathbf{M}_{\mathbf{H}_{h,i}} = \frac{\mathbf{H}_{h,i}^p}{(\mathbf{H}_{h,i}^p + \mathbf{P}_{h,i}^p)} \quad (4.5)$$


$$\mathbf{M}_{\mathbf{P}_{h,i}} = \frac{\mathbf{P}_{h,i}^p}{(\mathbf{H}_{h,i}^p + \mathbf{P}_{h,i}^p)} \quad (4.6)$$

where  $p$  denotes the power to which each individual element of the spectrograms are raised (typical values for  $p$  are 1 or 2).


Finally, the desired spectrograms are obtained by element-wise multiplication of the original spectrogram with the respective mask.

This approach has been translated to operate on the sparse CQT representation depicted in figure 4.1. Since only transients at lower frequencies need to be processed in the CQT pitch transposition approach, a percussion enhanced version  $x_p(n)$  of the input signal  $x(n)$  is generated using a soft mask  $\mathbf{M}_{\mathbf{P}}$  where  $\mathbf{M}_{\mathbf{P}_{h,i}} > 0$  only for frequencies below 3000 Hz. The harmonic enhanced signal  $x_h(n)$  is generated using the mask  $\mathbf{M}_{\mathbf{H}} = \mathbf{1} - \mathbf{M}_{\mathbf{P}}$ .

In sample 16 the results of the separation algorithm are demonstrated for the input signal used in sample b6.

	original   percussive part   harmonic part
	<b>Sample 16:</b> bass, drums, piano. The signal is separated in a percussive and a harmonic part.

In order to obtain the frequency transposed output signal, only the harmonic part of the input signal is processed and subsequently added to the unprocessed percussive part. The resulting output signals for different transposition factors are demonstrated in sample 17.

	original   +2   +4   +6   +12   -2   -4   -6   -12
	<b>Sample 17:</b> bass, drums, piano. Improved quality of bass and kick drum attacks in the transposed output signals.

By comparing the transposed versions of the input signal in sample b6 and b8 it can be appreciated, that the transient separation approach considerably improves the quality of kick drum and bass attacks.

As an alternative to the separation approach, the coefficient based phase realignment method outlined in section 3.2.1 can be applied to the shifted CQT representation. As for the CQT phase vocoder, the non-trivial integration of the phase realignment method into the CQT transposition algorithm, however, is beyond the scope of this thesis and remains a topic for future work.

#### 4.1.4 Sources of Artifacts

Constant frequency, constant amplitude sinusoidal components can be transposed in the CQT representation without introducing artifacts as the absolute values of the CQT coefficients do not change if the transposition corresponds to an integer number of bins (as is the case for all transpositions that are multiples of the CQT resolution). If time-varying sinusoidal components

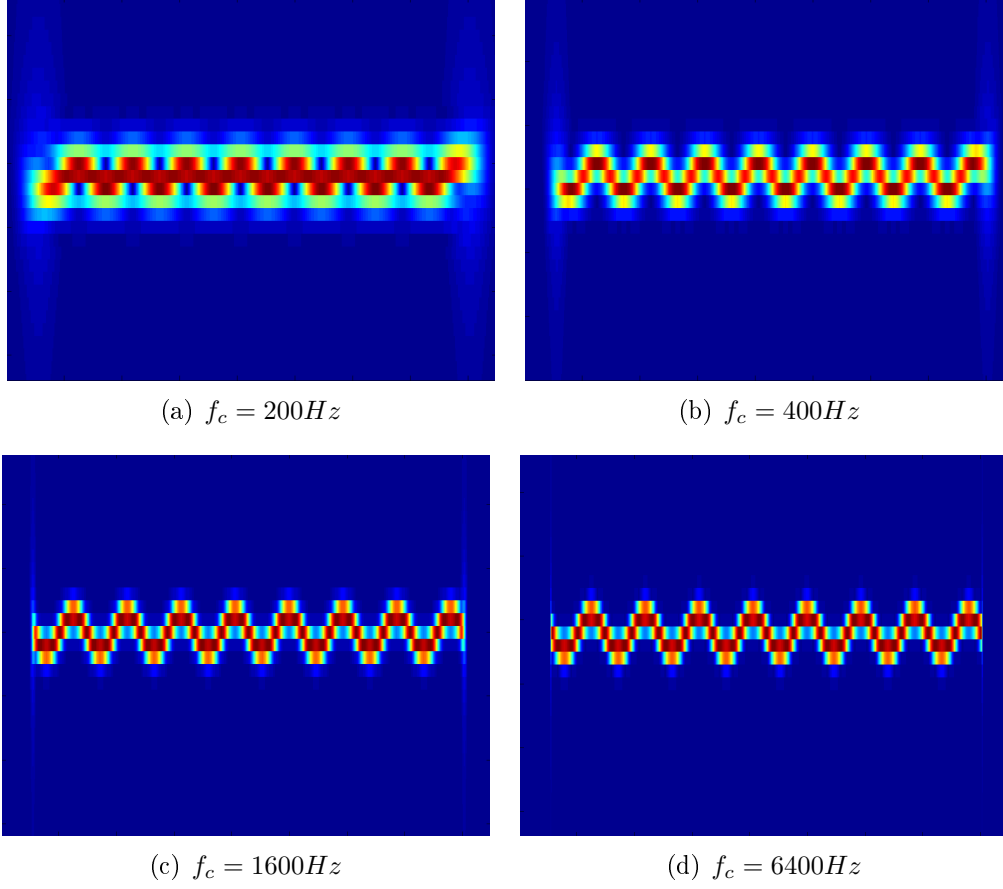


Figure 4.3: CQT representations of four frequency modulated sinusoid. The center frequency  $f_c$  is varied,  $f_m = 4Hz$  and  $f_\Delta = 0.02f_c$  for all signals.

are considered, however, the absolute values of the CQT coefficients also depend on the frequency region as the time resolution decreases towards lower frequencies. Consider the input signal  $x(n)$  being

$$x(n) = \sin \left( 2\pi \frac{f_c}{f_s} n - \frac{f_\Delta}{f_m} \cos \left[ 2\pi \frac{f_m}{f_s} n \right] \right) \quad (4.7)$$

that is,  $x(n)$  is a frequency modulated, constant amplitude sinusoid with center frequency  $f_c$ , the modulation frequency  $f_m$  and the maximum frequency deviation  $f_\Delta$ . In figure 4.3(a)-(d) the CQT representations ( $B=48$ ) of four different frequency modulated sinusoids are depicted, whereas  $f_c$  is varied. The modulation frequency  $f_m = 4Hz$  and the frequency deviation  $f_\Delta = 0.02f_c$

(approx.  $\frac{1}{3}$  semitone) for all cases. It can be observed that although the absolute frequency deviation increases for increasing center frequencies, the deviation in CQT bins is constant - an obvious fact that again demonstrates the advantage of a constant-Q resolution compared to the linear frequency resolution provided by the DFT. However, it can also be observed that due to different time-resolutions, the magnitude values of the CQT coefficients change with varying  $f_c$ . The most dramatic change can be identified from  $f_c = 200Hz$  to  $f_c = 400Hz$  since the temporal evolution of the frequency modulated sinusoid can not be captured properly for low frequencies<sup>1</sup>. By comparing figures 4.3(c) and 4.3(d) it can be observed, that as soon as the temporal resolution reaches a value where the temporal evolution can be captured properly, the magnitude changes are less severe. This demonstrates that for large transposition factors artifacts are introduced when the input signal exhibits quick temporal changes at low frequencies. Limiting the window size towards low frequencies (hybrid DFT-CQT solution as proposed in section 3.3 for the CQT phase vocoder) is not an option for pitch transpositions in the CQT domain as the strictly constant-Q resolution is crucial for this approach.

## 4.2 A Sinusoidal Modeling Approach

In section 4.1.3 it was shown that the quality of the transposed output signal can be improved when the input signal is divided into a (low frequency) percussive and a harmonic part and only the latter is transposed. The same notion could also be formulated in a sinusoidal modeling sense [MQ86] [SS87]. That is, sinusoidal components in the input signal are identified explicitly and subsequently transposed. The main difference between the approach proposed in section 4.1.3 and the sinusoidal modeling approach is, that the goal is to extract and transpose only partials rather than trying to suppress

---

<sup>1</sup>Note that if a CQT resolution of 24 bins per octave would have been chosen instead of 48, time resolution would have been twice as good. Since in the previous sound examples usually  $B=48$  was used, however, this value is kept also for these figures.

low frequency transients. This is reasonable since transient and noisy signal components should not be affected by pitch transpositions at all.

Usually a sinusoidal modeling scheme comprises three stages on the analysis side:

- **peak identification:** spectral peaks are marked as potential elements of sinusoidal trajectories
- **parameter extraction:** the parameters amplitude, frequency and phase associated with each peak are determined
- **peak continuation (tracking):** spectral peaks are grouped to form sinusoidal trajectories representing the temporal evolution of partials

The most challenging stage in this process is the linking of spectral peaks to obtain meaningful sinusoidal tracks representing real partials whereas spectral peaks stemming from transients or noise need to be discarded. The first tracking algorithm was proposed by McAulay and Quatieri in [MQ86] for the sinusoidal modeling of monophonic speech. In this approach two spectral peaks in consecutive frames are linked together if their frequency difference is lower than a given threshold (an extensive review of this approach is given in [CDL06]). For complex music signals, however, this simple approach often yields unsatisfying results since some peaks may be missing along a sinusoidal track and partials may cross if polyphonic, multi-instrument input signals are considered. Therefore, a range of more sophisticated approaches has been proposed to address these problems. In [DGR93] [KD08] a statistical approach based on Hidden Markov Models has been proposed to improve sinusoidal tracking. In [LMRR03] and [LMR04] partial tracking is improved by employing linear prediction.

In the context of pitch transposition, the estimated parameters amplitude and frequency of found partials could be altered to synthesize a transposed harmonic signal. Although the logarithmic frequency resolution of the CQT may improve parameter extraction for sinusoidal modeling, this approach is outside the scope of this thesis and thus is not investigated into. However,


the identified partials obtained by the analysis stage of a sinusoidal modeling system can also be used to extract harmonic sources from the CQT representation (a related application of the sinusoidal modeling approach has been proposed in [VK00]) to obtain a CQT representation that only contains sinusoidal components. Pitch transposition is then applied to this reduced representation as discussed in section 4.1.1 and subsequently added to the residual CQT representation containing transient and noise parts.

In the course of this thesis this approach has been implemented using a very simple tracking algorithm similar to the implementation proposed in [MQ86]. The results obtained for complex music signals, however, were outperformed by the implemented approach discussed in section 4.1. It is assumed that the main reason for that is the poor tracking algorithm and a more sophisticated approach probably would lead to better results. However, the incorporation of a more advanced tracking algorithm remains a topic for future work.

### 4.3 Formant Preservation

The ultimate goal of pitch transposition is to obtain an output signal that sounds as natural as possible. Transposing harmonic components while introducing no or very little artifacts is the first step towards this goal. The next step is to look at the physical properties of the transposed signal sources in order to achieve an output signal that sounds simply as if the audio signal was played or sung at a different pitch. Many instruments, especially the human voice, however, exhibit a spectral envelope that is more or less independent of the fundamental frequency. This is true for instruments where a source signal is filtered by a resonant body or, in case of the human voice, the vocal tract. Frequency regions that are amplified by this filter are called formants and are characteristic to many instruments and vowels in speech signals. In order to obtain a natural sounding transposed output signal these formants have to be preserved. For example, consider sample 18, where a recording of a female singer is scaled simply by transposing the frequency

content without minding the spectral envelope.

	original   +2   +4   +6   +12   -2   -4   -6   -12
<b>Sample 18:</b> female singer	

Obviously, the character of the voice in this sample changes dramatically as the frequency content is transposed. In order to retain characteristic formants, the spectral envelope (a smooth function passing through the prominent peaks of the spectrum) needs to be estimated. Using this estimation the spectral envelope of the input signal can be flattened prior to pitch-scaling and finally the original envelope is applied to the transposed output signal. Several techniques to gain an estimate of the spectral envelope have been proposed, most prominently approaches based on linear prediction [MG82] or the real cepstrum [OS75] such as [IA79], [GR90], [CM96] and [RR05].

Even though these techniques work well for speech signals or singing voice, the main problem with formant preservation in the context of pitch-scaling is, that the concept of spectral envelope preservation does not entirely hold for all instruments. For several instruments, e.g. the flute, the spectral envelope of the harmonic structure changes significantly as the pitch changes and also the first formant of the human voice changes its position in frequency. Hence, preserving the formant structure in complex audio signals (i.e. polyphonic, multi-source signals) is an extensive problem incorporating topics like blind source separation [CJ10] and musical instrument identification [LR04]. Since the CQT approach to pitch transposition predominantly aims at complex audio signals and formant preservation for this signal class is beyond the scope of this thesis, no spectral envelope preservation scheme has been implemented in the course of this work.

## 4.4 Note Selective Transposition

In section 4.1 it has been shown that pitch transpositions of the entire input signal can be performed with ease in the CQT domain. This frequency-domain approach to pitch-scaling provides an alternative to the two-staged time-scaling/resampling implementation. Manipulating an audio signal in the frequency domain, however, opens up the possibility to transpose only certain regions in the time-frequency plane, e.g. transposing only one or several notes in a chord. The implementation of this effect based on the STFT representation of the input signal, as proposed in [LD99b], exhibits several disadvantages:

- the detection of harmonic patterns in the STFT representation is not trivial
- each harmonic corresponding to the same note needs to be shifted by a different number of bins
- in order to determine the desired frequency shift for each partial, their exact frequencies need to be estimated
- in most cases these shifts correspond to fractional bin numbers, hence the shifted coefficients need to be interpolated

If note selective pitch transposition is performed in the CQT domain, however, none of these issues occur. Due to the logarithmic frequency resolution provided by the CQT, harmonics of a stable tone exhibit spectral peaks that can be described with a fixed pattern that is independent of the fundamental frequency. Using a CQT resolution of 48 bins per octave, the peaks of the first five overtones of a fundamental frequency  $F_0$  existing a peak bin  $p$  can be found exactly at bins  $p+48$ ,  $p+76$ ,  $p+96$ ,  $p+112$ ,  $p+124$ . Hence, harmonics in the CQT representation can be selected by applying a predefined spectral mask. To transpose one tone in the representation, the selected harmonics subsequently can be shifted up- or downwards by a simple copy&paste (or cut&paste) operation. As for the case where the entire representation is



transposed, the phases of the shifted frequency regions have to be updated to retain phase coherence. In section 4.1 it was shown, that this phase update can be solely based on the difference between the old and the new peak-bin frequency rather than the actual frequency shift without introducing severe artifacts. That is, harmonic patterns can be transposed in the CQT domain without performing any frequency estimations. Finally, as mentioned in section 4.1, if transpositions corresponding to integer multiples of CQT bins are considered, that is integer multiples of 25 cent using a CQT resolution of 48 bins per octave, no coefficient interpolation is needed.

### **Exemplary Demonstration**

In order to achieve a high quality output signal when performing selective note transposition, there are a few more things to consider. To prevent transient smearing, the attack part of a note should not be transposed. This is especially important for notes with low fundamental frequencies. As demonstrated in section 4.1.3 this can be avoided by separating low frequency transients prior to pitch transposition applying a separation approach based on median filtering. Another source of error when single notes in complex audio signals are transposed is the fact, that harmonics of different tones frequently heterodyne at a single CQT bin. For example consider the chord C1-E1-G1: the second overtone of C1 has the same frequency as the first overtone of G1, the fourth overtone of C1 has the same frequency as the third overtone of E1, and so on. In order to achieve a high quality output signal, the entire harmonic pattern of one note has to be extracted without corrupting the harmonic pattern of another note. The known problem of separating overlapping partials is closely related to sound source separation [Vir06] and has been explicitly addressed in [VK02] [VE06] [ES06]. Although a constant-Q resolution could trigger improvements in this research area, this issue has not been addressed in the course of this thesis and remains a topic for future work. However, in the sequel it is demonstrated (by means of a piano chord) that even without proper separation of overlapping partials selective note

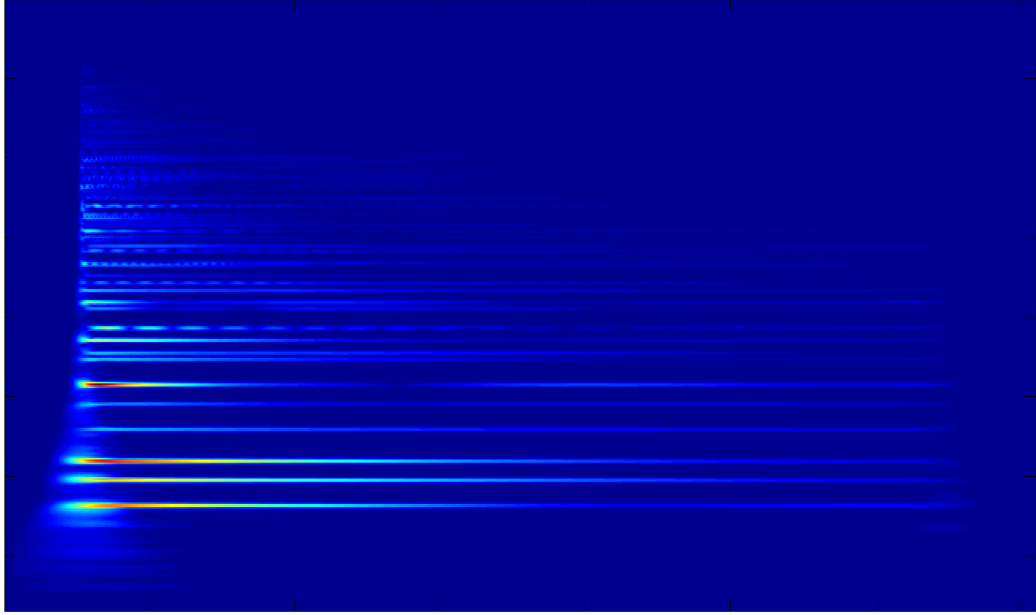


Figure 4.4: Rasterized CQT representation of the input signal (piano chord)

transposition can be performed with satisfying quality.

In figure 4.4 the CQT representation of a simple C chord played on a piano is depicted using a CQT resolution of 48 bins per octave. The notion is to transpose the major third (E) in this C chord down by one semitone ( $E_b$ ) to obtain a  $C_m$  chord. To do so, the first 6 harmonics of the major third are selected applying a predefined mask (figure 4.5) to each CQT frame<sup>2</sup>. Note that due to the varying time resolution across the frequency range the different harmonics do not start and end simultaneously. Hence, the coefficients in the onset and offset regions are only captured if the CQT representation exhibits a prominent peak at the corresponding bins. In figure 4.6 the selected harmonics are marked in the CQT representation. Subsequently the selected harmonics are shifted downwards by 4 bins applying a simple cut&copy operation. The CQT representation thus obtained is depicted in figure 4.7.

---

<sup>2</sup>In the actual implementation the sparse representation of the CQT is used, hence, not every frame contains coefficients different from zero for the respective bins.

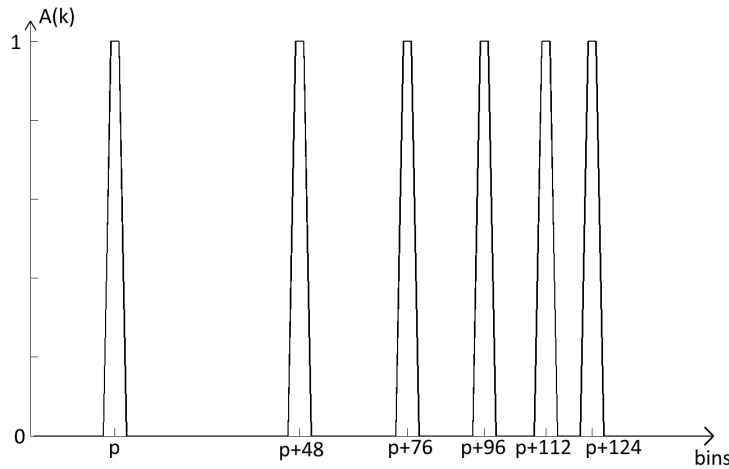


Figure 4.5: Predefined harmonic mask to capture the first six harmonics of a given note with a fundamental frequency exiting the peak bin  $p$ .

In sample 19 the original input signal and the altered output signal can be compared. It can be appreciated that the achieved quality is rather high although only the first 6 harmonics have been considered and the error due to the superposition of harmonics has been ignored.

original: C chord on piano | shifted to Cm

---

**Sample 19:** piano chord (C  $\rightarrow$  Cm)

To demonstrate that also greater shifts of single notes do not severely impair the quality of the output signal, the the same harmonic pattern has been transposed by +6 semitones (this time applying a copy&paste operation) to obtain a C7 chord. The result is demonstrated in sample 20.

original: C chord on piano | added minor 7 to get C7

---

**Sample 20:** piano chord (C  $\rightarrow$  C7)

Note that this is just a simple example to demonstrate the potential of note selective transposition in the CQT domain. If notes exhibiting frequency modulation (vibrato) are considered for example, the method needs to be expanded by sinusoidal trajectory tracking in order to adapt the harmonic

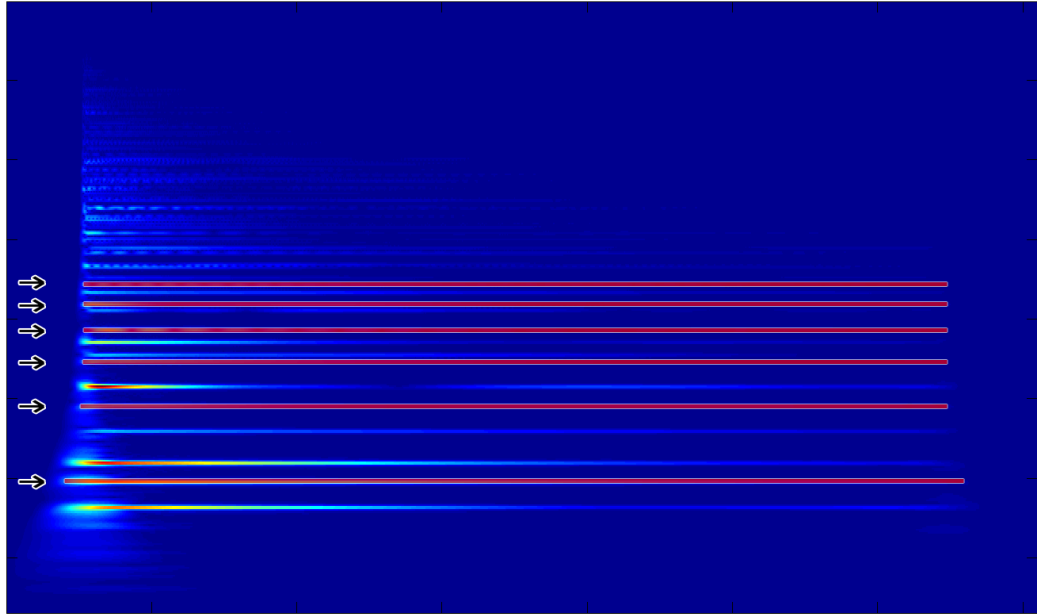


Figure 4.6: Rasterized CQT representation of the input signal. The selected harmonics that are about to be transposed are marked with red lines.

mask to capture the desired harmonics. However, the ease of simple shift operations remains also for tones with varying fundamental frequencies.

## 4.5 Conclusion

As the frequency resolution provided by the CQT matches the fundamental frequencies in western music, pitch transpositions in the frequency domain can be performed by simply shifting the CQT coefficients followed by a phase update stage to preserve phase coherence. In this chapter two methods to perform the phase update have been outlined:

- horizontal phase unwrapping (instantaneous frequency estimation) + vertical phase locking
- horizontal phase update based on the old and new peak-bin frequencies (no frequency estimation) + vertical phase locking

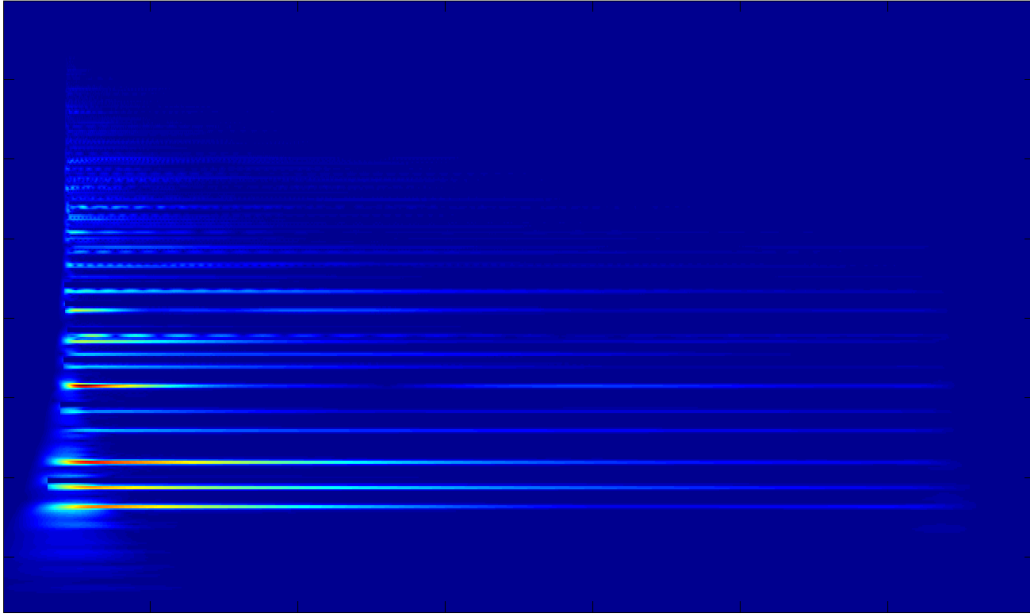


Figure 4.7: Altered CQT representation. The major third in the C chord is transposed by -1 semitone to obtain a Cm chord.

While being computational less expensive, the latter method introduces slight amplitude and frequency modulations. Informal listening test, however, suggest that these errors do not impair the perceived quality of the transposed output signal. That is, pitch transpositions in the CQT domain can be implemented very efficiently without performing frequency estimations. Extensive listening tests to prove this suggestion are yet to be conducted.

Another property that improves computational efficiency of pitch transpositions in the CQT domain compared to the phase vocoder time-scaling and resampling approach is, that the analysis and the synthesis hop sizes are equal. Hence, the hop size can be chosen to produce minimal redundancy yielding less CQT coefficients that need to be processed. This fact might also be appealing for time-scaling purposes by first transposing the entire input signal in the CQT domain and subsequently applying a re-sampling stage. Note that in order to avoid the loss of outer frequency bins in this scenario, the minimum and maximum frequency of the CQT have to be chosen accordingly. For large stretch factors this might include oversampling of the input

signal so that the maximum frequency can be chosen to be higher than the initial Nyquist frequency.

As for the CQT phase vocoder, low-pitched transients pose a problem in the context of pitch transpositions due to very long windows for low frequencies. It has been shown, that the adaption of the STFT based harmonic/percussive separation approach proposed in [Fit10] for the CQT signal representation is able to significantly mitigate transient artifacts occurring at low frequencies.

In this chapter also the basic principles of *note selective transpositions* have been outlined and its potential to produce high quality output signals using a simple implementation has been demonstrated. Further improvements concerning the achieved quality and the applicability to arbitrary music signals are expected by incorporating an energy based harmonic pattern separation approach and sinusoidal trajectory tracking ([LMR05] [LMR07] [KD08]).

## Chapter 5

# Summary and Future Work

In this thesis the CQT toolbox we proposed in [SK10] providing an efficient implementation of the CQT for broadband audio signals and a reasonable quality reconstruction of the original input signal from its CQT coefficients has been utilized to manipulate the time- and the pitch-scale of polyphonic music signals. The efficient implementation is based on the algorithm proposed in [BP92] for which two extensions are proposed:

- The CQT kernel matrix is defined only for the highest octave of interest. Thus a relatively small DFT frame length can be used causing the spectral kernel matrix to be sparse and CQT coefficients can be calculated from the DFT coefficients requiring only few complex multiplications. The CQT coefficients of lower octaves are calculated by applying the same kernel to the successively downsampled input signal.
- To further reduce computational complexity, the CQT kernel is populated with several atoms per frequency allowing for a large DFT hop size reducing the overall number of DFT frames.

The proposed approach to reconstruct the signal from its CQT coefficients is straightforward. If the used window functions and overlap factors are chosen such that time-frequency atoms *approximately* sum up to unity both in time

and frequency domain, the complex conjugate of the kernel matrix can be used for the inverse CQT yielding reasonable quality reconstruction (around 55 dB SNR) at a redundancy factor of around 4 to 5. In [SK10] we suggested to use Blackman-Harris windows in order to minimize the reconstruction error. In the course of this thesis, however, it was found that very low reconstruction errors can also be achieved by using squared Hann windows with the benefit of lower main lobe widths.

## 5.1 Contributions

Implementations for time- and pitch-scale modifications of polyphonic audio signals based on the CQT representation has been proposed. It has been shown that the quality of the time-scaled output signal can be significantly improved when the STFT in the standard phase vocoder approach is replaced with the CQT. This improvement stems from the frequency bin spacing featured by the CQT that matches the fundamental frequencies in western music enhancing the ability to resolve sinusoidal components in the input signal. In order to facilitate phase modifications in the CQT representation, the implementation of the CQT proposed in chapter 2 has been adapted:

- A modified Hann window was introduced that allows for fractional window lengths and guarantees sampling of the window center. Each atom stack can thus be centered at the exact same point in the temporal kernel (stack center) and rounding of the window lengths can be omitted.
- To obtain zero phase difference between adjacent bins excited by the same sinusoidal component, the phases of all atoms are adjusted such that their phases are zero at the common stack center.

It has been shown that, when the synthesis hop size is altered in order to obtain the desired stretching factor, phase coherence can be restored applying phase unwrapping and phase locking schemes known from the standard STFT phase vocoder. As the objective was to demonstrate the quality improvement



obtained by replacing the STFT with the CQT in the phase vocoder approach rather than developing a highly sophisticated time-scaling algorithm, extensions such as the incorporation of sinusoidal trajectory heuristics [KLB06] or shape-invariant approaches [Roe10] have not been implemented. In terms of transient signal components it has been shown that high pitched transients and attack transients can be preserved in the time-scaled output signal due to the improved time-resolution of the CQT at higher frequencies. On the other hand, due to large window lengths in low frequency areas, transient smearing for low-pitched transient components is an issue in the CQT phase vocoder. To address this problem a simple phase realignment approach has been outlined but is yet to be implemented as part of the CQT phase vocoder.

In a second contribution, a novel approach to pitch-scaling in the time-frequency domain has been proposed. Exploiting the geometrical bin spacing of the CQT representation it has been demonstrated that pitch transpositions can be performed by simply shifting the CQT coefficients and subsequently updating the phases. This marks a significant facilitation compared to pitch transpositions in the frequency domain based on the STFT representation. Two variations of the phase update process have been proposed:

- Phase unwrapping and phase locking as discussed for the phase vocoder.
- Phase update solely based on the frequency difference between the old and new peak bin position.

In the latter implementation estimating the true frequency of sinusoidal components is omitted, thus reducing the computational complexity of the algorithm. This simplification introduces slight amplitude and frequency modulations due to errors of the horizontal phase progression. However, informal listening tests suggest that these modulations are hardly audible in the transposed output signal. This is yet to be confirmed in formal listening tests.

As pitch transpositions usually degrade the quality of transient signal components, the percussive/harmonic separation algorithm proposed in [Fit10] has been adapted for the CQT representation since there is no need to transpose

percussive signal parts. After transposing the harmonic part the percussive part is added back to the output signal without further processing. It was demonstrated that the application of this efficient algorithm significantly improves the quality of transients in the transposed output signal.

By means of a piano chord it has been demonstrated that pitch transpositions are not confined to be applied to the entire input signal but can also be applied to single notes with ease. Harmonic patterns are easily spotted in the CQT representation and note selective transpositions can be obtained by simple copy&paste and cut&paste operations, respectively.

## 5.2 Future Work

Throughout this thesis several suggestions for possible future work have been given in each chapter. These suggestions will be summarized for the discussed applications in the next paragraphs.

**CQT phase vocoder:** To overcome the problems that arise for sharp transients in lower frequency areas several approaches have been outlined. The incorporation of one of the following techniques (or a combination of them) in the CQT phase vocoder implementation is an open issue:

- Phase realignment for transients: see section 3.2.1.
- Transient separation: the input signal is split into a percussive and a harmonic part (see section 4.1.3). Percussive events are then translated in time and added to the time-scaled harmonic part.
- Limiting the window lengths: to avoid very long windows in the first place, the window lengths could be limited towards low frequencies (think hybrid DFT/CQT phase vocoder).

Further improvements to the simple CQT phase vocoder implementation could be made by incorporating sinusoidal trajectory heuristics [KLB06] and shape invariant approaches [Roe10].

**Pitch transposition:** In order to obtain a fully functional pitch-shifting algorithm based on the CQT, a formant preservation technique needs to be implemented. As for the CQT phase vocoder, phase coherence could be further improved by incorporating sinusoidal trajectory heuristics and shape invariant approaches.

**Note selective transpositions:** The ease of transposing single notes in the CQT representation of polyphonic music signals has been demonstrated. However, the implementation of a piece of software that facilitates note selective transpositions in the CQT representation is a topic for future work.



# Bibliography

- [ABLS02] X. Amatriain, J. Bonada, A. Loscos, and X. Serra, *DAFX: Digital Audio Effects (Udo Zölzer ed.)*. John Wiley and Sons, Ltd, 2002.
- [AKZ<sup>+</sup>02] D. Arfib, F. Keiler, U. Zölzer, V. Verfaillie, and J. Bonada, “Time-frequency processing,” *DAFX: Digital Audio Effects*, pp. 219–278, 2002.
- [BDJ<sup>+</sup>11] P. Balazs, M. Dörfler, F. Jaillet, N. Holighaus, and G. Velasco, “Theory, implementation and applications of nonstationary gabor frames,” *Journal of Computational and Applied Mathematics*, 2011.
- [BM67] E. Brigham and R. Morrow, “The fast fourier transform,” *Spectrum, IEEE*, vol. 4, no. 12, pp. 63–70, 1967.
- [BO74] C. Braccini and A. Oppenheim, “Unequal bandwidth spectral analysis using digital frequency warping,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 22, no. 4, pp. 236–244, 1974.
- [Boa03] B. Boashash, *Time frequency signal analysis and processing: a comprehensive reference*. Elsevier Science, 2003.
- [Bon00] J. Bonada, “Automatic technique in frequency domain for near-lossless time-scale modification of audio,” in *Proceedings of In-*

- ternational Computer Music Conference*. Citeseer, 2000, pp. 396–399.
- [BP92] J. Brown and M. Puckette, “An efficient algorithm for the calculation of a constant  $q$  transform,” *JOURNAL-ACOUSTICAL SOCIETY OF AMERICA*, vol. 92, pp. 2698–2698, 1992.
- [Bro91] J. Brown, “Calculation of a constant  $q$  spectral transform,” *Journal of the Acoustical Society of America*, vol. 89, no. 1, pp. 425–434, 1991.
- [BS06] J. J. Burred and T. Sikora, “Comparison of frequency-warped representations for source separation of stereo mixtures,” in *121st Audio Engineering Society Convention*, San Francisco, CA, USA, 2006.
- [BS09] I. Bayram and I. W. Selesnick, “Frequency-domain design of overcomplete rational-dilation wavelet transforms,” *IEEE Trans. on Signal Processing*, vol. 57, no. 8, pp. 2957–2972, 2009.
- [CDL06] E. Coyle, D. Dorran, and R. Lawlor, “A comparison of time-domain time-scale modification algorithms,” in *Audio Engineering Society Convention 120*, 5 2006. [Online]. Available: <http://www.aes.org/e-lib/browse.cfm?elib=13478>
- [Chr03] O. Christensen, *An introduction to frames and Riesz bases*. Birkhauser, 2003.
- [CJ10] P. Comon and C. Jutten, *Handbook of Blind Source Separation: Independent component analysis and applications*. Academic Press, 2010.
- [CM96] O. Cappé and E. Moulines, “Regularization techniques for discrete cepstrum estimation,” *Signal Processing Letters, IEEE*, vol. 3, no. 4, pp. 100–102, 1996.

- [Coh95] L. Cohen, *Time-frequency analysis: theory and applications*. Prentice-Hall, Inc., 1995.
- [DCL04] D. Dorran, E. Coyle, and R. Lawlor, “An efficient phasiness reduction technique for moderate audio time-scale modification,” in *Conference papers*, 2004, p. 25.
- [DDS02] C. Duxbury, M. Davies, and M. Sandler, “Improved time-scaling of musical audio using phase locking at transients,” in *112th AES Convention*. Audio Engineering Society, 2002.
- [DGBA00] A. De Götzen, N. Bernardini, and D. Arfib, “Traditional implementations of a phase vocoder: the tricks of the trade,” in *Proceedings Workshop on Digital Audio Effects (DAFx-00)*, Verona, Italy, 2000.
- [DGR93] P. Depalle, G. Garcia, and X. Rodet, “Tracking of partials for additive sound synthesis using hidden markov models,” in *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on*, vol. 1. IEEE, 1993, pp. 225–228.
- [DL04] D. Dorran and R. Lawlor, “Time-scale modification of music using a synchronized subband/time-domain approach,” in *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP’04). IEEE International Conference on*, vol. 4. IEEE, 2004, pp. iv–225.
- [Dol86] M. Dolson, “The phase vocoder: A tutorial,” *Computer Music Journal*, vol. 10, no. 4, pp. 14–27, 1986.
- [Dör02] M. Dörfler, “Gabor analysis for a class of signals called music,” Ph.D. dissertation, PhD thesis, NuHAG, University of Vienna, 2002.
- [DS52] R. Duffin and A. Schaeffer, “A class of nonharmonic fourier series,” *Transactions of the American Mathematical Society*, pp.

- 341–366, 1952.
- [ES06] M. Every and J. Szymanski, “Separation of synchronous pitched notes by spectral filtering of harmonics,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 14, no. 5, pp. 1845–1856, 2006.
- [FCC06] D. FitzGerald, M. Cranitch, and M. T. Cychowski, “Towards an inverse constant Q transform,” in *120th Audio Engineering Society Convention*, Paris, France, 2006.
- [FEJ54] G. Fairbanks, W. Everitt, and R. Jaeger, “Method for time of frequency compression-expansion of speech,” *Audio, Transactions of the IRE Professional Group on*, vol. 2, no. 1, pp. 7–12, 1954.
- [Fit10] D. Fitzgerald, “Harmonic/percussive separation using median filtering,” in *Proc. of the 10th Int. Conf. on Digital Audio Effects (DAFx10)*, 2010.
- [Fla66] J. Flanagan, “Phase vocoder,” *The Bell System Technical Journal*, pp. 1493–1509, 1966.
- [Gab46] D. Gabor, “Theory of communication. part 1: The analysis of information,” *Electrical Engineers-Part III: Radio and Communication Engineering, Journal of the Institution of*, vol. 93, no. 26, pp. 429–441, 1946.
- [Gam71] G. Gambardella, “A contribution to the theory of short-time spectral analysis with nonuniform bandwidth filters,” *Circuit Theory, IEEE Transactions on*, vol. 18, no. 4, pp. 455–460, 1971.
- [Gam79] —, “The mellin transforms and constant-q spectral analysis,” *The Journal of the Acoustical Society of America*, vol. 66, p. 913, 1979.



- [Gar53] W. Garvey, “The intelligibility of abbreviated speech patterns.” *Quarterly Journal of speech*, 1953.
- [GL84] D. Griffin and J. Lim, “Signal estimation from modified short-time fourier transform,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 32, no. 2, pp. 236–243, 1984.
- [GR90] T. Galas and X. Rodet, “An improved cepstral method for deconvolution of source-filter systems with discrete spectra: Application to musical sounds,” in *Proc. Int. Computer Music Conf*, 1990, pp. 82–84.
- [HA08] F. Hlawatsch and F. Auger, *Time-frequency analysis: concepts and methods*. Wiley-ISTE, 2008.
- [Ham01] F. Hammer, “Time-scale modification using the phase vocoder,” *Institute for Electronic Music and Acoustics (IEM), Graz, Austria*, 2001.
- [Har76] F. Harris, “High-resolution spectral analysis with arbitrary spectral centers and arbitrary spectral resolutions\* 1,” *Computers & Electrical Engineering*, vol. 3, no. 2, pp. 171–191, 1976.
- [Har78] —, “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.
- [HKS<sup>+</sup>00] A. Härmä, M. Karjalainen, L. Savioja, V. Välimäki, U. K. Laine, and J. Huopaniemi, “Frequency-warped signal processing for audio applications,” *J. Audio Eng. Soc.*, vol. 48, no. 11, pp. 1011–1031, 2000.
- [HMC89] C. Hamon, E. Mouline, and F. Charpentier, “A diphone synthesis system based on time-domain prosodic modifications of speech,” in *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on*. IEEE, 1989, pp. 238–241.

- [IA79] S. Imai and Y. Abe, "Spectral envelope extraction by improved cepstral method (in japanese)," *Journal of IEICE*, vol. 62, pp. 217–223, 1979.
- [JL03] E. Jacobsen and R. Lyons, "The sliding dft," *Signal Processing Magazine, IEEE*, vol. 20, no. 2, pp. 74–80, 2003.
- [JSBA08] N. Juillerat, S. Schubiger-Banz, and S. Arisona, "Low latency audio pitch shifting in the time domain," in *Audio, Language and Image Processing, 2008. ICALIP 2008. International Conference on.* IEEE, 2008, pp. 29–35.
- [KC07] J. Kovacevic and A. Chebira, "Life beyond bases: The advent of frames (part i)," *Signal Processing Magazine, IEEE*, vol. 24, no. 4, pp. 86–104, 2007.
- [KD08] C. Kereliuk and P. Depalle, "Improve hidden markov model partial tracking through time-frequency analysis," *Digital Audio Effects*, 2008.
- [KLB06] T. Karrer, E. Lee, and J. Borchers, "Phavorit: A phase vocoder for real-time interactive time-stretching," in *Proceedings of the ICMC 2006 International Computer Music Conference*, 2006, pp. 708–715.
- [KMR85] K. Kashima and B. Mont-Reynaud, "The bounded-q approach to time-varying spectral analysis," Tech. Rep. STANM-28, Stanford University, Department of Music, Stanford, Calif, USA, Tech. Rep., 1985.
- [KV93] J. Kovacevic and M. Vetterli, "Perfect reconstruction filter banks with rational sampling factors," *IEEE Trans. on Signal Processing*, vol. 41, pp. 2047–2066, 1993.
- [LD97] J. Laroche and M. Dolson, "Phase-vocoder: About this phasiness business," in *Applications of Signal Processing to Audio*

- and Acoustics, 1997. 1997 IEEE ASSP Workshop on.* IEEE, 1997, pp. 4–pp.
- [LD99a] —, “Improved phase vocoder time-scale modification of audio,” *Speech and Audio Processing, IEEE Transactions on*, vol. 7, no. 3, pp. 323–332, 1999.
- [LD99b] —, “New phase-vocoder techniques for pitch-shifting, harmonizing and other exotic effects,” in *Applications of Signal Processing to Audio and Acoustics, 1999 IEEE Workshop on.* IEEE, 1999, pp. 91–94.
- [Lee72] F. Lee, “Time compression and expansion of speech by the sampling method,” *Journal of the audio engineering society*, vol. 20, no. 9, pp. 738–742, 1972.
- [LF99] B. Lawlor and A. Fagan, “A novel high quality efficient algorithm for time-scale modification of speech,” in *Sixth European Conference on Speech Communication and Technology*, 1999.
- [LI98] S. Levine and J. Iii, “A sines+ transients+ noise audio representation for data compression and time/pitch scale modications,” in *Proceedings of the 105th Audio Engineering Society Convention.* Citeseer, 1998.
- [LMR04] M. Lagrange, S. Marchand, and J. Rault, “Using linear prediction to enhance the tracking of partials [musical audio processing],” in *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP’04).* IEEE International Conference on, vol. 4. IEEE, 2004, pp. iv–241.
- [LMR05] M. Lagrange, S. Marchand, and J. Raultt, “Tracking partials for the sinusoidal modeling of polyphonic sounds,” in *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP’05).* IEEE International Conference on, vol. 3. IEEE, 2005, pp. iii–229.

- [LMR07] M. Lagrange, S. Marchand, and J. Rault, “Enhancing the tracking of partials for the sinusoidal modeling of polyphonic sounds,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 5, pp. 1625–1634, 2007.
- [LMRR03] M. Lagrange, S. Marchand, M. Raspaud, and J. Rault, “Enhanced partial tracking using linear prediction,” in *Proceedings of the Digital Audio Effects Conference, London*, 2003.
- [LR04] A. Livshin and X. Rodet, “Musical instrument identification in continuous recordings,” in *Proc. of the 7th Int. Conf. on Digital Audio Effects*, 2004, pp. 1–5.
- [LVS98] S. Levine, T. Verma, and J. Smith, “Multiresolution sinusoidal modeling for wideband audio with modifications,” in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 6. IEEE, 1998, pp. 3585–3588.
- [Mal09] S. Mallat, *A wavelet tour of signal processing*, 3rd ed. Academic Press, 2009.
- [MB96] P. Masri and A. Bateman, “Improved modelling of attack transients in music analysis-resynthesis,” in *Proceedings of the International Computer Music Conference*. INTERNATIONAL COMPUTER MUSIC ASSOCIATION, 1996, pp. 100–103.
- [MC90] E. Moulines and F. Charpentier, “Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones,” *Speech communication*, vol. 9, no. 5-6, pp. 453–467, 1990.
- [MG82] J. Markel and A. Gray, *Linear prediction of speech*. Springer-Verlag New York, Inc., 1982.
- [Moo78] J. Moorer, “The use of the phase vocoder in computer music applications,” *Journal of the Audio Engineering Society*, vol. 26,

- no. 1/2, pp. 42–45, 1978.
- [Moo95] B. Moore, *Hearing. Handbook of perception and cognition*. Elsevier Science & Technology, 1995.
- [Moo03] ———, *An introduction to the psychology of hearing*. Emerald Group Pub Ltd, 2003.
- [MQ86] R. McAulay and T. Quatieri, “Speech analysis/synthesis based on a sinusoidal representation,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 34, no. 4, pp. 744–754, 1986.
- [OJS71] A. Oppenheim, D. 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.
- [OMLR<sup>+</sup>08] N. Ono, K. Miyamoto, J. Le Roux, H. Kameoka, and S. Sagayama, “Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram,” in *Proc. EUSIPCO*, 2008.
- [OS75] A. Oppenheim and R. Schaffer, “Digital signal processing,” *Englewood Cliffs, New Jersey, Prentice Hall*, vol. 146, p. 148, 1975.
- [OSB<sup>+</sup>89] A. Oppenheim, R. Schaffer, J. Buck *et al.*, *Discrete-time signal processing*. Prentice hall Englewood Cliffs, NJ., 1989, vol. 1999.
- [Por76] M. Portnoff, “Implementation of the digital phase vocoder using the fast fourier transform,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 24, no. 3, pp. 243–248, 1976.
- [Puc95] M. Puckette, “Phase-locked vocoder,” in *Applications of Signal Processing to Audio and Acoustics, 1995., IEEE ASSP Workshop on*. IEEE, 1995, pp. 222–225.

- [QM92] T. Quatieri and R. McAulay, “Shape invariant time-scale and pitch modification of speech,” *Signal Processing, IEEE Transactions on*, vol. 40, no. 3, pp. 497–510, 1992.
- [Röb03] A. Röbel, “A new approach to transient processing in the phase vocoder,” in *Proc. of the 6th Int. Conf. on Digital Audio Effects (DAFx03)*. Citeseer, 2003, pp. 344–349.
- [Roe10] A. Roebel, “A shape-invariant phase vocoder for speech transformation,” in *Proc. Digital Audio Effects (DAFx)*, 2010.
- [RR05] A. Röbel and X. Rodet, “Efficient spectral envelope estimation and its application to pitch shifting and envelope preservation,” in *Proc. DAFx*. Citeseer, 2005.
- [RW85] S. Roucos and A. Wilgus, “High quality time-scale modification for speech,” in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP’85.*, vol. 10. IEEE, 1985, pp. 493–496.
- [Sel11] I. Selesnick, “Wavelet transform with tunable q-factor,” *IEEE transactions on signal processing*, vol. 59, no. 8, 2011.
- [Ser89] X. Serra, *A system for sound analysis/transformation/synthesis based on a deterministic plus stochastic decomposition*. Department of Music, Stanford University, 1989, no. 58.
- [SK10] C. Schörkhuber and A. Klapuri, “Constant-q transform toolbox for music processing,” in *7th Sound and Music Conf*, 2010.
- [SS87] J. Smith and X. Serra, *PARSHL: an analysis/synthesis program for non-harmonic sounds based on a sinusoidal representation*. Citeseer, 1987.
- [SS90] X. Serra and J. Smith, “Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochas-

- tic decomposition,” *Computer Music Journal*, vol. 14, no. 4, pp. 12–24, 1990.
- [SVW94] G. Spleesters, W. Verhelst, and A. Wahl, “On the application of automatic waveform editing for time warping digital and analog recordings,” *PREPRINTS-AUDIO ENGINEERING SOCIETY*, 1994.
- [VE06] H. Viste and G. Evangelista, “A method for separation of overlapping partials based on similarity of temporal envelopes in multichannel mixtures,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 14, no. 3, pp. 1051–1061, 2006.
- [VHDG11] G. Velasco, N. Holighaus, M. Dörfler, and T. Grill, “Constructing an invertible constant-q transform with nonstationary gabor frames,” in *Proceedings of the 14th International Conference on Digital Audio Effects*, 2011.
- [Vir06] T. Virtanen, *Sound source separation in monaural music signals*. Tampere University of Technology, 2006.
- [VK00] T. Virtanen and A. Klapuri, “Separation of harmonic sound sources using sinusoidal modeling,” in *Acoustics, Speech, and Signal Processing, 2000. ICASSP’00. Proceedings. 2000 IEEE International Conference on*, vol. 2. IEEE, 2000, pp. II765–II768.
- [VK02] ———, “Separation of harmonic sounds using linear models for the overtone series,” in *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*, vol. 2. IEEE, 2002, pp. II–1757.
- [VR93] W. Verhelst and M. Roelands, “An overlap-add technique based on waveform similarity (wsola) for high quality time-scale modification of speech,” in *Acoustics, Speech, and Signal Process-*

*ing, 1993. ICASSP-93., 1993 IEEE International Conference on*, vol. 2. IEEE, 1993, pp. 554–557.

- [YB78] J. Youngberg and S. Boll, “Constant-q signal analysis and synthesis,” in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP’78.*, vol. 3. IEEE, 1978, pp. 375–378.
- [ZF99] E. Zwicker and H. Fastl, *Psychoacoustics: Facts and models*. Springer Berlin, 1999, vol. 254.