CONSTANT-Q TRANSFORM TOOLBOX FOR MUSIC PROCESSING

Christian Schörkhuber
Institute of Electronic Music and Acoustics
University of Music and Performing Arts, Graz
cschoerkhuber@student.tugraz.at

Anssi Klapuri[1]
Centre for Digital Music
Queen Mary University of London
anssi.klapuri@elec.qmul.ac.uk

ABSTRACT

This paper proposes a computationally efficient method for computing the constant-Q transform (CQT) of a time-domain signal. CQT refers to a time-frequency representation where the frequency bins are geometrically spaced and the Q-factors (ratios of the center frequencies to bandwidths) of all bins are equal. An inverse transform is proposed which enables a reasonable-quality (around 55dB signal-to-noise ratio) reconstruction of the original signal from its CQT coefficients. Here CQTs with high Q-factors, equivalent to 12–96 bins per octave, are of particular interest. The proposed method is flexible with regard to the number of bins per octave, the applied window function, and the Q-factor, and is particularly suitable for the analysis of music signals. A reference implementation of the proposed methods is published as a Matlab toolbox. The toolbox includes user-interface tools that facilitate spectral data visualization and the indexing and working with the data structure produced by the CQT.

1. INTRODUCTION

Constant-Q transform (CQT) here refers to a technique that transforms a time-domain signal $x(n)$ into the time-frequency domain so that the center frequencies of the frequency bins are geometrically spaced and their Q-factors are all equal. In effect, this means that the frequency resolution is better for low frequencies and the time resolution is better for high frequencies. The CQT is essentially a wavelet transform, but here the term CQT is preferred since it underlines the fact that we are considering transforms with relatively high Q-factors, equivalent to 12–96 bins per octave. This renders many of the conventional wavelet transform techniques inadequate; for example methods based on iterated filterbanks would require filtering the input signal hundreds of times.

The CQT is well-motivated from both musical and perceptual viewpoints. 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. From auditory perspective, the frequency resolution of the peripheral hearing system of humans is approximately constant-Q over a wide range from 20kHz down to approximately 500Hz, below which the Q-values get progressively smaller [1]. From perceptual audio coding, we know that the shortest transform window lengths have to be of the order 3ms in order to retain high quality, whereas higher frequency resolution is required to carry out coding at low frequencies [2]. All this is in sharp contrast with the conventional discrete Fourier transform (DFT) which has linearly spaced frequency bins and therefore cannot satisfy the varying time and frequency resolution requirements over the wide range of audible frequencies.

There are at least three reasons why the CQT has not widely replaced the DFT in audio signal processing. Firstly, it is computationally more intensive than the DFT. Secondly, the CQT lacks an inverse transform that would allow perfect reconstruction of the original signal from its transform coefficients. Thirdly, CQT produces a data structure that is more difficult to work with than the time-frequency matrix (spectrogram) obtained by using short-time Fourier transform in successive time frames. 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 this paper, we propose solutions to these three problems.

As already mentioned above, constant-Q transform can be viewed as a wavelet transform. The wavelet literature is well-matured (see e.g. [3]) 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 [4, 5]. 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 [6, 7]. 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 effi-

[1] Equally contributing authors.

cient technique for computing constant-Q transforms with high Q factors based on the fast Fourier transform (FFT) and a frequency-domain kernel [8, 9]. 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 [10]. However, this is not true for music signals in general.

This paper proposes specific solutions to the three problems of CQT mentioned above. Our solution to the computational efficiency problem is based on the technique proposed by Brown and Puckette in [9] which we extend to further improve 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, we propose interface tools for the data structure that facilitate working with the signal in the transform domain. A reference implementation of the proposed methods is provided as a Matlab toolbox at `http://www.elec.qmul.ac.uk/people/anssik/cqt/`.

2. SIGNAL MODEL

The CQT transform $X^{\mathsf{CQ}}(k,n)$ of a discrete time-domain signal $x(n)$ is defined by

$$X^{\mathsf{CQ}}(k,n) = \sum_{j=n-\lfloor N_k/2 \rfloor}^{n+\lfloor N_k/2 \rfloor} x(j) a_k^*(j-n+N_k/2) \quad (1)$$

where $k = 1, 2, \ldots, K$ indexes the frequency bins of the CQT, $\lfloor \cdot \rfloor$ denotes rounding towards negative infinity and $a_k^*(n)$ denotes the complex conjugate of $a_k(n)$. The basis functions $a_k(n)$ are complex-valued waveforms, here also called time-frequency *atoms*, and are defined by

$$a_k(n) = \frac{1}{N_k} w\left(\frac{n}{N_k}\right) \exp\left[-i2\pi n \frac{f_k}{f_\mathrm{s}}\right] \quad (2)$$

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. The window function is zero outside the range $t \in [0, 1]$.

The window lenghts $N_k \in \mathbb{R}$ in (1)–(2) are real-valued and inversely proportional to f_k in order to have the same Q-factor for all bins k. Note that in (1) the windows are centered at the sample n of the input signal. Different window functions will be discussed in Sec. 5.

In the CQT considered here, the center frequencies f_k obey

$$f_k = f_1 2^{\frac{k-1}{B}} \quad (3)$$

where f_1 is the center frequency of the lowest-frequency bin, and B determines the number of bins per octave. In practice, B is the most important parameter of choice when using the CQT, because it determines the time-frequency resolution trade-off of the CQT.

The Q-factor of bin k is given by

$$Q_k \stackrel{\text{def.}}{=} \frac{f_k}{\Delta f_k} = \frac{N_k f_k}{\Delta\omega f_\mathrm{s}} \quad (4)$$

where Δf_k denotes the -3dB bandwidth of the frequency response of the atom $a_k(n)$ and $\Delta\omega$ is the -3dB bandwidth of the mainlobe of the spectrum of the window function $w(t)$, being $\Delta\omega \approx 1.50$ [DFT bins] for the Hann window and $\Delta\omega \approx 1.73$ for Blackman, for example. The Q-factors Q_k are by definition the same for all bins, therefore we omit the subscript and write simply Q below.

It is typically desirable to make Q as large as possible, so as to make the bandwidth Δf_k of each bin as small as possible and thus introduce minimal frequency smearing. However, we cannot employ arbitrarily high Q factors – otherwise portions of the spectrum between the bins would not be analyzed. The value of Q that introduces minimal frequency smearing but still allows signal reconstruction is

$$Q = \frac{q}{\Delta\omega(2^{\frac{1}{B}} - 1)} \quad (5)$$

where $0 < q \lessapprox 1$ is a 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.

Substituting (5) in (4) and solving for N_k, we get

$$N_k = \frac{q f_\mathrm{s}}{f_k(2^{\frac{1}{B}} - 1)} \quad (6)$$

where we see that the dependency on $\Delta\omega$ has disappeared.

It is not computationally reasonable to calculate the coefficients $X^{\mathsf{CQ}}(k,n)$ at all positions n of the input signal. To enable signal reconstruction from the CQT coefficients, successive atoms can be placed H_k samples apart ("hop size"). In order to analyze all parts of the signal properly and to achieve reasonable signal reconstruction, values $0 < H_k \lessapprox \frac{1}{2} N_k$ are useful.

3. ALGORITHM FOR COMPUTING THE TRANSFORM

The computationally-efficient forward CQT transform proposed here is based on the principles proposed by Brown and Puckette in [9]. Therefore we first explain the technique proposed in [9] and then describe the extensions in Subsection 3.2.

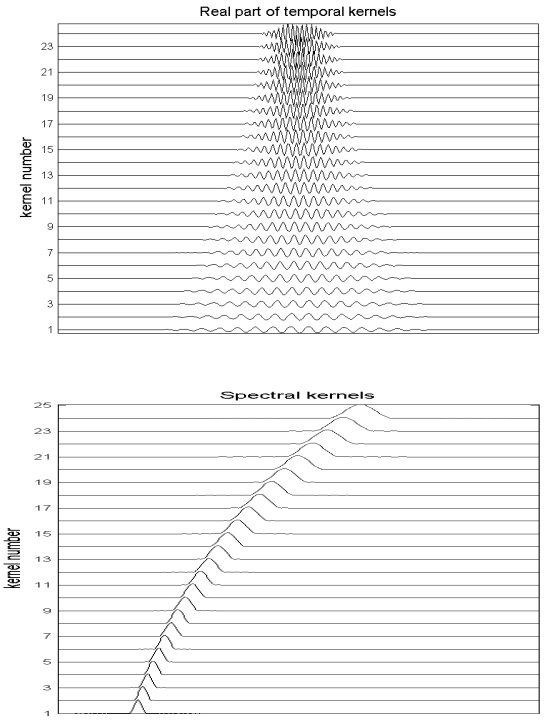

Figure 1. The upper panel illustrates the real part of the transform bases (temporal kernel) that can be used to calculate the CQT over two octaves, with 12 bins per octave. The lower panel shows the absolute values of the corresponding spectral kernel.

3.1 Algorithm of Brown and Puckette

Let us assume that we want to calculate the CQT transform coefficients $X^{\mathsf{CQ}}(k,n)$ as defined by (1) at one point n of an input signal $x(n)$. A direct implementation of (1) obviously requires calculating inner products of the input signal with each of the transform bases. The upper panel of Fig. 1 illustrates the real part of the transform bases $a_k(n)$, assuming here for simplicity only $B = 12$ bins per octave and a frequency range of two octaves.

A computationally more efficient implementation is obtained by utilizing the identity

$$\sum_{n=0}^{N-1} x(n)a^*(n) = \sum_{j=0}^{N-1} X(j)A^*(j) \tag{7}$$

where $X(j)$ denotes the discrete Fourier transform (DFT) of $x(n)$ and $A(j)$ denotes the DFT of $a(n)$. Equation (7) holds for any discrete signals $x(n)$ and $a(n)$ and stems from Parseval's theorem [3].

Using (7), the CQT transform in (1) can be written as

$$X^{\mathsf{CQ}}(k, N/2) = \sum_{j=0}^{N} X(j)A_k^*(j) \tag{8}$$

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. Following the terminology of [9], we will refer to $A_k(j)$ as the spectral kernels and to $a_k(n)$ as the temporal kernels. The lower panel of Fig. 1 illustrates the absolute values of the spectral kernels $A_k(j)$ corresponding to temporal kernels $a_k(n)$ in the upper panel.

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 (8) can be limited to values near the peak in the spectral kernel to achieve sufficient numerical accuracy – omitting near-zero values in $A_k(j)$. This is the main idea of the efficient CQT transform proposed in [9]. It is also easy to see that the summing has to be carried out for positive frequencies only, followed by multiplication by two.

For convenience, we store the spectral kernels $A_k(j)$ as columns in matrix $\mathbf{A}$. The transform in (8) can then be written in matrix form as

$$\mathbf{X}^{\mathsf{CQ}} = \mathbf{A}^*\mathbf{X} \tag{9}$$

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

3.2 Processing One Octave at a Time

There are two remaining problems with the method outlined in the previous subsection. Firstly, when a wide range of frequencies is considered (for example, eight octaves from 60Hz to 16kHz), quite long DFT transform blocks are required and the spectral kernel is no longer very sparse, since the frequency responses of higher frequency bins are wider as can be seen from Fig. 1. 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, where N_K is the window length for the highest CQT bin. Both of these factors reduce the computational efficiency of the method.

We propose two extensions to address the above problems. The first is processing by octaves. [2] We use a spectral kernel matrix $\mathbf{A}$ which produces the CQT for the highest octave only. After computing the highest-octave CQT bins over the entire signal, the input signal is lowpass filtered and downsampled by factor two, and then the same process is repeated to calculate the CQT bins for the next octave, using exactly the same DFT block size and spectral kernel (see (8)). This is repeated iteratively until the desired number of octaves has been covered. Figure 2 illustrates this process.

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

[2] We want to credit J. Brown for mentioning this possibility already in [8], although octave-by-octave processing was not implemented in [8, 9].

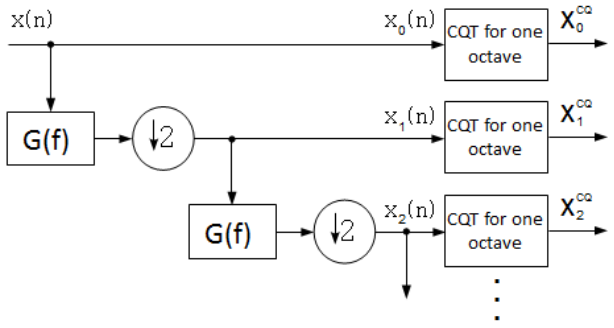

Figure 2. An overview of computing the CQT one octave at the time. Here $G(f)$ is a lowpass filter and $\downarrow 2$ denotes downsampling by factor two.

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 (8) 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.

Figure 3 illustrates the general structure of the kernel matrix applied in this paper. 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. Similarly, the two lowest kernel functions correspond to the same frequency, but different temporal locations. The detailed structure of the spectral kernel will be discussed in Sec. 5; 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 Fig. 2) 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_{DFT} which are positioned H_{DFT} samples apart (i.e., successive frames overlap by $N_{\mathsf{DFT}} - H_{\mathsf{DFT}}$ 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 $\mathbf{X}_d^{\mathsf{CQ}}$ for this octave d is calculated as

$$\mathbf{X}_d^{\mathsf{CQ}} = \mathbf{A}^* \mathbf{X}_d \tag{10}$$

where $\mathbf{A}^*$ is the conjugate transpose of the complex-valued spectral kernel matrix for one octave as described above. The column m of $\mathbf{X}_d^{\mathsf{CQ}}$ contains the CQT coefficients representing DFT block m and the different rows of $\mathbf{X}_d^{\mathsf{CQ}}$ 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 Fig. 2. Note that the kernel remains the same for all octaves. Also, the DFT length N_{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, $x(n)$.

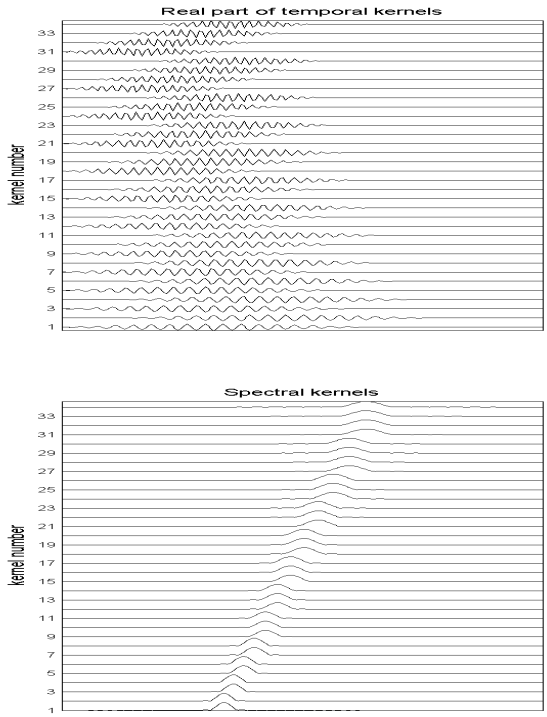

Figure 3. Illustration of the general structure of the kernel matrices used in this paper. The upper panel shows the real part of the temporal kernel used to compute the CQT for one octave. The lower panel shows the absolute values of the corresponding spectral kernel.

The decimated signals $x_d(n)$ are obtained from $x_{d-1}(n)$ by lowpass filtering and downsampling by factor two. For the lowpass filter $G(f)$, we use zero-phase forward-and-reverse filtering with a sixth-order Butterworth IIR filter that has a cut-off frequency $f_s/4$. 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 4 shows the magnitude response of the lowpass filter $G(f)$ (square of the magnitude response of sixth-order Butterworth filter). Downsampling 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 $x(n)$. We address this problem by padding $2^{D-1}N_1$ zeros at be beginning of the signal and $2^{D-1}N_{\mathsf{DFT}}$ 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_{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

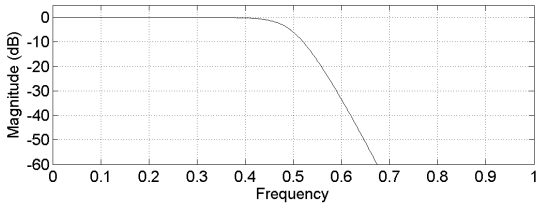

Figure 4. Magnitude response of the lowpass filter $G(f)$.

zero padding is done separately for each octave, but this would make the implementation less clear and therefore we assume that the number of zeros padded is negligible in comparison to the length of the input signal $x(n)$.

3.3 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 $\lfloor (L - N_{\mathsf{DFT}})/H_{\mathsf{DFT}} \rfloor + 1$. For the next octave, the number of fast Fourier transforms (FFTs) is roughly twice smaller, $\lfloor (L/2 - N_{\mathsf{DFT}})/H_{\mathsf{DFT}} \rfloor + 1$, in fact a bit more than twice smaller. For the next octave, the number of DFT transforms is rought four times smaller, and so forth. Since $1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots \approx 2$, the total number C of FFTs to compute is

$$C \leq 2\,(\lfloor (L - N_{\mathsf{DFT}})/H_{\mathsf{DFT}} \rfloor + 1) \qquad (11)$$

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 (10)). 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 8). The number of lowpass 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 [9], 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 (9) 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.

4. INVERSE CQT TRANSFORM

Figure 5 shows 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 $\mathbf{X}_d^{\mathsf{CQ}}$. The process is analogous to the forward transform, except that all is done in reverse order.

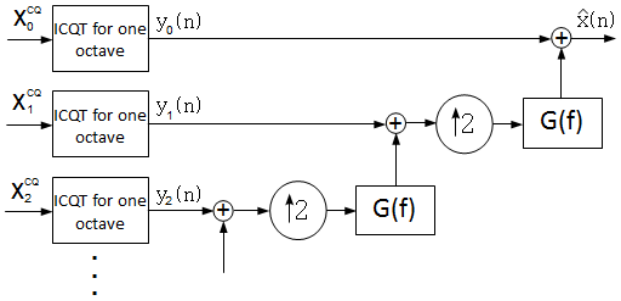

Figure 5. An overview of the inverse CQT transform (ICQT), where an approximation $\hat{x}(n)$ of the input signal $x(n)$ is reconstructed from the octave-wise CQT coefficient matrices $\mathbf{X}_d^{\mathsf{CQ}}$.

The block indicated by "ICQT for one octave" in Fig. 5 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:

$$\mathbf{Y}_d = \mathbf{V}^* \mathbf{X}_d^{\mathsf{CQ}} \qquad (12)$$

where the column m of $\mathbf{Y}_d$ contains the complex-valued DFT approximating the column m of $\mathbf{X}_d$ in (10), 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 Sec. 5: we use kernels for which $\mathbf{V} = \mathbf{A}^*$, meaning that the inverse kernel is a conjugate transpose of the forward transform kernel. [3]

Since each column of $\mathbf{Y}_d$ only contains the DFT spectrum for the positive frequencies, 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.

The 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 $(d+1, d+2, \ldots, D-1)$ in order to obtain a signal $\hat{x}_d(n)$ that approximates the input signal for octaves $d, d+1, \ldots, D-1$. The signal $\hat{x}_d(n)$ is then upsampled by factor two by inserting zeros between the original samples, multiplying the signal by two, and lowpass filtering using zero-phase forward-and-reverse filtering with a sixth-order Butterworth IIR filter having cut-off frequency $f_{\mathrm{s}}/4$ (the same that was used at the analysis stage).

The above process is repeated for each octave at a time, as illustrated in Fig. 5. After reconstructing all the octaves, $d = 0, 1, \ldots, D-1$, the resulting signal $\hat{x}_0(n) \equiv \hat{x}(n)$ is an approximate reconstruction of the input signal $x(n)$.

[3] Note that then $\mathbf{Y}_d = \mathbf{V}^* \mathbf{X}_d^{\mathsf{CQ}} = \mathbf{A}\mathbf{A}^* \mathbf{X}_d$, where multiplication by $\mathbf{A}\mathbf{A}^*$ actually implements a near-perfect one-octave bandpass filter.

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 we use $\mathbf{V} = \mathbf{A}^*$, the inverse kernel is sparse too.

5. KERNEL DESIGN

As already mentioned in Sec. 3.2, we use a transform kernel that contains frequency bins over one octave range, and several time-shifted atoms for each frequency bin k. These time-shifted atoms should cover the input signal over the H_{DFT} samples between the beginning of DFT frame m and the beginning of the next frame, $m + 1$.

5.1 General Considerations

It is natural to expect that all samples of the input signal $x(n)$ have an equal weight in the transform, and therefore to require that successive window functions $w(n)$ for bin k sum up to approximately unity over the entire signal. In the case of an analysis-synthesis system (CQT followed by inverse CQT), the *squares* of successive window functions, $[w(n)]^2$, have to sum to unity. This is 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. Applying windowing at the synthesis (ICQT) stage is necessary in order to avoid audible artefacts if the signal is manipulated in the CQT transform domain. If no processing takes place in the transform domain, the above requirement (that $[w(n)]^2$ sum to unity) leads to a high-quality reconstruction.

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.

Most window functions (e.g. Hann and Hamming) 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_{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_{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_{DFT} can be made large relative to atom sizes N_k, and thereby H_{DFT} approximately divisible by $\frac{1}{z}N_k$ for all k. This approach leads to a reasonable quality results. However, due to the fact that such a kernel contains a different number of atoms for each bin, accessing and manipulating the CQT coefficients in the transform domain becomes slightly more complicated numerically, and the quality of the reconstruction degrades since neighbouring atoms are not temporally synchronized. Also, sparsity of the spectral kernel $\mathbf{A}$ suffers since DFT frame size is large relative to the atom sizes N_k.

5.2 Proposed Kernel Structure

Due to the above reasons, we propose a kernel, where the atoms *within each octave* are synchronized in the sense that they are centered at the same, successive, points in time. In this case, the temporal ripple can be minimized by using a window function that roughly sums up to unity for a wide range of overlap values. Blackman and Blackman-Harris windows are particularly suitable here: provided that successive windows overlap at least by 66% and 75% for Blackman and B.-Harris windows, respectively, the exact amount of overlap is not important, but successive windows sum up to approximately a constant value which can be normalized to unity. Using such windows it is now possible to use the same number of temporal atoms for each bin by defining a common hop size H_{ATOM} for all atoms in the kernel. The relative hop size H_{ATOM}/N_k will thus vary accross bins k, but only up to the factor two between the smallest and the largest atom in the one-octave kernel.

The parameter H_{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_{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_{ATOM} is easy to calculate: we recommend using $H_{\mathsf{ATOM}}/N_K \approx \frac{1}{3}$ or $\frac{1}{4}$, where N_K denotes the lenght of the shortest atom within the one-octave kernel. This leads to redundancy factors around five and quality that is near to the optimal (around 55dB).

It should be noted that although the atoms are centered at same temporal positions within each octave, at the next-lower, octave, the atoms are centered at only every second of these temporal positions, due to the decimation by factor two before processing the next octave.

The number of temporally shifted atoms within the one-octave kernel is the same for all bins k in the above-described approach. In practice, we choose a DFT frame length that is the next power of two of the largest atom within the octave, and then populate the kernel with as many temporally shifted atoms as there fit. The DFT frame hop H_{DFT} is then chosen accordingly. This leads to the most sparse spectral kernel $\mathbf{A}$ and therefore the fastest implementation.

To understand why neighbouring bins sum up to unity over frequency (and not just over time), note that within each octave, neighbouring bins are centered at same point and their lengths N_k are only slightly different, assuming $B > 12$ bins per octave. As a result, the neighbouring bins perform sampling of time-frequency domain that is *locally* very similar to that of the DFT, and similarly to the DFT, sum up to an almost perfectly flat frequency response.

6. REDUNDANCY

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

$$R = \frac{2C_{\mathsf{CQT}}}{C_{\mathsf{IN}}} \tag{13}$$

where C_{CQT} and C_{IN} denote the amount of CQT coefficients and the amount of samples in the input signal, re-

spectively. 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 $C_{\mathsf{OCT}} = C_{\mathsf{IN}}B/(hN_K)$, where $h = H_{\mathsf{ATOM}}/N_K$ is the atom hop size relative to the length N_K of the shortest atom (hiqhest-frequency bin). Substituting the length N_K from (6), we get

$$C_{\mathsf{OCT}} = \frac{C_{\mathsf{IN}} f_K B(2^{\frac{1}{B}} - 1)}{hqf_{\mathrm{s}}} \approx \frac{0.7 C_{\mathsf{IN}} f_K}{hqf_{\mathrm{s}}} \qquad (14)$$

where the latter approximation is obtained by noting that $B(2^{\frac{1}{B}} - 1) \approx 0.7$ when $B \geq 12$. Here we have assumed that the number of bins per octave $B \geq 12$.

Since the length of the input signal decreases by the factor of two at each decimation, it is easy to see from (14) 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 $C_{\mathsf{OCT}}(1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots) \approx 2C_{\mathsf{OCT}}$. Substituting this to (13), the overall redundancy of the CQT transform is

$$R = \frac{2 \times 2 \times C_{\mathsf{OCT}}}{C_{\mathsf{IN}}} = \frac{2.8 f_K}{hqf_{\mathrm{s}}}. \qquad (15)$$

Here we can see that 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 5).

7. INTERFACE TOOLS FOR THE CQT DATA

In the described kernel structure, temporal positions where $X^{\mathsf{CQ}}(k, n)$ is calculated are the same for all bins within one octave (although the actual time resolution of course decreases from the highest to the lowest bin since the atom lengths vary). Moving down to the lower octaves, however, the number of points where $X^{\mathsf{CQ}}(k, n)$ is evaluated decreases by factor two at each octave, and therefore the number of time points where $X^{\mathsf{CQ}}(k, n)$ is evaluated is not the same for all bins from the lowest frequency bin (of the lowest octave) to the highest bin (of the highest octave).

In order to allow the user an easy access to the information without minding 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 "rasterised" CQT data structure is achieved by data interpolation between the time points $X^{\mathsf{CQ}}(k, n)$ that have been computed by the CQT. With the interface tools, the user can obtain the entire CQT matrix representing the input data, or access only extracts of it. It is also possible to access only a certain time slice n of the CQT transform $X^{\mathsf{CQ}}(k, n)$ 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 $X^{\mathsf{CQ}}(k, n)$ in a form similar to the DFT spectrogram using the described interpolation technique. Figure 6 shows the CQT transform of a four-second music excerpt containing singing, acoustic guitar, bass, and synthesizer sounds. More examples can be found online at the URL given in Introduction.

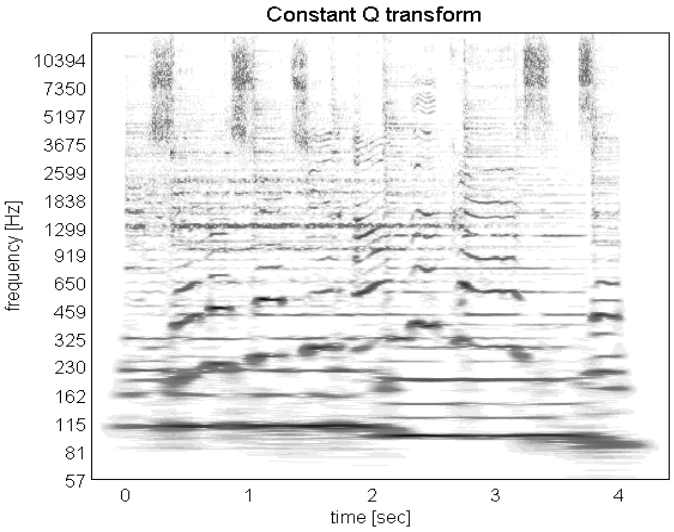

Figure 6. CQT transform of a music excerpt containing singing, acoustic guitar, bass, and synthesizer sounds.

8. RESULTS

Figure 7 shows the quality of the reconstructed time-domain signal $\hat{x}(n)$ as a function of the redundacy R (see (13)) 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 hopsize 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 $\approx$ 3dB 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 analysed: we used $f_K = f_{\mathrm{s}}/3 = 14.7$kHz for the highest CQT bin, and analyzed eight octaves down to 57Hz. Random noise represents a "worst case": for music signals, the reconstruction quality is typically a few decibels better. Redundancy factors were calculated by substituting $f_K = 14.7$kHz and $f_{\mathrm{s}} = 2 \times 14.7$kHz into (15), where the latter is the sampling rate required to represent the time domain signal up to 14.7kHz. [4]

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

$$\mathrm{SNR} = 10 \log_{10} \frac{\sum_n [x(n)]^2}{\sum_n [\hat{x}(n) - x(n)]^2} \qquad (16)$$

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

[4] Note that if an input signal is to be analyzed up to the Nyquist frequency ($f_K = f_{\mathrm{s}}/2$), the input signal has to be slightly upsampled (say, $f'_{\mathrm{s}} = \frac{4}{3} f_{\mathrm{s}}$) before applying the proposed method, since the lowpass filter $G(f)$ in Fig. 4 is not ideal.

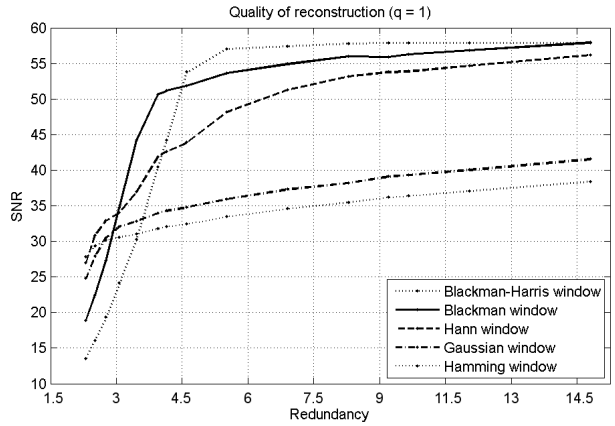

Figure 7. Quality of the reconstructed signal as a function of the window function $w(n)$ and the redundancy R.

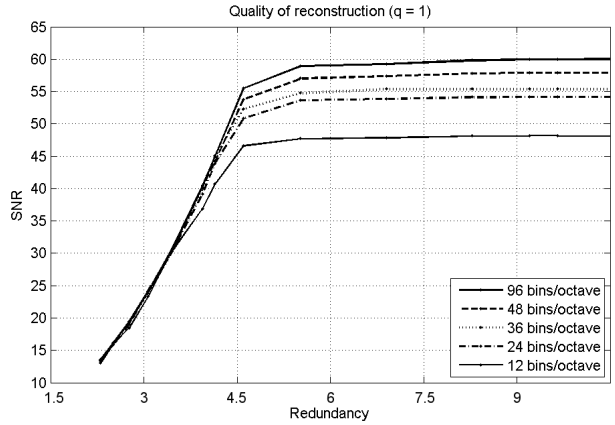

Figure 8. Quality of the reconstructed signal as a function of the number of bins per octave, B, and redundancy R.

$R > 4.5$ the Blackman-Harris window 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 7 shows that using Blackman-Harris window, SNR values of about 55dB are achieved with $R \approx 5$.

Fig. 8 shows the quality of the reconstructed time-domain signal $\hat{x}(n)$ as a function of the redundacy R (see (13)) 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.

9. ACKNOWLEDGEMENT

Thanks to Wen Xue from Queen Mary University of London for constructive comments on a draft of this paper.

10. CONCLUSIONS

Computationally efficient methods were proposed for computing the CQT and its inverse transform. The proposed techniques lead to a reasonable-quality reconstruction (around 55dB) of an input signal from its CQT coefficients while requiring only moderate redundancy (by factor four or five) in the CQT representation. A reference implementation of the methods is provided as a Matlab toolbox. It is hoped to be useful for several applications, including sound source separation, music signal analysis, and audio effects.

11. REFERENCES

[1] B. C. J. Moore, ed., *Hearing—Handbook of Perception and Cognition*. 2nd ed., 1995.

[2] ISO/IEC, "Information technology – Generic coding of moving pictures and associated audio information – Part 7: Advanced audio coding (AAC)," Tech. Rep. 13818-7:2006, ISO/IEC, 2006.

[3] S. Mallat, *A wavelet tour of signal processing*. Academic Press, third ed., 2009.

[4] 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.

[5] J. Kovacevic and M. Vetterli, "Perfect reconstruction filter bankds with rational sampling factors," *IEEE Trans. on Signal Processing*, vol. 41, pp. 2047–2066, 1993.

[6] 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.

[7] 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.

[8] J. C. Brown, "Calculation of a constant Q spectral transform," *J. Acoust. Soc. Am.*, vol. 89, no. 1, pp. 425–434, 1991.

[9] J. C. Brown and M. S. Puckette, "An efficient algorithm for the calculation of a constant Q transform," *J. Acoust. Soc. Am.*, vol. 92, no. 5, pp. 2698–2701, 1992.

[10] D. FitzGerald, M. Cranitch, and M. T. Cychowski, "Towards an inverse constant Q transform," in *120th Audio Engineering Society Convention*, (Paris, France), 2006.

[11] A. Makur and S. K. Mitra, "Warped discrete-Fourier transform: Theory and applications," *IEEE Trans. Circuits and Syst.-I:Fundd. Theor. Appl.*, vol. 48, no. 9, pp. 1089–1093, 2001.