THE MODIFIED CONSTANT Q SPECTROGRAM (MCQS) AND ITS APPLICATION TO PHASE VOCODING

by
Atul Ingle

A thesis submitted in partial fulfillment of
the requirements for the degree of

Master of Science
(Electrical and Computer Engineering)

at the
UNIVERSITY OF WISCONSIN-MADISON
2011

Acknowledgements

I take this opportunity to express my sincere gratitude toward my advisor Prof. William Sethares for his guidance and whole hearted support throughout this project. He has been a constant source of motivation over the past year and a half and our interactions have helped me immensely in all aspects of this project. His insight and ingenuity in solving problems related to audio signal processing has not only broadened my understanding of this area but has also changed the way I think about engineering problems in general.

I am grateful to Prof. John Gubner for his helpful suggestions on solving least-squares problems. I am also thankful to Prof. James Bucklew for guidance over the last two years, as my academic advisor.

Atul Ingle
May 13, 2011
Madison, WI

Abstract

This thesis discusses the development of a modified constant Q spectrogram representation which is invertible in a least-squares sense. A good quality inverse is possible because this modified transform method, unlike the usual sliding window constant Q spectrogram, does not discard any data samples when performing the variable length discrete Fourier transforms on the original signal. The development of a phase vocoder application using this modified technique is also discussed. It is shown that MCQS phase vocoding is not a trivial extension of the regular FFT-based phase vocoder algorithms and some of the mathematical subtleties related to phase reconstruction are addressed.

Contents

List of Figures

Chapter 1

Introduction

The conventional method of analyzing signals in the frequency domain has been through the use of the Fourier Transforms for analog signals. For a time domain signal, $x(t)$, the following transform-inversion pair of equations is commonly used,

$$FT(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt \tag{1.1}$$

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} FT(\omega)e^{j\omega t}d\omega. \tag{1.2}$$

Here $FT(\omega)$ denotes the Fourier Transform of the signal $x(t)$ evaluated at an angular frequency ω and t denotes time. Notice that the transform Equation (1.1) is simply a "correlation" of a complex exponential of a certain frequency ω with the given time signal, averaged out for all times. The inversion Equation (1.2) is the reverse operation that synthesizes the time signal by "weighted averaging" of the correct frequencies formed by a collection of complex exponentials.

For a discrete time signal of length N, the Discrete Fourier Transform (DFT) is represented via the following transform-inversion equation pair,

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, \quad k = 0, ..., N-1 \tag{1.3}$$

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j2\pi kn/N}, \quad n = 0, ..., N-1. \tag{1.4}$$

The interpretation is analogous to the continuous time case, except that n denotes the sampled time index and k is the frequency bin index.

One of the drawbacks of this method arises when one attempts to use the Fourier Transform to get a spectrogram representation for audio signals that vary considerably over time. A spectrogram is a time-varying portrait of the frequency spectrum which is used for localizing information in the time-frequency plane. This representation is a very useful tool in audio analysis and visualization and allows, for instance, the detection of the specific note being played at a particular time or the pitch of a singer or an instrument. Information is localized by picking out a section of the time signal, often by "windowing" it with a window function that tapers off to zero at its ends and taking the Fourier Transform of this windowed segment. This leads to a tradeoff analogous to the Heisenberg's uncertainty principle — using shorter windows gives good time localization but poor frequency resolution, using longer windows gives good frequency resolution but blurs the time resolution. Various methods can be employed to bypass this problem; such as the use of variable windows and the Wavelet Transform. Variable length windows are fundamental to the development

of the constant Q transform [1].

There is an additional structure to audio signals that is often seen especially in the case of music signals. Most of the sound energy is localized around frequency bands that are geometrically spaced in ratios of powers of 2. This is known as the octave structure of musical notes and forms the basis of the note scale in Western music. Furthermore, psychoacoustic experiments have revealed that the response of the human ear to sound is more or less constant Q [2]. Hence what may appear to humans as equispaced frequencies, are really equispaced on a log-frequency axis. It is not immediately clear if the $\log(\omega)$ behavior can be captured using a regular Fourier Transform. As discussed later, the solution to this problem is harder than just stretching linear data so it fits on a log-scale.

Many audio editing and effect-insertion techniques operate by modification of certain aspects of the spectrogram obtained from the original sound signal. The altered sound is then synthesized by inverting this edited spectrogram back to the time domain. Spectrogram magnitudes have an easy interpretation, however, the phase values are harder to control and alter. The phase vocoder builds phase values back into the edited spectrogram such that the peaks in adjacent spectral frames connect to each other smoothly. For instance, it maintains frequencies by adjusting the phases in time proportional to the particular frequency. It keeps continuity between audio segments by smoothening out abrupt variations in phase that may otherwise lead to clicks or discontinuities.

Traditionally phase vocoders have been designed to operate on FFT-based spectrograms that have a linear frequency axis. However, it is desirable to have a spectrogram editing technique that is capable of operating on the constant Q spectrogram structures that obey logarithmic frequency spacing. It will not be far-fetched to expect that such editing techniques may produce results that sound better than those obtained from techniques that use linear frequency spacing. As it will become clear in Chapter 4, the constant Q phase vocoder is not a trivial extension of the FFT-based phase vocoder and there are some mathematical and algorithmic subtleties involved in its implementation. This thesis resolves some of these subtleties.

Chapter 2

Literature Review

2.1 Constant Q Transform

Probably the first work to formalize the idea of a constant Q transform for capturing the octave structure of music in a transform was introduced by Judith Brown in 1991 [1]. This transform method uses variable window lengths where each window is "tuned" to a particular frequency characterized by the Q factor. The Q factor can be understood as the ratio of the center frequency corresponding to the window length to the frequency resolution. This transform has a major drawback, however, in that it is not exactly invertible due to both temporal and frequency decimation. Nevertheless, the present work owes a lot to the methods proposed in [1] by building on the idea of variable length windowing.

In a sequel, Brown discusses an efficient algorithm for calculation of the constant Q transform [3]. The method uses Parseval's relation to switch to an equivalent set up in the frequency domain and then harnesses

the symmetry properties of this frequency domain interpretation of the transform equation to speed up calculation. It also capitalizes on the fact that many of the window coefficients are nearly zero and hence very few multiplications are required in reality.

The MCQS introduced here takes a different approach by setting up the transformation as a linear operator that produces the vectorized spectrogram directly from the time domain signal. This operation uses a matrix that has a large number of zeroes, which may provide a computational vantage point as that in the efficient algorithm in [3].

Other authors over the years have proposed modifications to the original constant Q transform formulation where all the variable length windows are aligned to the leftmost sample in the given time domain signal. For instance Bradford et al [4] suggest changing the location of these windows so that they either align to the last sample or the center sample of the signal. Aligning to the center is more meaningful because then the windows analyze the "same part of the signal.". It is natural to assume that information is localized with reference to the center of the window. This idea is used when setting up the MCQS. However, the method in [4] suffers from the same invertibility issues as the original constant Q transform due to temporal decimation.

Interestingly, the idea of using variable windowing when evaluating the short-time Fourier transform (STFT) also arises in areas other than signal processing. Stockwell [5] proposed the use of Gaussian windows

of widths inversely proportional to the frequency when calculating the continuous time STFT. This was inspired by a problem in geophysics and seismology and is called the S-Transform. In contrast to this method, the MCQS is not limited to any specific window type. It is targeted toward audio applications where capturing the geometric spacing between frequencies is important, a requirement that is not a crucial part of the original formulation of the S-Transform. Moreover, the phase adjustment strategy in the MCQS phase vocoder rests on the fact that the frequencies are geometrically spaced thereby allowing an elegant approximation for the phase values. This is further explored in Chapter 4.

Many authors have tried to answer the question of invertibility of the constant Q transform. A very early paper by Gambardella [7] observes the similarities between the constant Q transform and the Mellin transform in continuous time and argues that the inversion formula is given by the Mellin transform inversion relation. However, this work only focuses on the long time constant Q transform and makes no comments on the invertibility of the short-time transform that is of interest when localizing information in both frequency and time.

FitzGerald et al [6] take an optimization approach for inverting the constant Q transform. Instead of obtaining the time signal directly, they first propose taking an intermediate step of generating the DFT. Their method rests on the observation that in certain cases, when the constant Q transform is mapped to the DFT domain, it results in a sparser representation of the signal than in time domain. This holds for most audio signals

containing only pitched instruments. Therefore some of the compressed sensing techniques of sparse vector reconstruction via ℓ_0 and ℓ_1 norm minimization can be used to reconstruct the DFT from the constant Q transform vector. Notably, this work points out that a simple pseudo-inverse method fails to give a good inverse in the classical constant Q case due to the time-decimation issue. In this thesis it is shown that for the MCQS, the problem of temporal decimation is avoided by analyzing all time samples with every window. As a result, taking the compressed sensing approach is not required and the standard pseudo-inverse technique can be used to get acceptable quality of reconstruction. Moreover, the sparsity assumption for the DFT may not hold for all types of music signals. The MCQS inversion technique bypasses this limiting assumption on the structure of the DFT of the underlying signal.

There has been growing interest in the utility of the constant Q transform as a particularly attractive way of approaching problems of detecting structures in audio recordings. For instance, Smaragdis proposes a relative pitch tracking algorithm [8], that operates on a constant Q spectrogram rather than a normal STFT spectrogram because it brings out the octave spacing of notes more clearly. This algorithm locates peaks in the constant Q spectrogram and tracks the pitch using a modified EM algorithm. This is an interesting application which can potentially be extended to operate on the MCQS instead of the regular constant Q spectrogram.

Recognizing the importance of this transform method for audio analysis, Schorkhuber and Klapuri [10] recently proposed an efficient computational

toolbox for music analysis built using the constant Q idea. They further improve the computational efficiency of the transform calculation algorithm and also develop a better quality reconstruction technique. First, they propose the idea of processing by octaves where each octave is analyzed and the signal is downsampled by a factor of 2 for analyzing the next lower octave. This avoids the use of very wide windows at low frequencies. The transform for an octave is formulated as a matrix operation using a spectral kernel. Inversion is done by reversing these steps, first using the inverse matrix operator (they call it the inverse spectral kernel) followed by up sampling at each lower octave. The idea of obtaining the transform using a matrix operator is used for the MCQS too. However, the exact formulation is different in order to enable analysis of all time sample using every window.

2.2 Phase Vocoder

The name "phase vocoder" or "phase voice-coder" is a remnant of an algorithm from the late 1930s [11] of encoding voice for analysis and recovering it back from the analyzed encoding. Today the term phase vocoder is used for any technique that is capable of operating on the magnitude and phase values in a time-frequency representation and able to reconstruct a meaningful audio signal from the modified spectrogram. Flanagan and Golden [12] were one of the earliest to propose a method phase vocoding similar to the one used today. They describe a continuous-time version of a phase vocoder that analyzes speech signals using short time phase and magnitude spectra.

A more modern interpretation of this method appears in a tutorial by Dolson (1986) [13]. Two mathematically equivalent interpretations of phase vocoding are discussed in this tutorial – the filter bank interpretation and the Fourier transform interpretation. The filter bank interpretation analyses each frequency slice, one at a time, using many band pass filters. On the other hand, the Fourier interpretation depicts the phase vocoder as operating on the Fourier transforms of windowed time slices. This latter interpretation is used for developing the MCQS phase vocoder algorithm.

From the point of view of practical implementation of a phase vocoder to get audibly satisfactory results, Laroche and Dolson published two studies [15], [16] on the behavior of various phase assignment techniques in the phase vocoding algorithm. These methods are based on the underlying observation that phase values must be adjusted in a way that is consistent with the corresponding frequency and the time difference between adjacent windows in order to maintain phase continuity throughout the duration of the audio signal. Adapting these ideas in the MCQS phase vocoder is complicated by the fact that the frequencies are spaced geometrically instead of the usual linear scale in a DFT.

Puckette (1995) suggests a "phase locking" technique where the assumption is that the phase values at various DFT bins are locked in some definite way to the phase at the bin where the magnitude has a peak [19]. This can be explained based on the phase profiles of Fourier transforms of commonly used window functions that taper at the ends. This idea forms

the core of the phase assignment strategy presented in this thesis and it will be seen that some mathematical insight is needed to characterize the phase locking behavior in the constant Q case.

The Matlab implementation of the MCQS draws some of the ideas from Moller-Nielson (2002) [17] and Sethares (2007) [20] where some of the practical implementation issues in regular FFT/STFT based phase vocoders are discussed.

Chapter 3

Modified Constant Q Spectrogram

3.1 Review of Constant Q Transform

The constant Q transform [1] overcomes the problem associated with log-frequency transformation of STFT data. Instead of calculating the transform on a linear frequency scale, the calculation is performed on selected frequencies in the first place. For music signals, the choice of frequencies is naturally the frequencies in the equitempered scale. These frequencies begin at the note C_0 which is about 16.35 Hz followed by its multiples in powers of $2^{\frac{1}{12}}$. A set of 12 such frequency points from some frequency f up to $2f$ completes one musical octave. An octave therefore has twelve frequency "bins" of interest. In practice, however, for obtaining more data on the frequency axis, analysis can be done with 24 or even 36 bins per octave to yield 2 or 3 frequency components per musical note respectively.

A fixed length DFT gives constant resolution at all frequencies. For

instance, a window that is 1024 samples long and is sampled at 44.1 kHz (which is typical "CD-quality" sampling rate) gives a frequency resolution of about 43.1 Hz. This is too large to detect the difference between, say, the notes C_0 and $C_0^\sharp$ that are separated by only 1 Hz. On the other hand, this resolution is wasteful at high frequencies; for instance, the notes C_7 and $C_7^\sharp$ are separated by 124.5 Hz.

A more parsimonious approach is to maintain a constant ratio of center frequency to frequency resolution by choosing different window lengths at different frequencies. This ratio is essentially the "Q" of the transform. As an example, choosing 48 analysis bins in each octave gives a variable resolution of $(2^{\frac{1}{48}} - 1) = 0.0145$ times the center frequency which gives $Q = f/\delta f = f/0.0145f = 67$. In general, if λ is the number of bins per octave,

$$Q = \frac{1}{2^{1/\lambda} - 1}. \tag{3.1}$$

The correct length for each DFT can be found as follows. Assume a sampling rate of f_s. Any frequency f_k associated with a resolution δf_k requires a DFT window length $N_k = f_s/\delta f_k = Q\ f_s/f_k$. The transformation method reduces to finding the N_k-long DFT for each frequency, f_k, of interest and then picking out the Q^{th} DFT coefficient (where Q is suitably rounded to the nearest integer). Mathematically,

$$X_k = \frac{1}{N_k} \sum_{n=0}^{N_k - 1} w(k,n) x(n) e^{\frac{-j2\pi Qn}{N_k}} \tag{3.2}$$

Here $x(\cdot)$ is the original digital data sequence and $w(\cdot,\cdot)$ is a suitable window function. An efficient way to calculate this using FFTs is discussed in [3].

From Equation (3.2) it is clear that the transform does not analyze all data samples at high frequencies where the window length is small. Hence, the constant Q transform is not invertible. For example, consider a hypothetical[1] situation shown in Figure (3.1) that shows the window placement when calculating a 6-point CQT of the underlying signal piece.

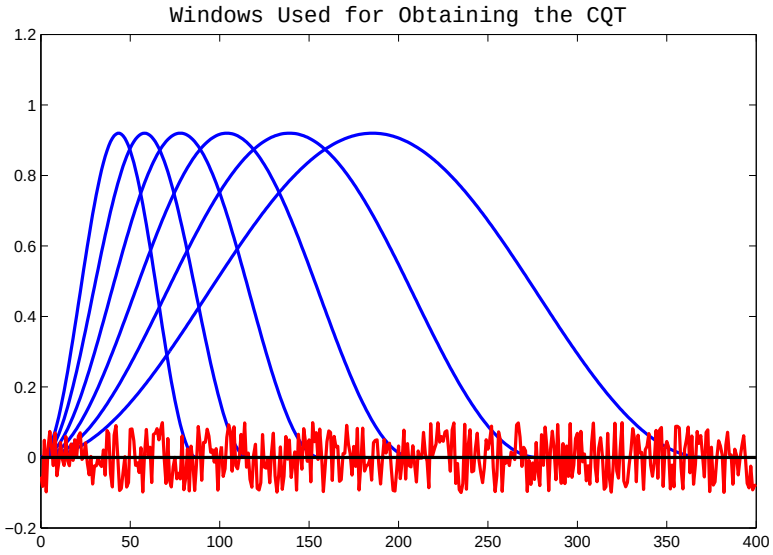

Figure 3.1: Windows used in calculating a hypothetical 6-point CQT

A constant Q spectrogram is obtained by stacking together columns of constant Q transforms of adjacent pieces of the time signal. This is analogous to the STFT spectrogram that is formed by stacking columns of DFTs. To localize frequency information at a particular time, the constant Q

[1]The situation is hypothetical because a 6-point CQT has too little information to be of any practical use!

spectrogram picks out a small piece (called a "time-slice") from the given signal and takes the constant Q transform of this slice. For this method to work, there must be at least as many samples in the time-slice as the longest window that is used when calculating the constant Q transform. It may also be desirable to use a certain fraction of overlap between adjacent pieces, as done with the STFT spectrogram.

Exact interpretation of the adjacent window overlap factor is a complicated issue in the constant Q spectrogram. Note that there are two kinds of windowing operations occurring in the evaluation of the spectrogram. The first operation is that of picking out a time-slice — this can be thought of as using a rectangular window around the time instant of interest. For the spectrogram, the overlap can be interpreted as referring to the overlap between pieces of the signal that are extracted for constant Q analysis. The next windowing operations occur when the constant Q transform of this piece of the signal is calculated using variable length windows aligned at the left edge as shown in Figure (3.1). Hence, in reality, there is a variable amount of overlap between the actual analysis windows. The longer windows will have a larger overlap fraction whereas the smallest windows may not overlap at all depending on how the adjacent signal pieces were chosen. This issue can be resolved through the MCQS as described in the next section.

3.2 The Modified Constant Q Spectrogram

As described in Section 3.1, the original constant Q formulation does not analyze all samples in the data sequence. Moreover, the interpretation of window overlap fraction is complicated due to the use of variable length windows. The modified constant Q spectrogram (MCQS) method uses a sliding window with a preset percentage overlap between adjacent windows to obtain the spectrogram directly from a given data sequence. This is unlike familiar spectrogram procedures that stack columns of a frequency domain transform of pieces of the time signal. In the MCQS, for every frequency of interest, there is an associated window length specified by the constant Q. Since higher frequencies have smaller windows and lower frequencies use longer windows, the sliding windows produce fewer coefficients at lower frequencies than higher frequencies. This is different from the usual fixed window length STFTs, where an equal number of spectrogram points are obtained for each frequency. However, it allows a more natural interpretation of window overlap. It is now possible to maintain a constant percentage overlap irrespective of the size of the window. Note that the original time signal itself should be sufficiently longer than the longest window for this scheme to work.

When viewed in the time-frequency plane, instead of generating a uniformly spaced rectangular grid of numbers, the MCQS produces non-uniformly spaced points, where time points are linear but non-uniform and frequencies are log-spaced. See Figure (3.2) that shows the locations of the centers of all the analysis windows. This is an exaggerated plot made using

unrealistic values for some of the parameters but it serves the purpose of accentuating the structure of this grid. Observe that the vertical axis is on a log scale so the frequencies appear equispaced when in reality they are geometrically spaced. Observe that at low frequencies fewer data points are obtained because the windows are longer than those used at higher frequencies. Also notice the first point in each row appears later in time for lower frequencies. This is again due to the fact that longer windows are used for lower frequencies and the points shown in Figure (3.2) are referenced to the centers of the windows.

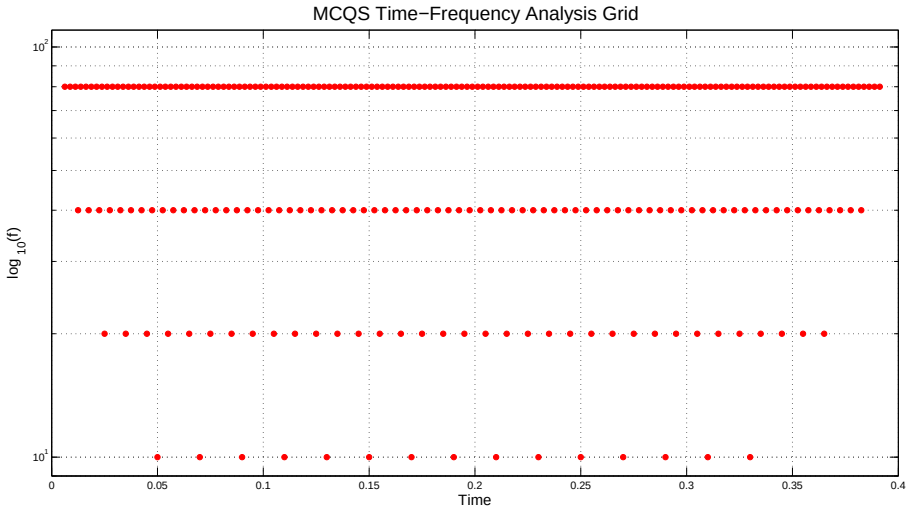

Figure 3.2: An exaggerated view of the centers of the windows used for generating an MCQS using $f_{min} = 10$ Hz, $f_{max} = 200$ Hz, $f_s = 400$ Hz, 1 bin/octave, signal length 0.4 seconds, 80% overlap between adjacent frames

An immediate consequence of having a non-rectangular matrix for the constant Q spectrogram data is that it cannot be displayed directly as a conventional image. In order to get a visual representation of this data one can perform interpolation on a set of points forming a uniform grid in the time-

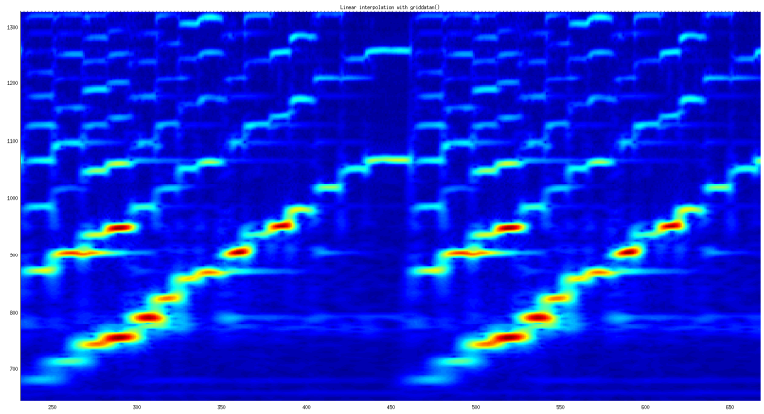

Figure 3.3: MCQS of a violin piece (f_{min}=110 Hz, f_{max}=1975 Hz, bins/oct = 48, overlap = 90%). Successive tones that are equally spaced in a perceptual sense are equally spaced visually too

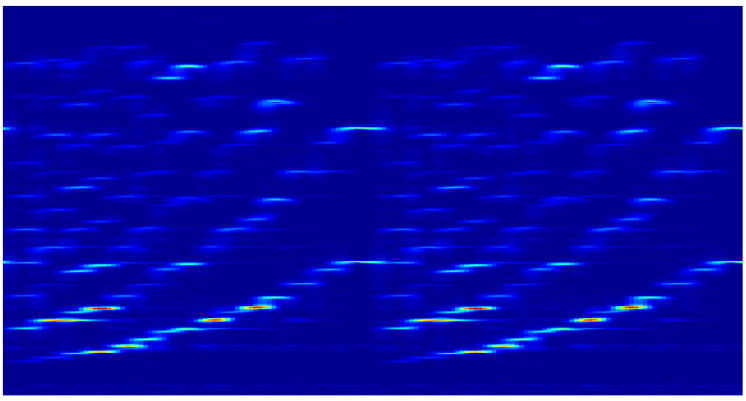

Figure 3.4: STFT spectrogram of the same violin piece (f_{min}=100 Hz, f_{max}=2000 Hz, window = 1024, overlap = 90%). Successive tones that are equally spaced in a perceptual sense are compressed visually in the lower frequencies and expanded visually in the higher frequencies

frequency plane. This method gives visually appealing results as shown in Figure (3.3). Unlike the usual STFT in Figure (3.4), consecutive notes appear equispaced in the MCQS, in agreement with the equal-on-a-log-scale perception by the human ear.

3.2.1 Matrix Formulation

The MCQS produces a nonuniform grid for the spectrogram and it is convenient from a computational point of view to vectorize the transform and store all the numbers as a single column vector.

In order to arrive at a matrix representation of the transform process, consider how a particular coefficient in the MCQS is generated. Let $x(n)$ be the time domain signal (that will later be represented as a column vector x). For given analysis frequency f_k with an associated constant Q window length N_k, the first element in the k^{th} row of the MCQS is generated using

$$X_0^{f_k} = \sum_{n=0}^{L-1} w_0(n)x(n) \tag{3.3}$$

where L is the length of the data sequence and $w_0(n)$ is the zero padded windowed complex exponential given by

$$w_0(n) = \begin{cases} v(n)e^{\frac{-j2\pi Qn}{N_k}}, & \text{if } 0 \leq n \leq N_k - 1 \\ 0, & \text{if } N_k \leq n \leq L-1. \end{cases}$$

Here $v(n)$ is any appropriate window function [18]. The operation in Equation (3.3) is an inner product between the data sequence $x(n)$ and a windowed complex exponential appended with zeros to make its length equal to the length of $x(n)$.

The subsequent coefficients in the same row can be obtained through a simple modification of Equation (3.3). Circularly rotated versions of the

$w_0(n)$ vector that maintain the required overlap between the windowed exponential sections can be used and the inner product operation with the data sequence $x(n)$ can be repeated. If $w_i(n)$ is the vector obtained after i circular shifts on $w_0(n)$, the i^{th} element of this MCQS row can be calculated as [2]

$$X_i^{f_k} = \sum_{n=0}^{L-1} w_i(n)x(n). \tag{3.4}$$

The circularly shifted versions of the $w_0(n)$ vectors corresponding to the analysis frequency f_k can be stored in the rows of a matrix $\mathbf{A}_k$. Hence the k^{th} row of the MCQS can be obtained as a vector through a simple linear operation,

$$\mathbf{A}_k\mathbf{x} = \mathbf{b}_k$$

where $\mathbf{b}_k$ is the vectorized form of the k^{th} row of the MCQS. A fraction of entries in the $\mathbf{A}_k$ matrix are zero which can be used to speed up this matrix multiplication (for instance, by storing it as a sparse matrix in Matlab). For future reference, note that the number of rows (r_k) in $\mathbf{A}_k$ depends on the length of the original signal L, the size of the window N_k, and the overlap fraction p, via the following relation, [3]

$$r_k = \left\lceil \frac{(L - N_k)}{N_k(1-p)} \right\rceil \tag{3.5}$$

[2]Note that the zero padded windowed complex exponential vector $w_0(n)$ depends on the frequency of interest and hence is a function of k. This is not shown explicitly in favor of cleaner notation.

[3]The actual implementation in software has been done according to the equation

$$r_k = \min_{l \in \mathbb{Z}^+} \{[1 + lN_k(1-p)] \geq L - N_k\}$$

where $\mathbb{Z}^+$ is the set of non-negative integers and the $[\cdot]$ denotes rounding to the nearest integer. This number may actually be off by 1 as compared to Equation (3.5). This minor rounding effect at the last overlapping window can be ignored in favor of the closed form expression for r_k in Equation (3.5).

where $\lceil z \rceil$ is the smallest integer greater than or equal to z (also called the integer-ceiling function). Next, all the $\mathbf{A}_k$ matrices can be stacked up into a tall $\mathbf{A}$ matrix,

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_0 \\ \hline \mathbf{A}_1 \\ \hline \vdots \\ \hline \mathbf{A}_k \\ \hline \mathbf{A}_{k+1} \\ \hline \vdots \end{bmatrix}.$$

This matrix when operated on the time domain signal $\mathbf{x}$ produces a vectorized form of the entire spectrogram. The complete operation can now be compactly represented as a matrix multiplication,

$$\mathbf{Ax} = \mathbf{b}$$

where $\mathbf{b}$ is the vectorized form of the MCQS.

3.2.2 Invertibility

The inverse problem for the matrix formulation can be posed as an unconstrained least squares optimization problem as follows.

$$\min_{\mathbf{x}\in\mathbb{R}^L} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$$

where $\mathbf{A} \in \mathbb{C}^{M\times L}$ and $\mathbf{b} \in \mathbb{C}^M$. Here L is the length of the data vector and M is the total number of rows in $\mathbf{A}$.

At first sight this may appear to be a *constrained* least squares optimization problem because $\mathbf{A}$ and $\mathbf{b}$ are complex valued whereas the data vector $\mathbf{x}$ is constrained be real valued. However, this constraint can be removed by decomposing into real and imaginary parts as shown below.

Splitting the transform matrix into real and imaginary parts

$$\mathbf{A} = \mathbf{A}_R + j\mathbf{A}_I$$

and repeating for the vectorized MCQS yields

$$\mathbf{b} = \mathbf{b}_R + j\mathbf{b}_I.$$

The goal is to solve for $\mathbf{x} \in \mathbb{R}^L$ which satisfies the following two conditions in the least squares sense:

$$\mathbf{A}_R\mathbf{x} = \mathbf{b}_R$$

and

$$\mathbf{A}_I\mathbf{x} = \mathbf{b}_I.$$

Forming a new augmented kernel matrix

$$\mathbf{\Lambda} = \left[\begin{array}{c} \mathbf{A}_R \\ \hline \mathbf{A}_I \end{array}\right]$$

and an augmented vector

$$\beta = \left[\begin{array}{c} \mathbf{b_R} \\ \hline \mathbf{b_I} \end{array}\right]$$

gives the following modified problem which contains only real terms,

$$\min_{\mathbf{x}\in\mathbb{R}^L} \|\mathbf{\Lambda x} - \beta\|^2$$

where

$$\mathbf{\Lambda} \in \mathbb{R}^{2M\times L}$$

and

$$\beta \in \mathbb{R}^{2M}.$$

This can be solved using any of the standard least squares methods like the Moore-Penrose pseudo-inverse.

This matrix formulation also raises the question of where exactly in time each MCQS value localizes information. For the sake of convenience, it is assumed that the magnitude and phase information in each MCQS value corresponds to the center of the window that was used to generate it. The magnitude interpretation seems quite natural however it may seem odd that the phase is also referenced to the center of the window (instead of the beginning of the window as it should be if using a FFT-based spectrogram). It will later become clear that in the implementation of the phase vocoder, the center of the window makes phase assignment easier. This is also consistent with the psychoacoustic observation that the human ear is relatively insensitive to phase differences between two signals at different frequencies. Hence, even if each row of the MCQS is offset by a constant phase that is different for each row, the perceptual difference may be minimal.

It is also useful to picture the situation if the centers of the windows at all frequencies were to line up exactly. As shown in Figure (3.5), the different constant Q windows used for analyzing the underlying time signal line up symmetrically about their centers. Hence it makes sense in referring to the collection of these MCQS elements as an actual time-slice for the time instant corresponding to the vertical line. Obviously, this situation will occur very rarely and in other cases it will be necessary to deal with some kind of approximation to extract a "pseudo-time-slice"

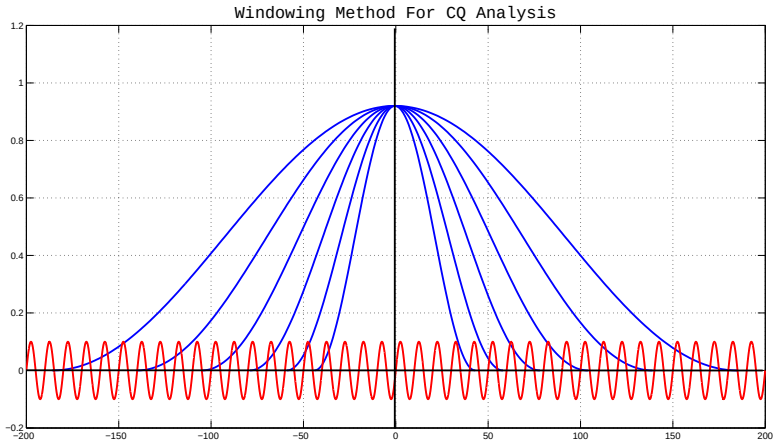

Figure 3.5: Constant Q windows in MCQS calculations lined up at symmetrically about their centers

from the MCQS. This topic will be further explored in Chapter 4 on phase vocoding using the MCQS.

Chapter 4

Implementation of a Phase Vocoder using the MCQS

4.1 Review of Present Phase Vocoding Techniques

The first phase vocoder (channel vocoder) was proposed in the early days of analog signal processing [12]. The idea of a phase vocoder specifically for musical applications was proposed by Dolson [13] as a means to perform audio analysis and modifications such as time scale modifications, pitch scaling and tempo editing.

The phase vocoder is an analysis-synthesis technique that operates on the spectrogram of the signal and modifies the amplitude and phase values in that domain. It rests on the assumption that there is a certain underlying time-varying model that describes the signal mathematically. The analysis

section attempts to estimate this model whereas the synthesis section reconstructs the audio signal after modifications. The synthesis section is often called "additive synthesis" to emphasize the fact that phase vocoders treat any audio signal as a collection of time shifted bursts of sine waves of specific amplitude envelopes and frequencies.

The main advantage of implementing phase vocoders with the constant Q spectrogram instead of STFTs is that the constant Q representation is closer to how humans perceive music. Previous attempts at incorporating phase vocoding technique in the constant Q transform [14] were inadequate and debatable. Here a novel technique that follows the lines of [15], [16] and [17] is developed.

FFT-Based Phase Vocoder

The analysis section of a conventional FFT-based phase vocoder operates by picking out consecutive time-slices from the given audio signal. It locates the peaks in the consecutive FFT vectors and estimates the frequency at the peak by using the relationship between the phase difference and the time difference as

$$f_n = (\theta_2 - \theta_1 + 2\pi n)/(2\pi(t_2 - t_1)). \tag{4.1}$$

Here t_1 and t_2 are the reference times at the two adjacent time windows, θ_1 and θ_2 are the phases at a particular peak in these adjacent spectral frames and the integer n is chosen such that the estimated frequency is close to the FFT-bin frequency. This method gives better estimates than just picking the frequency at the peak because the energy in FFTs bins usually spreads out

over multiple neighbors due to spectral leakage.

The modification step assigns new phase and amplitude values to the FFT bins (the exact assignment depends on the audio editing operation). Various schemes of phase assignment have been suggested [15], [17], [20] in order to maintain phase coherence during this editing operation. Most of these methods are heuristic and are based on visual analysis phase profiles in spectrograms of real world signals or by studying the behavior of phase when certain types of tapering-end windows are used. One of the most commonly used phase assignment strategy is called the phase-locked vocoder [19] that exploits the property of windowed FFTs that the phase values of bins under the peak are related and "locked" in some way to the phase at the peak. It is observed that for most commonly used window functions such as Hamming, Hanning and Gaussian, the phases at the bins neighboring the peak are either 0 or π radians offset from the phase at the peak. This zero-π pattern can be construed from the phase pattern of the Fourier Transform of these windows in continuous-time. The exact offset depends on how far the bin is from the peak bin. In fact, if the phase at the peak is θ_k, a good phase assignment strategy is to use the relation

$$\theta_{k\pm n} = ((\theta_k + mod(n, 2)\pi))_{(-\pi/2,\pi/2)} \tag{4.2}$$

to assign the phase at a bin that is n indexes away from the peak. Here $mod(n, 2)$ is the remainder after dividing n by 2 and $((\cdot))_{(-\pi/2,\pi/2)}$ indicates the operation of wrapping the phase to the interval $(\frac{-\pi}{2}, \frac{\pi}{2})$ which is the usual range of the principal values of the inverse tangent function. For

instance, see Figure (4.1). It shows a 64-point signal and the magnitude and phase components of its DFT. The frequency axes in the magnitude and phase plots are aligned to enable the examination of the phases under the peak. Observe that the phase at the peak is about $\pi/2$ and that at the bins neighboring the peak is $-\pi/2$ which is off by π from the peak.

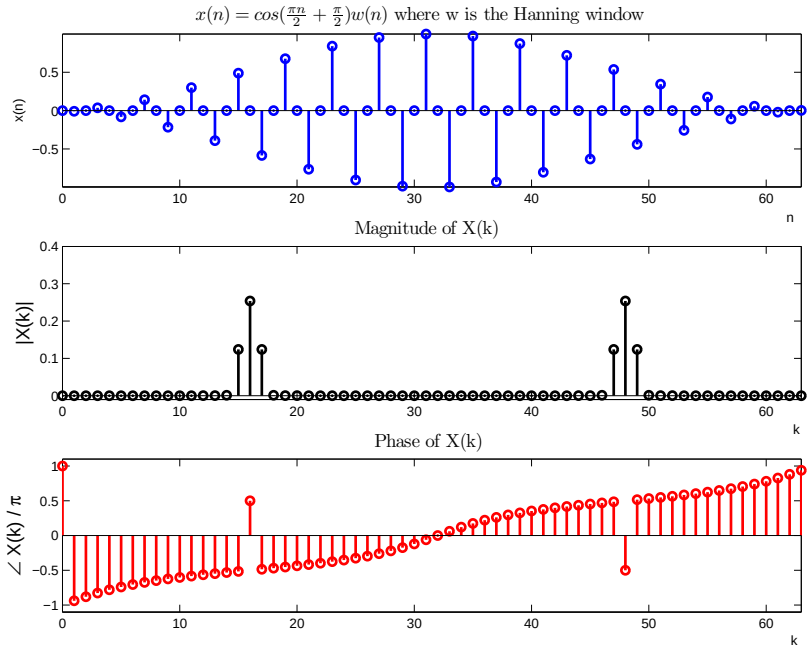

Figure 4.1: Phase under the peak for a regular 64-point DFT

The classical additive-synthesis section of the phase vocoder then inverts these FFTs back to time domain and then overlap-adds the time bursts to generate the modified sound signal.

4.2 Implementation with the MCQS

The constant Q phase vocoder operates on the MCQS instead of FFTs. The exact process is now described here in detail.

4.2.1 Analysis

The analysis section starts by choosing an input hop size and an output hop size. For time stretching applications, the output hop size is taken to be a multiple of the input hop size, where the multiplication factor is equal to the time stretch desired.

The iterative algorithm moves through the input hop size and picks out a new time-slice from the MCQS. However, since the points in the MCQS are not evenly spaced, a literal time-slice does not exist. A "pseudo-time-slice" can be constructed by the following correction method. The points that are nearest to time instant of interest are chosen and their phases are corrected to reference this instant. This phase correction is proportional to the frequency and the time delta. The magnitudes are set equal to the nearest points chosen. This process is depicted for a few points in Figure (4.2).

The analysis section locates the spectral peaks in this pseudo-time-slice and estimates the frequency at the peak using the same procedure as a regular phase vocoder in Equation (4.1).

In the modification step, new phase values and amplitudes are assigned to

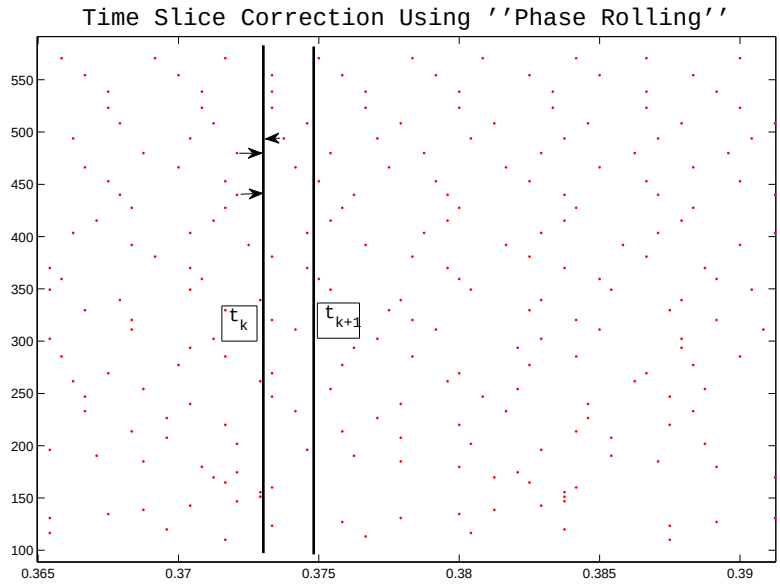

Figure 4.2: Time-slice correction by adjusting the phases

the corresponding bins in the new MCQS. To achieve time stretching, the magnitudes are kept unchanged and the phase values are assigned using the relation

$$\theta_k = \theta_{k-1} + 2\pi f \ \delta t_{out}$$

where the subscript k denotes the k^{th} peak, the frequency f is estimated in the analysis step and δt_{out} is the output time hop. In other words, a new phase value is assigned so that the frequency appears to roll the phase through a different time hop in order to achieve time scale modification.

As with the FFT-based phase vocoder, the phase in the region around the spectral peak must be locked to the movement of the phase at the peak. It was discussed in Section 4.1 that for the FFT-based phase vocoder, the phase-locking strategy is to assign a phase that is offset from the phase at

the peak by either 0 or π depending on whether the FFT-bin is an even or odd number of bins away from the peak, respectively. In case of the MCQS, Equation (4.2) does not hold because the MCQS time-slices are not literally FFT vectors. It is possible to mathematically analyze the phases under the peak to arrive at the correct phase-locking technique for the MCQS setting. It turns out that the phase assignment strategy for the MCQS is simpler than the FFT phase vocoder and it suffices to assign constant phase equal to that at the peak over the entire peak region. So, the phase at any bin that is n indexes away from the peak bin (assumed to be at the index k) must be assigned according to the simple relation

$$\theta_{k\pm n} = \theta_k. \tag{4.3}$$

See Figure (4.3) for an example on the behavior of the phase under the peak[1] in a time-slice of an MCQS. Some of the mathematical subtleties involved in this crucial phase assignment step shown in Equation (4.3) are further explored in Appendix A. This strategy does result in wrong phases getting assigned to bins far away from the peak but their magnitudes are relatively small and so the phase assigned to those points is relatively unimportant. In fact, this error far away from the peak region also affects regular FFT-based phase vocoders because Equation (4.2) holds only under the peak and not for the complete FFT vector.

The reason for assigning constant phase under the peak can also be

[1]Note that the plots in Figure (4.3) were generated by assuming that no time-slice correction was needed and the centers of all the windows were lined up. See also Section 3.2.2 and Figure (3.5).

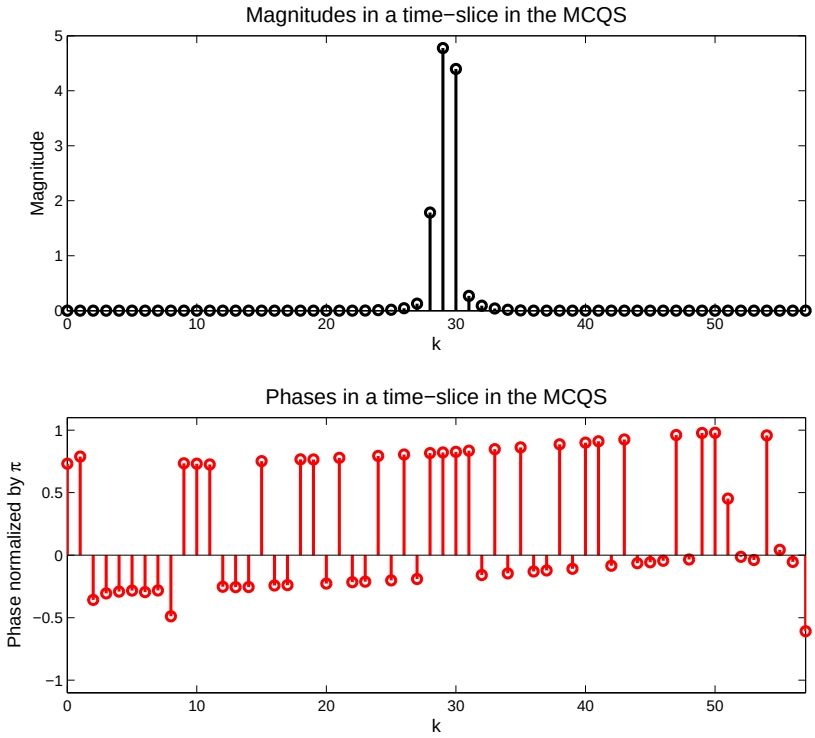

Figure 4.3: Phase values under the peak of a time-slice in an MCQS. The MCQS was generated for a sinusoid of frequency 255.5 Hz sampled at 1200 Hz using $f_{min} = 110$ Hz, $f_{max} = 6000$ Hz, 24 bins/octave and 95% overlap

understood intuitively. Suppose the signal being analyzed is a single sinusoid of frequency f_k. Suppose a certain constant Q window length N_k is tuned to analyze this frequency f_k. The constant Q window length for the adjacent frequency f_{k-1} is given by $N_{k-1} = N_k \cdot 2^{1/\lambda}$ where λ is the number of bins per octave. Similarly, for the frequency f_{k+1} the window length is given by $N_{k+1} = N_k \cdot 2^{-1/\lambda}$. For typical values of λ that are around 24 or 48 bins per octave, these adjacent frequency windows have lengths very close to N_k. The procedure of finding a particular MCQS coefficient is basically a correlation of a windowed section of the signal with a complex exponential

at some frequency. Correlating with the correct window N_k results in a coefficient of largest magnitude and a certain phase value. When the correlation is done with a window for a slightly different frequency, say using N_{k-1} or N_{k+1}, the magnitude of the coefficient drops. However, since the enveloped sinusoid frequency differs only slightly from the actual frequency, the phase offset that gives the best correlation is still close to the phase obtained when the correlation is done with the actual frequency f_k. In conclusion, the best correlation with the complex exponential at these slightly different frequencies f_{k-1} and f_{k+1} is obtained at a phase shift that is very close to the phase obtained by correlating with the correct frequency f_k. This intuition is formalized in Appendix A.

It is important to note that this assumption holds only for "nice" windows such as the Hamming, Hanning, Gaussian and Blackman windows that roll off to negligibly small values near the end. Otherwise the end terms may introduce larger errors in the phase obtained from the correlation and the constant phase under the peak assumption is no longer true. Detailed mathematical analysis to quantify the error incurred in this constant phase assumption is also given in Appendix A.

4.2.2 Resynthesis

In the resynthesis step, the edited MCQS is inverted using the least squares technique described in Section 3.2.2. A subtle issue here is what transform matrix should be used during inversion. It is necessary to use a new matrix, A′, that uses a different overlap factor to achieve the necessary time scaling

but generates the same MCQS structure as the original transform matrix. This is akin to stretching the original MCQS like a rubber membrane to obtain a new time scaled MCQS that is to be inverted. The new overlap factor can be calculated from the ratio of output to input time hop size, the old overlap factor and Equation (3.5). Suppose the signal is to be scaled by a factor s. The goal is to find a new overlap factor p' that gives the same number of rows r_k for this stretched signal of length sL. From Equation (3.5), it suffices to have

$$\frac{L - N_k}{(1-p)N_k} = \frac{sL - N_k}{(1-p')N_k}$$

or

$$p' = 1 - \frac{(sL - N_k)(1-p)}{(L - N_k)}. \tag{4.4}$$

It is clear from Equation (4.4) that the new overlap factor depends on the frequency index k indicating that a different overlap factor will be needed for each window length N_k in order to realize a structure for $\mathbf{A}'$ that is identical to $\mathbf{A}$. This is not compatible with the current structuring of the transform matrix that requires that the same overlap factor be used for every window. This problem can be fixed by ensuring $L \gg N_k$ so that

$$p' = 1 - s(1-p). \tag{4.5}$$

This approximation gives rise to "end-effect" errors for a few window lengths at the very end of the analysis duration. This is due to the small differences in the number of rows in the submatrices $\mathbf{A}'_k$ and $\mathbf{A}_k$, causing $\mathbf{A}'$ and $\mathbf{A}$ to have different number of rows. In actual implementation, this can be fixed by changing the size of the edited MCQS, either by appending

dummy values or truncating so that the MCQS row lengths becomes compatible with the structure of the inversion matrix $\mathbf{A}'$.

The final step is to invert the edited MCQS using the pseudo-inverse of $\mathbf{A}'$ as discussed in Section 3.2.2.

The actual software implementation of the analysis-resynthesis sections of the phase vocoder suffers from memory limitations too. For very long audio signals, the size of the transform matrix ($\mathbf{A}$) may get unwieldy and calculating the pseudo-inverse impossible. One way of bypassing this is phase vocoding smaller pieces of the full audio signal and then stitching all the edited pieces together. However, additional processing will be required to get rid of any discontinuities that may occur at the points where the pieces are stitched to generate the full length audio.

Chapter 5

Future Work

This thesis addressed the development of a modified constant Q spectrogram that prevents time decimation unlike the original constant Q transform formulation. It was also shown that the transform can be mathematically set up as a matrix operation and a good quality inverse can be obtained by solving a least-squares problem.

It will be interesting to explore how the MCQS is related to other kinds of transforms, possibly the Mellin transform and wavelet transform. The use of variable length windows for generation of the MCQS makes it quite similar to wavelet-based spectrograms. However, the non-uniform structure of the time-frequency grid will make exact comparison quite complicated.

It was shown that a good quality inverse can be obtained using a matrix pseudo-inverse. This is an improvement over regular constant Q spectrograms that cannot be inverted so easily because not all windows

analyze all data points. The MCQS overcomes this by analyzing each data point with sliding windows of varying lengths. The parameters are chosen so that the transform matrix has more rows than columns. It may be an interesting problem to explore the causes for the loss of rank of this transform matrix. Guaranteeing a full rank transform matrix can give exact (unique) inverse unlike the current method that gives only the best answer in the least-squares sense.

A versatile phase vocoder application built using the MCQS was also discussed in this thesis. It was shown that the phase assignment step in phase-vocoding requires some mathematical insight on the behavior of phase values under spectral peaks. Only one type of assignment strategy called "phase-locked vocoder" was presented and it may be instructive to examine the performance of other strategies analogous to the regular FFT-based vocoders.

It was shown that assigning constant phase under the peak is approximate and the error term was quantified in Appendix A. It is not clear if correcting for these small error terms in the phase assignment step can give any perceptible improvement over the method of constant phase assignment. Moreover, it may be possible to improve the time-slice correction method which is another source of error in the MCQS phase vocoder.

A time stretching application using the MCQS phase vocoder was implemented in Chapter 4 of this report. Other applications worth exploring include tempo scaling, pitch scaling and spectral morphing [24]. Spectral

morphing allows smooth transition between a source frequency spectrum and destination spectrum over a specific time interval. Typical spectral morphing algorithms create smooth fictitious transitions between spectral peaks that are nearby in frequency. Since the human ear detects closeness between frequencies on a log-scale, it is reasonable to expect that running a spectral morph on an MCQS (instead of a regular STFT spectrogram) should produce better results.

Appendix A

Detailed Derivation of Phase Under the Peak

This appendix analyzes phase values of MCQS coefficients under the peak. For concreteness, the analysis uses a Hanning window and considers the continuous time analogs for simplicity. The analysis naturally extends to other windows similar to the Hanning window, such as Hamming and Blackman.

Assume that a constant Q window tuned to some angular frequency ω_1 is used to analyze a signal that is a pure sinusoid of some angular frequency ω. The behavior of the MCQS coefficient when ω_1 is close to the actual frequency ω can be characterized explicitly.

The continuous time Hanning window centered about the y-axis and

tuned to the frequency ω_1 with a constant Q is given by

$$w(t) = \begin{cases} \frac{1+\cos(\omega_1 t/Q)}{2}, & \text{if } -\pi Q/\omega_1 \leq t \leq \pi Q/\omega_1 \\ 0, & \text{otherwise.} \end{cases}$$

This is pictured in Figure (A.1). The time axis has been normalized appropriately to fit in [-1,1]. Notice that as Q gets larger, the window gets wider in the time domain indicating that its frequency response gets narrower (and more "resonant") in the frequency domain.

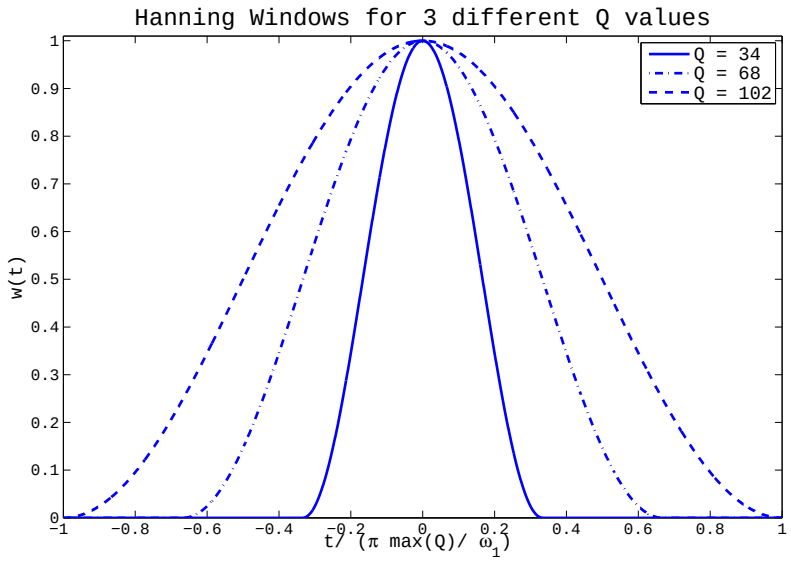

Figure A.1: Continuous time Hanning window, centered and tuned to ω_1 for three different Q values

The Fourier Transform evaluated at a frequency ω_1 using the constant

Q tuned window is given by

$$\begin{aligned} FT(\omega_1) &= \int_{-\pi Q/\omega_1}^{\pi Q/\omega_1} \cos(\omega t + \phi) \frac{1 + \cos(\omega_1 t/Q)}{2} e^{-j\omega_1 t} dt \\ &= \int_{-\pi Q/\omega_1}^{\pi Q/\omega_1} \frac{\cos\phi}{2} \cos(\omega t)(1 + \cos(\omega_1 t/Q)) \cos(\omega_1 t) dt \\ &\quad + j \int_{-\pi Q/\omega_1}^{\pi Q/\omega_1} \frac{\sin\phi}{2} \sin(\omega t)(1 + \cos(\omega_1 t/Q)) \sin(\omega_1 t) dt \\ &=: \frac{\cos\phi}{2} FT_R(\omega_1) + j \frac{\sin\phi}{2} FT_I(\omega_1). \end{aligned}$$

Define

$$\begin{aligned} A &= \frac{\sin\left(\frac{\pi Q\,(\omega+\omega_1)}{\omega_1}\right)}{\omega + \omega_1} \\ B &= \frac{\sin\left(\frac{\pi Q\,(\omega-\omega_1)}{\omega_1}\right)}{\omega - \omega_1} \\ C &= \frac{\sin\left(\frac{\pi Q\left(\omega+\omega_1+\frac{\omega_1}{Q}\right)}{\omega_1}\right)}{2\left(\omega + \omega_1 + \frac{\omega_1}{Q}\right)} \\ D &= \frac{\sin\left(\frac{\pi Q\left(\omega-\omega_1+\frac{\omega_1}{Q}\right)}{\omega_1}\right)}{2\left(\omega - \omega_1 + \frac{\omega_1}{Q}\right)} \\ E &= \frac{\sin\left(\frac{\pi Q\left(\omega_1-\omega+\frac{\omega_1}{Q}\right)}{\omega_1}\right)}{2\left(\omega_1 - \omega + \frac{\omega_1}{Q}\right)} \\ F &= \frac{\sin\left(\frac{\pi Q\left(\omega+\omega_1-\frac{\omega_1}{Q}\right)}{\omega_1}\right)}{2\left(\omega + \omega_1 - \frac{\omega_1}{Q}\right)}. \end{aligned}$$

Then

$$\begin{aligned} FT_R(\omega_1) &= A + B + C + D + E + F \\ FT_I(\omega_1) &= -A + B - C + D + E - F, \end{aligned}$$

and in terms of these quantities,

$$\begin{aligned} |FT(\omega_1)| &= \frac{1}{2}\sqrt{(FT_R(\omega_1))^2 \cos^2\phi + (FT_I(\omega_1))^2 \sin^2\phi} \\ \angle FT(\omega_1) &= \tan^{-1}\left(\frac{FT_I(\omega_1)}{FT_R(\omega_1)} \tan\phi\right). \end{aligned} \tag{A.1}$$

Observe that when $\omega_1 \to \omega$,

$$\begin{aligned} A, C, D, E, F &\to 0 \\ B &\to \frac{\pi Q}{\omega} \end{aligned}$$

so that

$$\begin{aligned} |FT(\omega_1)| &\to \frac{\pi Q}{2\omega} \\ \angle FT(\omega_1) &\to \tan^{-1}(\tan\phi). \end{aligned}$$

Consider the behavior of $\angle FT(\omega_1)$ when ω_1 varies around ω by substituting $\omega_1 = \omega + \Delta\omega$. This is typically the case when examining "bins

under the peak" in the MCQS. In practice, $\Delta\omega$ can be understood as the spacing between the bins under the peak. Defining a new quantity

$$\epsilon := \frac{\Delta\omega}{\omega}.$$

the phase function in Equation (A.1) can be expressed as a function of ϵ. Notice that $0 < \epsilon < 1$ whenever the neighboring frequency bin is close to ω. Taking the Taylor series expansion of $\angle FT(\epsilon)$ about $\epsilon = 0$ yields

$$\angle FT(\epsilon) = \tan^{-1}(\tan(\phi)) - \frac{\sin(\phi)\cos(\phi)}{4Q^2 - 1}\epsilon + o(\epsilon^2) \tag{A.2}$$

where $o(\epsilon^2)$ denotes all the terms containing the second and higher powers of ϵ. It is clear from Equation (A.2) that for small ϵ, the first and higher order terms are negligible, especially when Q is quite large. Also note that in the MCQS formulation, the quantities $\Delta\omega$ and Q are coupled via the number of bins per octave (λ) parameter. Choosing a larger value of λ not only makes Q larger but also makes $\Delta\omega$ and ϵ smaller.

It is also possible to get a bound on the first order error term as follows.

$$\begin{aligned}
\left| -\frac{\sin\phi\cos\phi}{4Q^2-1}\epsilon \right| &\leq \frac{|\Delta\omega|}{\omega(4Q^2-1)} && \text{(A.3)}\\
&= \frac{\omega(2^{1/\lambda}-1)}{\omega\left(4\left(\frac{1}{2^{1/\lambda}-1}\right)^2-1\right)} && \text{(A.4)}\\
&= \frac{(2^{1/\lambda}-1)^3}{4-(2^{1/\lambda}-1)^2}\\
&\leq \frac{(2^{1/\lambda}-1)^3}{3} && \text{(A.5)}\\
&< (2^{1/\lambda}-1)^3.
\end{aligned}$$

where Equation (A.3) follows from the definition of ϵ and the fact that $|\sin\phi\cos\phi| \leq 1$; Equation (A.4) follows by considering the bin which is an immediate neighbor of the peak bin and using Equation (3.1) to substitute for Q and finally Equation (A.5) follows from the fact that $0 < (2^{1/\lambda}-1) < 1$.

To put this in perspective, consider a typical value of $Q = 34$ (corresponding to λ = 24 bins/ octave) that is often used in practice. Then the first order error term is bounded by 2.5160×10^{-5}.

The next step is to analyze the phases when a rectangular window

having the appropriate constant Q width is used.

$$\begin{aligned} FT(\omega_1) &= \int_{-\pi Q/\omega_1}^{\pi Q/\omega_1} \cos(\omega t + \phi) e^{-j\omega_1 t} dt \\ &= \int_{-\pi Q/\omega_1}^{\pi Q/\omega_1} \cos(\omega t + \phi) \cos(\omega_1 t) dt \\ &\quad -j \int_{-\pi Q/\omega_1}^{\pi Q/\omega_1} \cos(\omega t + \phi) \sin(\omega_1 t) dt \\ &=: FT_R(\omega_1) + jFT_I(\omega_1) \end{aligned}$$

where

$$\begin{aligned} FT_R(\omega_1) &= \int_{-\pi Q/\omega_1}^{\pi Q/\omega_1} \cos(\omega t + \phi) \cos(\omega_1 t) dt \\ FT_I(\omega_1) &= -\int_{-\pi Q/\omega_1}^{\pi Q/\omega_1} \cos(\omega t + \phi) \sin(\omega_1 t) dt. \end{aligned}$$

In terms of these newly defined quantities,

$$\angle FT(\omega_1) = \tan^{-1}\left(\frac{FT_I(\omega_1)}{FT_R(\omega_1)}\right). \tag{A.6}$$

The exact same steps as those for the Hanning window can now be imitated. Substituting for FT_I and FT_R in Equation (A.6), letting $\omega_1 = \omega + \Delta\omega$, expressing $\angle FT(\omega_1)$ as a function of $\Delta\omega/\omega =: \epsilon$ and finally taking the Taylor series expansion about $\epsilon = 0$ yield

$$\angle FT(\epsilon) = \tan^{-1}(\tan(\phi)) + \sin(\phi)\cos(\phi)\epsilon + o\left(\epsilon^2\right). \tag{A.7}$$

Observe that the first order error term in Equation (A.7) in case of the rectangular window differs from that in Equation (A.2) for the Hanning window

by a factor of $(4Q^2 - 1)$. Typically, Q is at least 34 which indicates a 4000 times magnification in the first order error term when the rectangular window is used. A slight extension of this result also explains why the constant phase assignment strategy does not work for windows that do not taper to zero at the end points.

Appendix B

Inverting an MCQS with only Magnitude Information

Many spectrogram editing techniques rely on modifying the magnitude of a spectrogram and inverting it back to obtain a new audio signal. Previous attempts at reconstructing audio from phase-less spectrograms have been proposed for regular STFT spectrograms. For instance, [21] proposes an iterative algorithm to obtain an audio signal whose magnitude STFT is close to the given STFT in the sense of a certain distance measure. An analysis of the convergence properties of this iterative algorithm emphasizes that it converges at least to a local optimum. However, this algorithm is computationally burdensome and not suited for real time applications. A modified version of this algorithm that reconstructs phase one frame at a time so it is more suited to real time implementation is described in [22] .

This appendix describes a phase reconstruction method for a phase-less MCQS using the phase vocoder strategy. The method has two steps

like the regular phase vocoder — analysis and synthesis. However, because there is no ground truth available for the phase information, Equation (4.1) cannot be used for reliable estimates of frequency. However, it is still possible to estimate the frequency by using the information about the width of the peak in a given MCQS time-slice and the knowledge of the actual frequencies at the MCQS bins.

Note that unlike other spectrogram reconstruction techniques, this method is not iterative so there are no convergence issues or associated computational complexity.

B.1 Analysis

The analysis starts by choosing a hop size. The algorithm moves through this input hop size and extracts a new frame from the MCQS. The next step is to use time-slice correction to generate a pseudo-time-slice as described in the analysis section for the MCQS phase vocoder in Section 4.2.1. There is now an additional subtlety that phases can be referenced to the pseudo-time-slice only from those points that have phases already assigned to them through previous loops of the algorithm. The magnitudes are chosen to be equal to these respective points in the MCQS. Now, in order to estimate the frequency at the peak, a method similar to DFT interpolation can be used [23]. Taking a näive approach, a second order polynomial fit can be applied in the peak region and using the peak of the parabola gives a better estimate of the frequency for the peak. As noted previously, it is not possible to use the regular analysis Equation (4.1) because there is no phase

information available from the "future" time-slices. Figure (B.1) illustrates

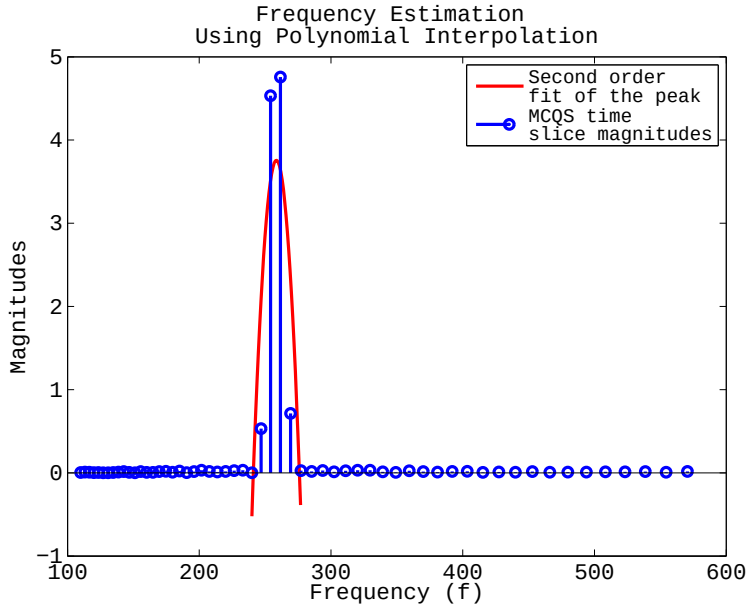

Figure B.1: Frequency estimation using a second order polynomial fit for magnitudes around the peak

this estimation method for the case of an audio signal that contains a single sinusoid of frequency 254.2 Hz. Observe that using just the peak frequency 261.6 Hz would have incurred more error than when using the peak of the parabola (at 258.9 Hz). This is still a considerably large error compared to the case of a regular MCQS phase vocoder where all the phase information is available. This highlights the importance of the knowledge of the original phases in obtaining satisfactory frequency estimates.

This estimated frequency is then used for adjusting the phases at the peaks through the time hop size and assigning these adjusted phases at the peaks in the future time-slice. As in the MCQS phase vocoder, the other phase values in the peak region can be locked to the phase at the peak so

that they are all equal.

B.2 Resynthesis

The resynthesis step reconstructs the audio signal using the matrix inversion technique discussed in Section 3.2.2. The reconstruction quality is not as good as the regular MCQS phase vocoder. This is not surprising because when reconstructing phases from scratch, the frequency estimates are not as accurate as the case where the ground truth about all the phase values is available beforehand.

References

[1] J. C. Brown, Calculation of a constant Q spectral transform, *J. Acoust. Soc. Am.*, January 1991.

[2] J. G. Roederer, The Physics and Psychophysics of Music: An Introduction, Springer, 3^{rd} Ed. (2001), pp. 24–28.

[3] J. C. Brown, M. S. Puckette, An efficient algorithm for the calculation of a constant Q transform, *J. Acoust. Soc. Am.*, November 1992.

[4] R. Bradford, J. ffitch, R. Dobson, Sliding with a constant Q, Proc. of the 11th Int. Conference on Digital Audio Effects (DAFx-08), Espoo, Finland, September 2008.

[5] R. G. Stockwell, L. Mansinha, R. P. Lowe, Localization of the Complex Spectrum: The S Transform, *IEEE Trans. on Sig. Proc.*, April 1996.

[6] D. FitzGerald, M. Cranitch, M. T. Cychowski, Towards an inverse constant Q transform, Audio Engineering Soc. Convention Paper, May 2006.

[7] G. Gambardella, The Mellin transforms and constant-Q spectral analysis *J. Acoust. Soc. Am.*, September 1979.

[8] P. Smaragdis, Relative Pitch Tracking of Multiple Arbitrary Sounds, *J. Acoust. Soc. Am.*, May 2009.

[9] O. Izmirli, A Hierarchical Constant Q Transform for Partial Tracking in Musical Signals, Proceedings of the 2nd COST G-6 Workshop on Digital Audio Effects (DAFx99), Trondheim, Norway. December, 1999.

[10] C. Schoerkhuber, A. Klapuri, Constant-Q transform toolbox for music processing, 7th Sound and Music Computing Conference, Barcelona, Spain, July 2010.

[11] H. Dudley, The Vocoder, Bell Labs, Record 18, pp. 122–126, December 1939.

[12] J.L. Flanagan, R. M. Golden, Phase Vocoder, *Bell System Technical Journal*, 1966.

[13] M. Dolson, The Phase Vocoder: A Tutorial, *Computer Music Journal*, Vol. 10, No. 4, 1986.

[14] J. Garas and P. C. W. Sommen, Time/Pitch Scaling Using the Constant-Q Phase Vocoder, Proceedings of ProRISC/IEEE Workshop on Circuits, Systems and Signal Processing, Mierlo, Netherlands, November 1998.

[15] J. Laroche, M. Dolson, Phase vocoder: About this phasiness business, Proceedings of IEEE ASSP Workshop on application of signal processing to audio and acoustics, 1997.

[16] J. Laroche, M. Dolson, Improved phase vocoder time-scale modification of audio, *IEEE Transactions on Speech and Audio Processing*, Volume 7, May 1999.

[17] P. Moller-Nielsen, Sound Manipulation in the Frequency Domain, Retrieved Feb 18, 2011 from `http://www.daimi.au.dk/~pmn/sound/`.

[18] J. G. Proakis, D. K. Manolakis, Digital Signal Processing, Prentice Hall, 4^{th} Ed. (2006) pp. 666–667.

[19] M. Puckette, Phase locked vocoder, IEEE ASSP Conference on Applications of Signal Processing to Audio and Acoustics, 1995.

[20] W. A. Sethares, Rhythm and Transforms, Springer-Verlag London, 1^{st} Ed. (2007), pp. 117–121. Chapter available online at `http://sethares.engr.wisc.edu/vocoders/Transforms.pdf`.

[21] D. W. Griffin, J. S. Lim, Signal Estimation from Modified Short-Time Fourier Transform, *IEEE Transactions on Acoustics, Speech and Signal Processing*, April 1984.

[22] X. Zhu, G. T. Beauregard, L. L. Wyse, Real-Time Signal Estimation From Modified Short-Time Fourier Transform Magnitude Spectra, *IEEE Transactions on Audio, Speech and Language Processing*, Vol. 15, No. 5, July 2007.

[23] V. K. Jain, W. L. Collins, D. C. Davis, High Accuracy Analog Measurements via Interpolated FFT, *IEEE Transactions on Instrumentation and Measurement*, Vol. 28, No. 2, June 1979.

[24] W. A. Sethares, A. Milne, S. Tiedje , A. Prechtl and J. Plamondon, Spectral tools for dynamic tonality and audio morphing, *Computer Music Journal*, Vol. 33, No. 2, pp. 71–84, Summer 2009.