TOOLS FOR INTERACTIVE AUDIO SIGNAL ANALYSIS BASED ON SLIDING DFT

Elisa Russo
Università degli Studi di Milano, Italy
visiting at
Department of Computer Science
University of Bath, UK
e.russo@bath.ac.uk

ABSTRACT

This article describes an application the author developed in order to compare analysis and synthesis of musical audio signals through Short Time Fourier transform (STFT), Constant Q and Sliding Discrete Fourier Transform (SDFT). This software is the basis for applications of SDFT and Constant Q to other consolidated synthesis techniques. By itself, it is a stand alone instrument for calculating and quickly comparing spectrum analysis and synthesis of musical signals. No expertise is required and it can for example be easily used by music composers without in depth knowledge of DSP processing tools.

1. INTRODUCTION

Sliding DFT (SDFT) is an analysis technique that thanks to the continuous growth in computer power can become an effective alternative to traditional FFT for DFT calculus [1, 2, 3]. Thanks to the CPU power constantly increasing, maybe in a future not so far away, the time of SDFT calculus will be acceptable even in real time applications [4]. The high precision intrinsic in the SDFT makes it a powerful instrument to analyze musical signals. In addition it surpasses some limitations like a power of 2 window size and constant bandwidth required in spectrum analysis. The combination with the Constant Q technique [5, 6] can be convenient in spectrum elaboration [2], most of all for the analysis of non stationary signals. After an introduction of the background and expectations in section2, I briefly present the theory of Sliding DFT and Constant Q in section 3 and 4. Subsequently, I present the software facilities in 6, 7 and 8. Description and considerations about the data structure, mostly related to the multithreading structure are covered in 9

2. BACKGROUND AND EXPECTATIONS

A musical audio signal is a non-stationary signal. Therefore, it is inappropriate to process an FFT on the whole signal during the spectrum analysis. The traditional way to go on is to apply an STFT that manages the time line through multiple superimposed and discontinuous frames. The discontinuity between consecutive frames brings the first issue we want to be surpassed. The second issue is the compromise between time and frequency resolution during the STFT calculus results. It is well-known that in order to have a good resolution in frequency the frame to be considered has to be long. On the other hand a long frame causes non-good time resolution. The Constant Q technique [5, 6] does not still solve yet these problems completely, however, the logarithm scale applied to the frame size is a good compromise between time and frequency for the human ear perception of musical signals. The issue is completely solved through the Sliding DFT in which the transformation from the time to the frequency domain is managed in a continuous way. The heavy quantity of information to be processed can be diminished applying to the SDFT the Constant Q technique. This is clear listening to the synthesis audio results in the tool of section 8, where the time reconstruction based on the spectrum just obtained is exactly the same of the original signal.

The tool developed lets the user transform an audio signal from the time to the frequency domain through both STFT and SDFT. The STFT output is visualized in a joint time frequency analysis representation. The same visualization has not been realized for the SDFT analysis since, with a common n sampling rate (22000 Hz), a continuous $n \times n$ matrix construction is still too heavy for the CPU power in hand.

Now let us introduce the basic theory used in the system, citing concepts presented in [1, 2, 5, 7].

3. SLIDING DFT

The SDFT is an algorithm for frequency-domain signal construction where the analysis window is moved by one sample.

From the definition of the Discrete Fourier Transform starting at time i:

$$X_j(n) = \sum_{k=0}^{N-1} x(k+j)e^{-2\pi ikn/N} \tag{1}$$

the following relation gives a full transform at every point of the signal:

$$X_j(n+1) = [X_j(n) - x(n) + x(n+N)]e^{2\pi ij/N} \tag{2}$$

This means that knowing the DFT of a signal at an N-points window it is possible to obtain the DFT of the time window advanced of one sample just with one complex multiplication and two real additions. Thanks to this, every new DFT is calculated directly from the result of the previous DFT. This increment by one point of the time window during the DFT computation leads to the name Sliding DFT. The complex multiplication can be computed calculating every time the sine or cosine signal, or simply reading a table in which the sine and cosine shapes are stored. The latter choice is clearly faster than the former one.

Just to sum up, the fundamental points of this algorithm are the following ones:

- Fourier analysis calculated on the single sample rather than on an N-sized array of samples;
- algorithm based on recursive interpretation of DFT equation;
- use of static look-up table in place of calls of trigonometric approximation functions for both Fourier analysis and resynthesis;
- computer speed depending only on the number of frequency bands;
- because the SDFT considers only half of the usual frequency interval, there is no doubling of the harmonics as known from the FFT.

In the case a new DFT output spectrum is desired every sample, or every few samples, the sliding DFT is computationally simpler than the traditional radix-2 FFT [8].

If only a finite number of trigonometric function-points are appearing speed gains.

A problem of discontinuity has to be pointed out since if an undesired single discontinuity happens in the original signal, the transformation will remain invalid for a longer time than for the FFT. Windowing with for example Hanning window overcomes the problem.

Finally, the formula 3 lets us reproduce the original signal in the time domain.

$$x_j(n) = \frac{1}{N}\sum_{k=0}^{N-1} X_j(k) e^{2\pi i j k/N} \tag{3}$$

4. CONSTANT Q

Basically, the Constant Q Transform aims to increase the low-frequencies resolution and decrement the high-frequency latency with a logarithmic ratio, emulating the human ear perception. This approach is very effective for musical signals since a sound with harmonic frequencies components gives rise to a constant pattern in the log frequency domain [5].

In our spectrum we want to consider frequencies in a logarithmic ratio and for every frequency a logarithmic number of points in the time domain representation.

Supposing that we want to be able to perceive quarter tones like different sounds, we have to subdivide an octave in 24 segments. Choosing f_{min} like the lowest frequency considered, the k-th spectral component is

$$f_k = (2^{1/24})^k f_{min} \tag{4}$$

Calling Q the distance between two contiguous frequencies and δf the bandwidth (having the frequency f_k, the bandwidth considered is $f_{k+1} - f_k$),

$$Q = \frac{f}{\delta f} = \frac{f_k}{f_{k+1} - f_k} = \frac{1}{2^{\frac{1}{b}} - 1} \tag{5}$$

Having $b = 24$:

$$Q = \frac{1}{2^{\frac{1}{24}} - 1} = 34 \tag{6}$$

If S is the sampling rate and T the sample time, $S = 1/T$.

The length of the window in samples at frequency f_k ($N[k]$ = size of frame k) is

$$N[k] = \left\lceil \frac{S}{\delta f_k} \right\rceil = \left\lceil \frac{S}{f_k} Q \right\rceil \tag{7}$$

Q says how many cycles are needed to make an analysis. The window of equation 7 contains Q complete cycles for each frequency f_k, since the period in samples is S/f_k. For example in order to distinguish between f_{k+1} and f_k when their ratio is $2^{1/24} \approx 34/33$ we must look to at least 33 cycles.

Finally, the number of observed harmonics is

$$k_{max} = \left\lceil b \log_2 \left(\frac{S}{f_{min}} \right) \right\rceil \tag{8}$$

The constant Q transform is equation 10. Since the number of terms varies with k, the result needs to be normalized by dividing the sum by $N[k]$.

The period in samples is $N[k]/Q$ (points in time domain containing an entire cycle of frequency k) and the cycles analyzed are always Q. Instead, in the traditional STFT (equation 9) the period in samples is N/k and the number of cycles analyzed is k.

$$X[k] = \sum_{n=0}^{N-1} W[n] x[n] e^{-j2\pi k n/N} \tag{9}$$

$$X[k] = \frac{1}{N[k]} \sum_{n=0}^{N[k]-1} W[k,n] x[n] e^{-j2\pi Q n/N[k]} \tag{10}$$

A well-known way to calculate Constant Q has been introduced by Judith Brown and Miller Puckette [7] in 1992. They proposed an algorithm able to increase the velocity of the Constant Q calculation merely based on the basic formula. The algorithm in object uses the Parseval theorem in order to multiply the FFT of the input signal with the equation called Kernel:

$$K[k, k_{cq}] = \sum_{n=0}^{N-1} w\left[n - \left(\frac{N}{2} - \frac{N(k_{cq})}{2} \right), k_{cq} \right] \times e^{j\omega_{k_{cq}}(n-N/2)} e^{-j2\pi k n/N} \tag{11}$$

5. SDFT WITH A CONSTANT Q

When we are considering a SDFT were the numbers of bins for every window vary according to a Constant Q decrease we are talking about a Sliding with a constant Q [2]. For the SDFT constant Q calculation, among the cases considered by the authors of [2], we have considered the offset of the frame aligned to the left, see equation 13.

$$X_j(n+1) = e^{2\pi i j Q/N[k]} \times \left(X_j(k) + \frac{e^{-2\pi i j Q} x(N[k]+n) - x(n)}{N[k]} \right) \tag{12}$$

Again the information needed during the synthesis is already calculated during the analysis. Therefore inside every step of the loop devoted to the analysis calculation of every bin the resynthesis of every bin is calculated as well.

$$x_j(n) = \sum_{k=0}^{B} X_j(n) e^{2\pi i Q/N[k]} \tag{13}$$

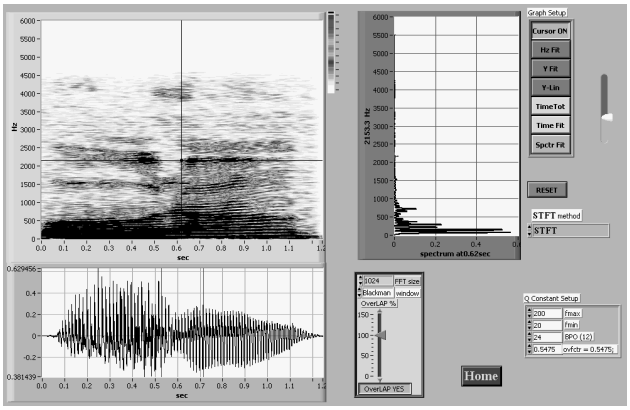

Figure 1: *STFT calculus.*

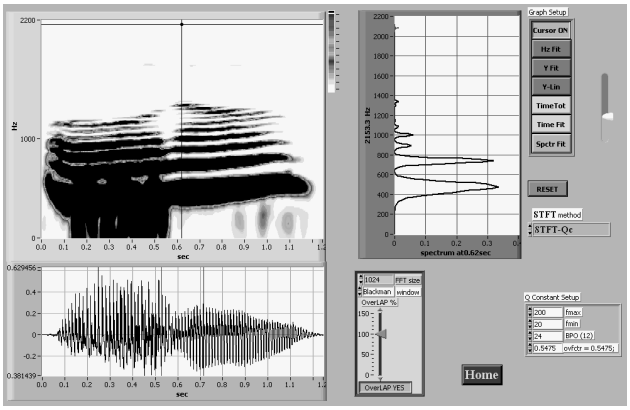

Figure 2: *STFT with a Constant Q calculus.*

6. ANALYSIS TYPE SELECTION

Now let us describe the structure and the aim of the testing tool for the above cited techniques. First, the user has to choose between the creation of the spectrogram and the Resynthesis of the original sound. The Spectrogram can be obtained through STFT or STFT with a constant Q with a particular visualization of every single frame in the time and frequency domain (figures 1 and 2). The resynthesis is performed in real time making the user conscious of the time spent during the calculus (figure 3) and letting him change parameter values in order to understand what is influencing the computing.

It is possible to upload a wave file or to directly create a sum of sinusoids concatenated during the time or added together. Through the first option sample rate and precision are fixed and visualized in the window, through the second one the user has to set his favorite values. The second option enables the user to make quick tests. At the moment stereo signals are analyzed only in their left channel, however it is possible to obtain other settings through a little effort.

7. STFT AND CONSTANT Q

The spectrum analysis can be obtained selecting "STFT" (figure 1) or "STFT-Qc" (figure 2) using the combo box under the "RESET" button. The third option of the combo box is "Qc Kernel" through which the user can visualize the values of equation 11.

It is possible to activate or deactivate the frame selection visualization, normalize the values currently visualized on the spectrum through the spectrogram selection, represent the amplitude values in a linear or logarithm way, visualize in the time domain all the signal or only the bins that correspond to the frame selected, normalize the spectrogram, process the calculus using the latest selections. In the STFT calculus it is possible to select the percentage of overlap wished.

Figure 2 shows an example of Constant Q calculus: the frequencies under 1000 Hz are now logarithmically emphasized. In "Constant Q SETUP" area the user can set the values of equation 8. In the Constant Q representation the user can control the minimum and maximum frequency and the bins per octave (BPO, that we also call b in section 4, the segments in which we want to subdivide the octave). Every window can have an overlap percentage and a particular sized and shaped window. In the showed example a vocal audio signal has been used.

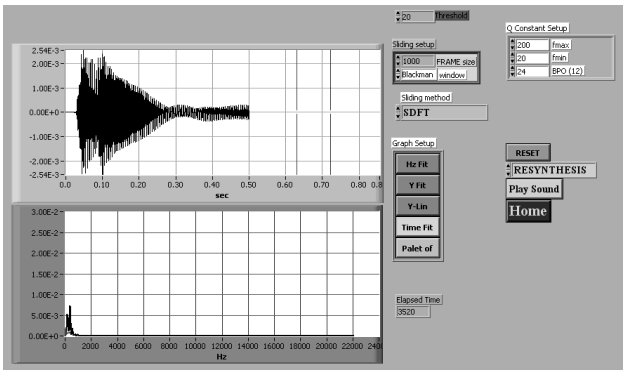

Figure 3: *SDFT synthesis process at the 3520th millisecond.*

8. SDFT AND CONSTANT Q

The window visualized after the selection of the resynthesis calculus is visible in the screen shot of figure 3. The milliseconds elapsed during the calculus are visualized under the label "Elapsed Time". Now we are not talking about STFT anymore but about SDFT and in addition we do not manage only the analysis but also the reconstruction of the original signal using only the spectral information just obtained. In the sliding process the resynthesised content is achieved through almost the same operations already performed for the analysis calculus. This is clear looking at the formulas 2 and 3.

The same contemporary use of the same code for the analysis and resynthesis process is devoted to the analysis and resynthesis of SDFT with a constant Q. In this case the equations considered for the analysis and the resynthesis have been 13 and 13. Differently from the traditional use of DFT, like FFT, SDFT does not need a windowing, since applying the formula 3 the signal synthesised is just exactly the same of the original signal, regardless of the size of the frame.

The combo box labeled "Sliding method" lets the user choose to analyze and synthesize a signal through SDFT or SDFT with a constant Q. The parameters that can be set are again the minimum and the maximum frequency, the maximum and the BPO. The Sliding setup lets the user choose the size of the frame sliding on the samples and the shape of the window we want to apply to

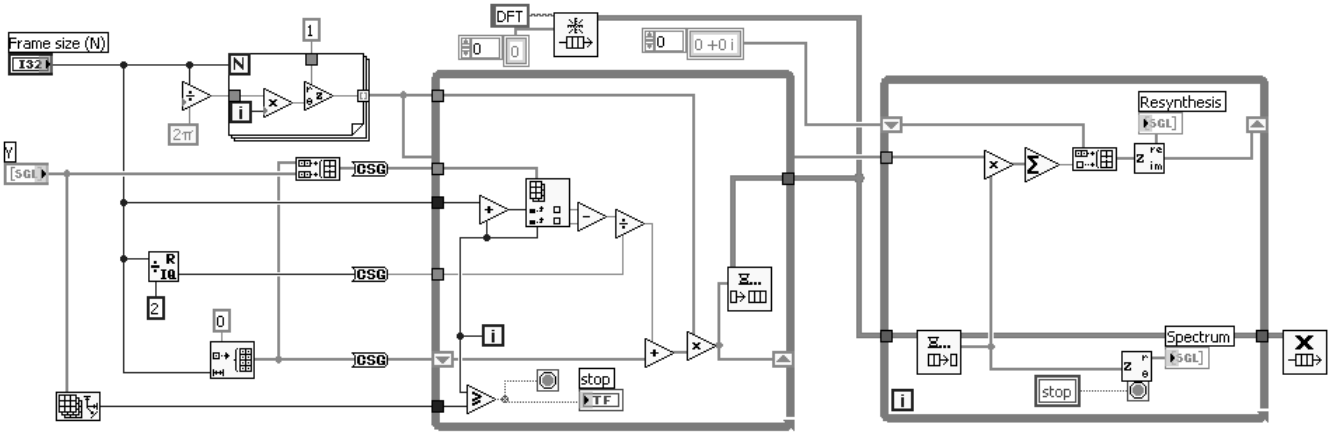

Figure 4: *LabVIEW Block Diagram - Multithreading basic structure*

the result. The graph setup normalizes the content of the graphs during the real-time calculus. The user can listen to the original audio file or to the synthesized result at the end of the reconstruction or during it. The audio file processed in figure 3 is the pluck of a violin.

9. DEVELOPMENT LANGUAGE AND PROGRAM STRUCTURE

The implementation has been achieved through LabVIEW 7.1, using lvsound.dll, on a Dell M1330, Intel(R) Core (TM)2 Duo CPU T7250 @2.00 GHz @2.00 GHz, 2.00 GB of RAM working with Microsoft Windows XP Professional Version 2002 Service Pack 3. The two processors are used in parallel during the SDFT analysis and resynthesis calculus. Two threads are dedicated in parallel to one operation each. In figure 4 there is the essential dataflow multithreading management dedicated to the analysis (the while loop contained in the black square in the middle) and to the resynthesis (the while in the black square on the right) processes. The data shared by the both processors are exchanged through a queue. The system can be interfaced with OSC and MIDI for real time interactions. A dataflow programming language has been chosen for its accomplished convenience in DSP manipulation. LabVIEW has been chosen for its completeness and robustness.

10. CONCLUSIONS

The work represents a useful instrument for everyone who wants to become familiar with STFT, Constant Q and SDFT analysis and resynthesis. The software described gives a quick idea about the accuracy gained and the time required in the sound manipulation between the time and the frequency domain. It also represents the basis for the SDFT integration to other consolidated analysis/resynthesis techniques. In particular the author is studying how to apply the SDFT to the partials tracking in sound analysis based on Sinusoidal Modeling [9].

11. ACKNOWLEDGMENTS

I am very grateful to Prof. John Fitch for his help and suggestions during my stay in Bath and to Prof. Goffredo Haus who is supporting my PhD.

12. REFERENCES

[1] Fitch J., Dobson R., and Bradford R. Sliding is Smoother than Jumping. In *International Computer Music Conference (ICMC)*, pages 287–290, 2005.

[2] Fitch J., Dobson R., and Bradford R. Sliding with a Constant Q. In *Digital Audio Effects (DAFx)*, 2008.

[3] Fitch J., Dobson R., and Bradford R. Slidinf DFT for Fun and Musical Profit. In *Linux Audio Conference (LAC)*, 2008.

[4] James A. Moorer. Audio in the New Millenium. *J. Audio Eng. Soc.*, 48(5):490–498, May 2000.

[5] Judith C. Brown. Calculation of a Constant Q spectral transform. *Journal of the Acoustical Society of America*, 1(89):425–434, January 1991.

[6] D. Fitzgerald, M. Cranitch, and Marcin T. Cychowsky. Towards an inverse constant q tranform. In *Proceedings 120th AES Convention*.

[7] Judith C. Brown and Miller S. Puckette. An efficient algorithm for the calculation of a constant Q transform. *Journal of the Acoustical Society of America*, 92(5):2698–2701, 1992.

[8] E. Jacobsen and R. Lyons. The Sliding DFT. *IEEE Signal Processing Magazine*, 2003.

[9] R. Mcaulay and T. Quatieri. Speech analysis/synthesis based on a sinusoidal representation. *Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing], IEEE Transactions on*, 34(4):744–754, 1986.

[10] Fitch J., Dobson R., and Bradford R. The Sliding Phase Vocoder. In *International Computer Music Conference (ICMC)*, pages 449–452, 2007.

[11] X. Serra. *Musical Sound Modeling with Sinusoids plus Noise*, pages 91–122. Swets & Zeitlinger, 1997.