A Bounded-Q Fast Filter Bank for Audio Signal Analysis

Filipe C. C. B. Diniz, Iuri Kothe, Luiz W. P. Biscainho, Sergio L. Netto

***Abstract*— This paper describes a new spectral analysis tool, the bounded-Q fast filter bank (BQFFB), which consists of a high selectivity filter bank with a piecewise linear spacing between its channels. This approximately geometric representation yields relatively low computational cost and a suitable frequency resolution, thus making the BQFFB an efficient tool for musical analysis applications. A description of the BQFFB algorithm is presented along with a complete evaluation of its corresponding computational complexity. Computer simulations are then included to illustrate the BQFFB performance against other methods from the literature, such as the fast Fourier transform, the standard fast filter bank, and the constant-Q transform.**

***Index Terms*— Spectral analysis, audio analysis, filter banks.**

I. Introduction

Audio applications involving spectral analysis, like music transcription [1], rely on high selectivity in frequency, which is not attainable by the widely employed techniques based on the discrete Fourier transform (DFT) [2]. To overcome this difficulty, Lim and Farhang-Boroujeny [3] combined the structure of the fast Fourier transform (FFT) with more selective filters, giving rise to the fast filter bank (FFB), which allows fast implementations, as described in [4].

In the equal tempered scale (ordinarily used in Western music) the musical notes are geometrically spaced in frequency; this can be roughly linked to the organization of the human auditory system [5]. As a consequence, employing a linear resolution analysis in music-oriented applications implies an unnecessary amount of computations. The first attempt to overcome this difficulty was the constant-Q transform (CQT) [6], which can be seen as a set of DFTs of variable lengths, each of them contributing an individual channel to the final representation. Even though it has an efficient approximate implementation [7], the CQT is deprived of an FFT-like fast algorithm by its own structure. Moreover, it retains the poorly selective DFT channels.

These issues call for a solution that provides a better trade-off among adequately variable resolution (in Hz), high selectivity, and reasonable complexity. The desired properties are achieved by the bounded-Q fast filter bank (BQFFB) introduced in this paper. In the bounded-Q transform (BQT) [8], the frequency channels are linearly distributed inside each octave, thus making the overall frequency scale only approximately geometric. The partially linear spacing, however, allows an FFT-like algorithm to be used, thus greatly reducing the overall BQT implementation cost. The high selectivity is then achieved by employing FFB kernel filters in the BQFFB structure. Table I compares the resolution, filter selectivity, and complexity issues for the spectral methods mentioned in this paper, namely FFT, FFB, CQT, BQT, and BQFFB. From this table, one can notice how the BQFFB achieves an excellent compromise between all desired spectral analysis characteristics.

The authors are with PEE/COPPE & DEL/Poli – UFRJ, Caixa Postal 68504, 21941-972, Rio de Janeiro, RJ, Brazil. email: {filiped, iuri, wagner, sergioln}@lps.ufrj.br

TABLE I

Comparison between different spectral analysis tools. The (*) indicates computational complexity in the context of high-selectivity methods.

Analysis Tool	Frequency Resolution	Channel Selectivity	Computational Complexity
FFT	linear	low	low
FFB	linear	high	low (*)
CQT	geometric	low	high
BQT	linear-geometric	low	medium
BQFFB	linear-geometric	high	medium (*)

It may be pointed out that classic analog-domain design techniques are unsuitable for this type of application. This is due to the fact that high-selectivity analog filters are of very high order, which would lead to digital filters with a prohibitively large number of non-zero coefficients. In addition, the resulting digital structure would tend to exhibit severe numerical problems, thus not allowing any kind of fast implementation.

The remaining of this paper is organized as follows: Section II introduces the BQFFB algorithm and Section III presents a complete analysis of its computational complexity. Section IV discusses the practical issues associated to the frequency organization of the BQFFB channels. Section V describes some computer experiments illustrating the proposed filter bank properties. Finally, Section VI concludes the paper, emphasizing its main contributions.

II. The BQFFB Algorithm

The BQFFB structure was conceived as an attempt to improve the spectral analysis in a musical context. In the proposed algorithm, instead of calculating all the channels in a geometric way, just the octaves have their geometric spacing maintained. Inside each octave, the channels are linearly distributed, in order to decrease the computational cost.

A numerical example follows for the sake of a better understanding of the bounded-Q scheme. For a sampling frequency of 44.1 kHz, the highest octave ranges from 11.025 kHz to 22.050 kHz, the second highest octave ranges from around 5.512 kHz to 11.025 kHz, and so on, following a geometric pattern, as expected. In the next step, each octave is divided into N' equally spaced channels. If $N' = 32$, for

instance, then each channel would be around 344 Hz wide inside the highest octave, 172 Hz wide inside the second-highest octave, and so on. The linear channel separation allows one to employ an FFT-like algorithm within each octave, thus greatly reducing the overall computational burden of the complete BQFFB algorithm, as further analyzed in Section III.

The following procedure describes the complete implementation of the BQFFB algorithm. Its goal is to divide an input signal $x[n]$ into D octaves and subsequently sub-divide each of these octaves into N' channels.

1) Pass the signal $x[n]$ through an N-channel FFB, which linearly slices the frequency spectrum into N bins ranging from 0 to 2π. To obtain N' channels in a given octave, one must use $N = 4N'$. The reason for this relationship will become clear from the discussion provided in the next step.
2) Store the outputs associated to the frequency range from $\alpha\pi/2$ to π, corresponding to the highest octave being analyzed. The parameter α controls the cut-off frequency for the decimation filter used in the next step. This decimation filter should not be made very steep if one wants to keep its additional computational load acceptable. A good design results from defining its cut-off frequency as $\alpha\pi/2$, with $0 \ll \alpha < 1$. As the FFB spreads the N bins within the whole range $[0, 2\pi]$, when $\alpha \approx 1$ there will be only about $N' = N/4$ channels within the range $[\alpha\pi/2, \pi]$.
3) Submit $x[n]$ to the half-band decimation filter, with a passband given by $[0, \alpha\pi/2]$, thus eliminating the N' frequency components stored in the previous step, and retaining the N' lowest ones.
4) Decimate the output signal by a factor of 2.
5) Apply again the N-channel FFB to the decimated signal, this time storing the results for all channels between $\alpha\pi/2$ and $\alpha\pi$.
6) Repeat steps 3 to 5 for the $(D-3)$ subsequent intermediate octaves, i.e., until the last division of the spectrum is perfomed.
7) For the 2 lowest octaves, after the last spectrum division, store the results corresponding to all channels between 0 and $\alpha\pi$.

III. BQFFB Complexity Analysis

To analyze the computational complexity associated to the BQFFB algorithm, we first proceed to analyze the computational cost of the standard FFB scheme. The structure of the FFB [3] resembles the tree-like organization of the FFT, as seen in Fig. 1. Due to the use of linear-phase filters and to the occurrence of many null coefficients, the N-channel FFB exhibits a low computational cost. For the same filters designed in [3], the number of coefficients per level, according to [9], is presented in Table II. Each FFB lowpass prototype filter has a highpass complementary companion, but the latter does not require additional complex multiplications: the output of the complementary filter can be computed as the difference between the input signal and the output of the prototype filter (see Fig. 1).

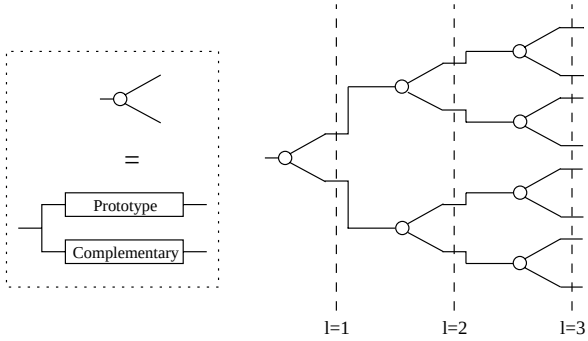

Fig. 1. An 8-Channel 3-Level FFB tree structure showing prototype and complementary filters.

TABLE II
NUMBER OF NON-ZERO COEFFICIENTES PER LEVEL OF THE SUB-FILTER STRUCTURE OF THE FFB.

Cascade level (l)	Distinct coefficients per filter	Prototype filters	Coefficients per level	Accumulated coefficients $C(l)$
1	7	1	7	7
2	6	2	12	19
3	3	4	12	31
4	3	8	24	55
5	2	16	32	87
6	2	32	64	151
7	2	64	128	279
8	2	128	256	535
⋮	⋮	⋮	⋮	⋮
$\log_2 N$	2	$N/2$	N	$2N+23$

From Table II, it can be seen that for an l^{th}-level FFB the number of complex multiplications per sample, is equal to

$$C(l) = 2N + 23 \tag{1}$$

for $l \geq 5$. For practical values of N, this relationship can be approximated by $C(l) \approx 2N$. This indicates that the FFB implementation is actually only slightly more complex than the standard radix-2 FFT.

As explained above, the BQFFB employs one FFB for each of the D octaves being analyzed. The fact that only about $1/4$ of the FFB channels are used per round accounts for a computational overhead of a factor of 4. The signal is decimated $(D-1)$ times. Considering L^{th}-order decimation filters, the number of complex multiplications per sample is

$$\begin{aligned} C_{BQFFB} &= (D-1)(L+1) + 4C(l)D \\ &\approx D(L + 4C(l)), \end{aligned} \tag{2}$$

where $C(l)$ is given by the last column of Table II.

A possible alternative scheme to the BQFFB would be employing all channels geometrically spaced in the frequency domain. Such approach is the so-called CQFFB formerly presented in [9], [10]. Due to its truly geometric nature, the CQFFB is not able to take advantage of FFT-like algorithms in its implementation. A general comparison between the BQFFB and the CQFFB complexities is depicted in Fig. 2. In typical applications using a total of around 100–300 channels, the computational complexity of the BQFFB is about three

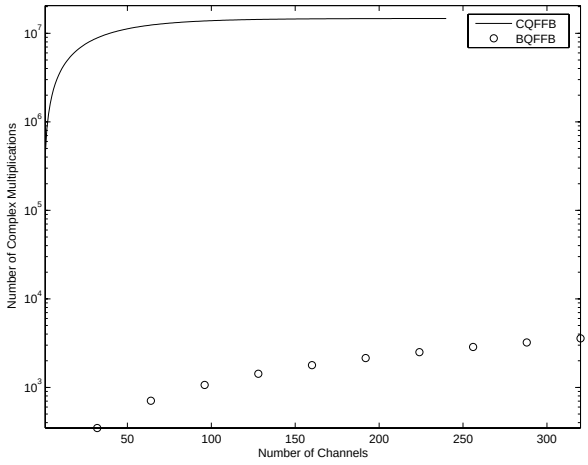

Fig. 2. Complexity comparison between BQFFB (represented by circles) and CQFFB (in solid line).

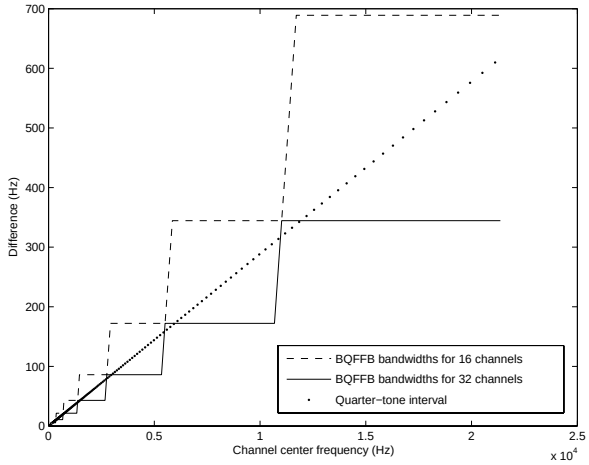

Fig. 3. Difference between central frequencies of contiguous BQFFB channels (considering $N' = 16$ and $N' = 32$ channels per octave) compared with the CQFFB quarter-tone resolution.

orders of magnitude lower than the one required by the CQFFB.

IV. The BQFFB Frequency Resolution

To achieve a reasonable computational complexity, the BQFFB scheme employs a linear-geometric channel distribution in the frequency domain. Such channel distribution does not constitute a handicap for the BQFFB if one employs this scheme with a proper number of channels per octave.

To illustrate this fact, Fig. 3 plots the difference between two adjacent channels, as a function of the central frequency of the first channel, for the BQFFB and the CQFFB methods. In this analysis, the CQFFB is chosen as the reference for being the best competitor among the mentioned methods, in that it combines high selectivity with a reduced number of channels. In that figure, a quarter-tone frequency resolution is used, which would allow one to discriminate two musical notes separated by one half-tone.

For the CQFFB, the channels follow a perfectly geometric pattern represented in Fig. 3 by the dotted line. The two other plots correspond to the BQFFB with $N' = 16$ (dashed line) and $N' = 32$ (continuous line) channels per octave. Both curves present a staircase-like form, since within a certain frequency range (corresponding to one complete octave) the channel spacing is made constant. Clearly, from these plots, while the 16-channel BQFFB does not meet the desired quarter-of-tone resolution, the 32-channel BQFFB surpasses it in practically every channel, at a quite reasonable computational cost. A 32-channel BQFFB thus becomes a preferable choice in this case.

Additionally, one may be reminded that in the tempered scale only the fundamental frequencies are geometrically spaced, whereas their harmonics appear at integer multiples of the corresponding fundamental frequencies. Therefore, the mixed linear-geometric characteristic of the BQFFB attains a good compromise for a complete analysis of real musical signals.

V. Computer Experiments

In this section, some computer simulations are carried out to assess the performance of the proposed BQFFB.

Example 1: First, we compare in a static sense the performances of the FFT, FFB, CQT, and BQFFB algorithms when analyzing one-second test signal formed by 4 pure tones of unitary amplitudes at 185.0, 196.0, 587.3, and 622.3 Hz (corresponding to the notes F#3, G3, D5, and D#5, respectively). These frequencies were chosen as two half-tone pairs placed at distant octaves.

Aiming at the classification of musical notes in half-tones, a quarter-tone resolution would suffice. The corresponding frequency ratio is $2^{1/24}$. If one wants to use constant-Q channels, as in the CQT, the corresponding quality factor is given by [9]

$$Q = \frac{f_k}{(\delta f)_{CQT}} = \frac{f_k}{(2^{1/48} - 2^{-1/48}) f_k} \approx \frac{1}{0.0289} \approx 34, \quad (3)$$

where f_k is the central frequency (in a geometric sense) and $(\delta f)_{CQT}$ is the bandwidth of any given channel k. This value will be adopted in the CQT simulation setup, and will also serve as a reference in choosing the number of channels for the remaining methods.

To keep the comparison fair, the channel with the worst resolution in the linear-spacing tools should satisfy the quarter-tone constraint. This restriction applies to the lowest channel, which must contain the lowest test-tone. To meet these conditions, both FFT and FFB divide the spectrum in 4096 channels from 0 to 22050 Hz (assuming a sampling rate of 44100 Hz), each one 5.94 Hz wide.

By its turn, the BQFFB divides the spectrum from its highest limit in seven octaves plus the remaining lower frequency band (which includes the lowest test-tone). Each of these eight sub-bands is linearly divided into $N' = 32$ channels, thus keeping in the lowest band the same resolution as the FFT and the FFB tools. Finally, besides choosing an eighth-order Chebyshev Type-I lowpass filter as the decimation filter, a parameter $\alpha = 0.8$ (see Section II) was employed.

Figs. 4, 5, 6, and 7 show the responses of the FFT, FFB, CQT, and BQFFB, respectively, to the test signal in example 1. From these figures, it becomes evident that the FFT is not capable to distinguish the peaks associated with the test tones, due to the poor selectivity of the associated filters. In contrast, the FFB is able to detect the peaks with the same unnecessarily large number of channels. The CQT identifies the tones via much less channels, but increasing considerably the computational cost. Since the CQT is based on FFT channels, its inherently poor selectivity makes the tones identification harder, which also becomes clear from the figures. The use of the BQFFB attains the same performance as the FFB with a better channel distribution in the frequency domain.

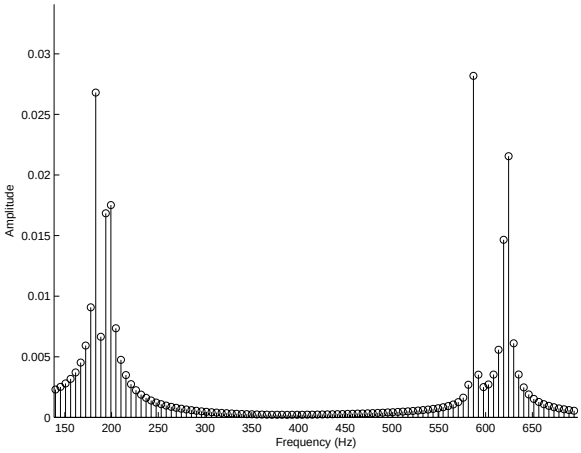

Fig. 4. FFT response to the test signal in example 1.

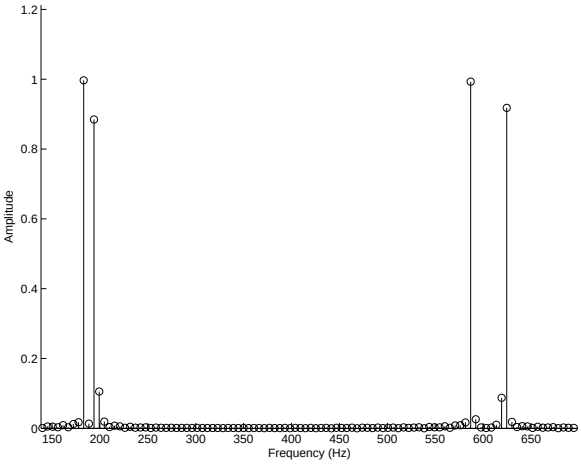

Fig. 5. FFB response to the test signal in example 1.

It is fair mentioning that both the FFB and the BQFFB plots (Figs. 5 and 7, respectively) show the second tone as two peaks. This happens because the channel centers do not necessarily coincide with the tones present in the signal. All the discussed systems are equally suscetible to this effect, which may require a simple correction procedure (e.g. a 3-point interpolation). Naturally, the finer the resolution, the more common this splitting becomes.

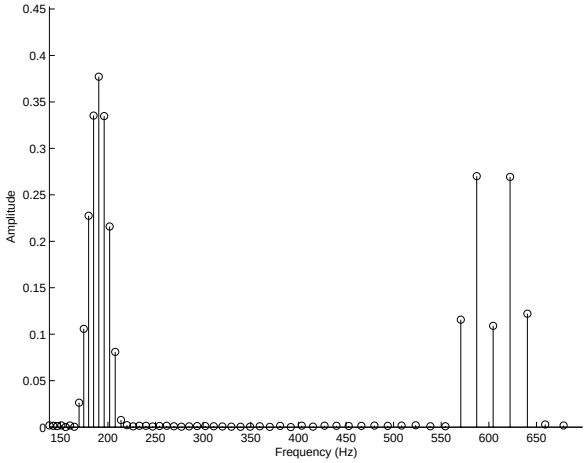

Fig. 6. CQT response to the test signal in example 1.

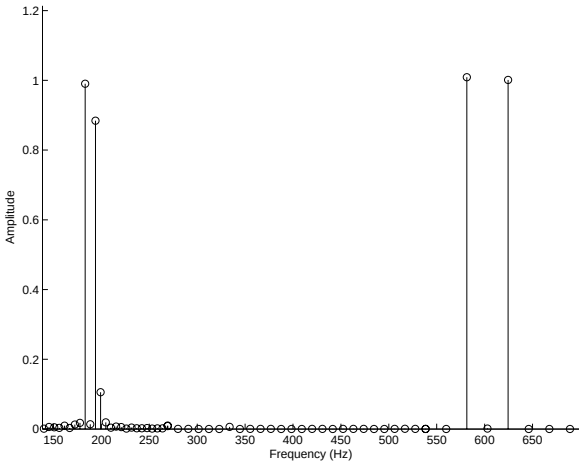

Fig. 7. BQFFB response to the test signal in example 1.

Example 2: Next, the BQFFB dynamic response is assessed. In this case, we analyze two sinusoids of frequencies 263 Hz and 296 Hz, along with their corresponding 2^{nd}, 3^{rd}, and 4^{th} harmonics. To introduce some transient, the 296 Hz sinusoid and associated harmonics appear one second after the 263 Hz components, which has a total duration of two seconds.

Figs. 8, 9, and 10 show in detail the 4^{th}, 5^{th}, and 6^{th} octaves, respectively, of the dynamic response of the BQFFB, with $N' = 32$ channels per octave. From these figures, one notices how all frequencies contained in the test signal can be perfectly identified.

For a better visualization, Fig. 11 shows a spectrogram-like plot called sonogram, in which the vertical axis represents frequency, the horizontal axis represents time, and the grey level indicates the BQFFB output amplitude. In this plot, the darker the shade, the higher is the amplitude level at a particular time and frequency line. From Fig. 11, once again all frequency components contained in the test signal are readily and properly identified.

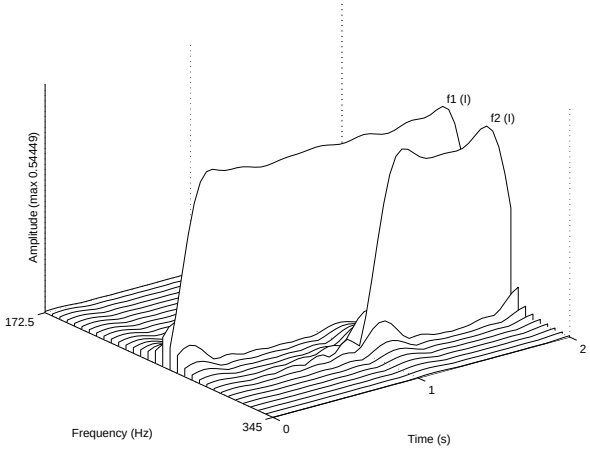

Fig. 8. 4^{th} octave of the BQFFB dynamic response to the test signal in example 2.

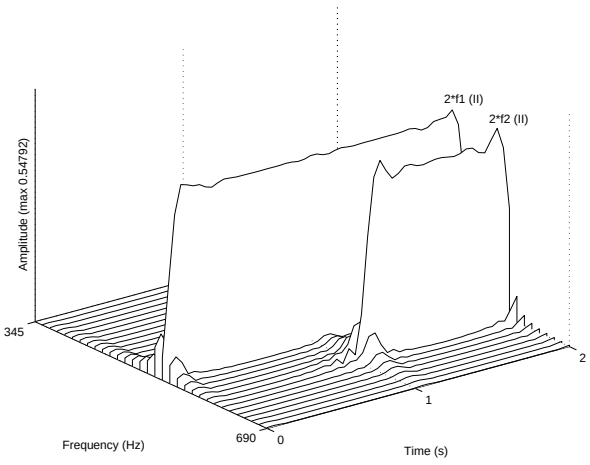

Fig. 9. 5^{th} octave of the BQFFB dynamic esponse to the test signal in example 2.

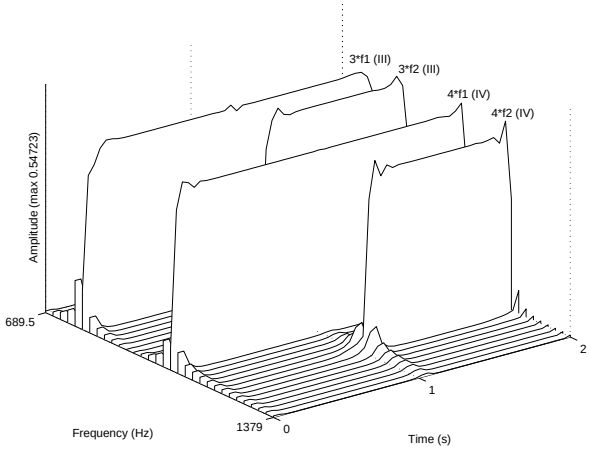

Fig. 10. 6^{th} octave of the BQFFB dynamic response to the test signal in example 2.

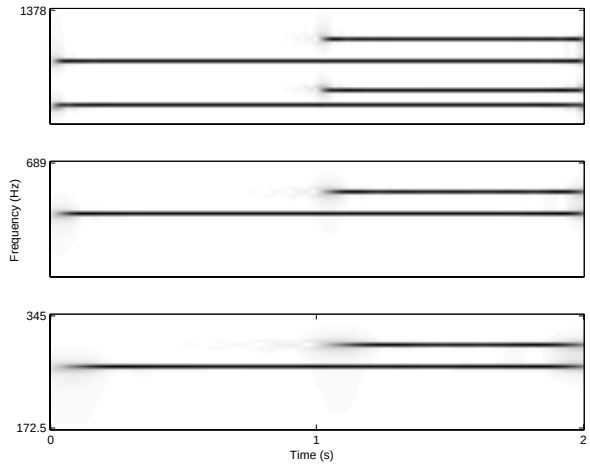

Fig. 11. Sonogram for the BQFFB dynamic response to the test signal in example 2.

VI. Conclusions

This paper described a new spectral analysis tool, the BQFFB, which slices each octave of the spectrum linearly in a fixed number of channels, thus keeping approximately the geometric spacing desirable in musical applications. As to the selectivity, the proposed method compares most favorably with the FFT. In addition, while keeping the same selectivity as the worst-case of the FFB method, the BQFFB employs a much lower number of channels, since these channels are distributed more homogeneously by the BQFFB along the spectrum. Moreover, the BQFFB compares favorably to the formerly proposed CQFFB, by presenting a computational complexity about three orders of magnitude lower. The result is a robust and efficient tool with wide applications in the spectral analysis of musical signals, as illustrated by some simulation examples.

Acknowledgments

The authors would like to thank Cristiano N. dos Santos and Danilo B. Graziosi for their contributions in the early stages of this work.

References

[1] A. Klapuri, "Automatic transcription of music," in *Proc. of SMAC 03 - Stockholm Music Acoust. Conf.*, R. Bresin, Ed., Stockolm, Sweden, August 2003, vol. 2, pp. 587–590.

[2] P. S. R. Diniz, E. A. B. da Silva, and S. L. Netto, *Digital Signal Processing: System Analysis and Design*, Cambridge, Cambridge, UK, 2002.

[3] Y. C. Lim and B. Farhang-Boroujeny, "Fast filter bank (FFB)," *IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 39, no. 5, pp. 316–318, 1992.

[4] Y. C. Lim and L. J. Wei, "Matrix formulation: fast filter bank," in *Proc. of ICASSP 2004 - Int. Conf. Audio, Speech, and Signal Processing.*

[5] B. Scharf, "Critical bands," in *Foundations of Modern Auditory Theory*, J. V. Tobias, Ed., vol. 1, chapter 5, pp. 157–202. Academic, New York, NY, USA, 1970.

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

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

[8] K. L. Kashima and B. Mont-Reynaud, "The bounded-Q approach to time-varying spectral analysis," Tech. Rep. STAN-M-28, Stanford University, Dep. of Music, 1985.

[9] D. B. Graziosi, C. N. dos Santos, S. L. Netto, and L. W. P. Biscainho, "A constant-Q spectral transformation with improved frequency response," in *Proc. of ISCAS 2004 - Int. Symp. Circuits and Systems*, Vancouver, Canada, May 2004, IEEE, vol. 5, pp. 544–547.

[10] C. N. dos Santos, S. L. Netto, L. W. P. Biscainho, and D. B. Graziosi, "A modified constant-Q transform for audio signals," in *Proc. of ICASSP 2004 - Int. Conf. Audio, Speech, and Signal Processing.*