INVERSION OF SHORT-TIME FOURIER TRANSFORM MAGNITUDE SPECTROGRAMS WITH ADAPTIVE WINDOW LENGTHS

Volker Gnann and Martin Spiertz

Institut für Nachrichtentechnik
RWTH Aachen University
D-52056 Aachen, Germany
{gnann, spiertz}@ient.rwth-aachen.de

ABSTRACT

In this paper, we extend the Real-Time Iterative Spectrogram Inversion method (RTISI) for generating a time-domain audio signal from a magnitude spectrogram such that it can handle changing spectrogram window lengths. For each desired window length, we use a separate buffer structure and synchronize the buffers each time the window length changes. This way, the proposed method helps to improve the time/frequency-resolution trade-off for algorithms that operate on magnitude-only spectra.

Index Terms— Short-time Fourier transform, phase estimation, magnitude spectrum inversion, window switching

1. INTRODUCTION

Magnitude spectrogram inversion, also known as phase estimation, has a wide range of applications, such as time/pitch scale modification or source separation. Until now, all phase estimation algorithms require a fixed window length for the short-time Fourier transform (STFT). This implies a constant trade-off between the time and frequency resolution due to this constant window length and Heisenberg's uncertainty principle [1]. As a rule of thumb [2], we can characterize audio signals as large parts with slow changes, interrupted by short parts of sudden changes (transients). When using the STFT, the choice of the window length is critical: For non-transient audio signal parts, a high frequency resolution is more important than a high temporal resolution, thus a long window is preferred. Audio parts containing transients need a high temporal resolution, so shorter window lengths better suit to them. For these reasons, window switching, first introduced in [3], has become an important method to adapt this trade-off to the signal in audio coding.

In this paper, we present an approach for signal reconstruction from magnitude spectrograms with different window lengths. It extends the Real-Time Iterative Spectrogram Inversion method (RTISI, [4]) by using two buffers, one for each window length. When a sequence of spectra occurs, the phase estimator copies the audio data from the long-window buffer to the short-window buffer. Then, the estimation is performed on the short-window buffer. The phase estimation result is transferred to the long-window buffer such that the overlap-add property is preserved.

This paper is organized as follows: In Section 2, we describe the generation of spectrograms with window switching that can be handled by the proposed RTISI modification. In Section 3, we present the modifications that enable RTISI to work with different window lengths. The paper closes with an example and an outlook.

2. GENERATION OF WINDOW-SWITCHED SPECTROGRAMS

Audio coding standards like Advanced Audio Coding (AAC, [5]) typically use the overlap-add method to reconstruct audio signals from the decoded time frames. In order to achieve perfect reconstruction, the overlap-add method requires the overlapping window functions to sum up to a constant. To ensure this overlap-add property, AAC uses bridge windows when the window length changes. For maximum coding efficiency, the window length L is usually twice the distance between the start samples of adjacent time frames S, i.e. $L = 2S$.

The requirements for phase estimation with RTISI are different. As explained in Section 3.1, RTISI basically exploits the overlap between adjacent frames. For that reason, a higher overlap ratio L/S is preferred; a good choice is $L = 4S$ [4]. This makes the construction of bridge windows rather inconvenient. Additionally, bridge windows imply multiple window forms of the same length, which makes the spectral leakage properties more difficult to be predicted. For these reasons, we do not apply bridge windows. Instead, we simply represent every long-window time frame which we consider to have transients by a sequence of overlapping short-window time frames. We fulfill the overlap-add property by introducing a window compensation step within the phase estimator. The overlap ratio L/S whithin a short-window sequence is the same as within long-window sequences. As a result, we can calculate the number N of short spectrograms that we need to represent a long one with

$$N = 1 + \frac{L_{\text{long}} - L_{\text{short}}}{S_{\text{short}}}. \tag{1}$$

As window function $w(n)$, we employ the slightly modified Hamming window as defined in [6]:

$$w(n) = \begin{cases} \frac{2\sqrt{S}}{\sqrt{(4a^2+2b^2)L}}(a + b\cos(2\pi\frac{n}{L})), & \text{if } 1 \le n \le L, \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where $a = 0.54$, and $b = -0.46$. This window fulfills the condition

$$\sum_{m=-\infty}^{\infty} w^2(n - mS) = 1, \forall n, \tag{3}$$

so that perfect reconstruction in the overlap-add step is achieved if $w(n)$ is used as an analysis and as a synthesis window.

The generation process is illustrated in Figure 1. After windowing, we calculate the discrete Fourier transform magnitude for

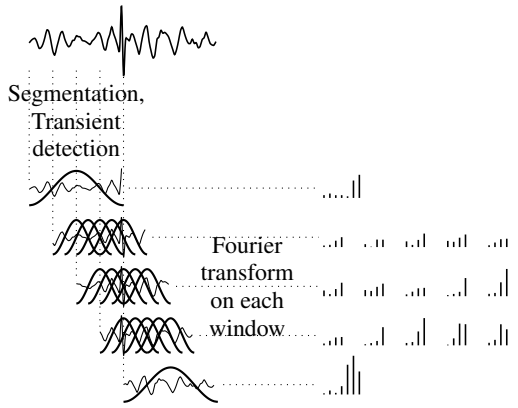

Fig. 1: *Generation of spectrograms using window switching. The first step (segmentation) is equivalent to STFT windowing with a rectangle window. A transient detector decides if a segment is processed with one single long or with multiple overlapping short Hamming windows. After the actual windowing, each windowed segment is transformed into the frequency domain. Note that only the first $\frac{L}{2}+1$ coefficients are needed to characterize a spectrum due to its symmetry for audio signals, which are real-valued.*

each windowed time frame. As usual, we can alter these magnitudes in various ways to implement frequency-domain audio effects. We must keep in mind that we have two different configurations of magnitude coefficients per time frame because of the different window lengths: Either one spectrum with $L_{\text{long}}/2+1$ coefficients or N spectra with each $L_{\text{short}}/2+1$ coefficients. After these modifications, we need a way to re-convert the manipulated magnitudes back to the time domain, as described in Section 3.

The overlap-add property is not fulfilled at the signal boundaries and during the window-switching operation, so we must take special care on this problem when estimating the phase.

3. PHASE ESTIMATION FOR WINDOW-SWITCHED SPECTROGRAMS

As described in the previous section, our phase estimation algorithm is basically an extension of RTISI [4] with two buffers. The two buffers store the same audio data, but analyzed with two window lengths. Since the short window length is only used for transient processing, we consider the use of the long-window-length buffer as the normal mode. The short-window-length buffer processes the transient signals. The buffer structure is illustrated in Fig. 2.

Each buffer can be interpreted as a table with rows and columns. Both of them represent a time axis. Each buffer row represents an STFT frame, storing the current signal estimation in the time domain and (not shown in the illustration) the according Fourier transform magnitude. The buffer rows are synchronized such that each column index represents the same sample index in all rows.

3.1. Standard RTISI processing

To initialize the RTISI buffer, the buffer rows are filled with zeros. If a signal estimation is already known, we window it (using the analysis and synthesis window) and fill the according buffer row with the result instead. Since RTISI is an iterative algorithm, the result quality depends on the initialization.

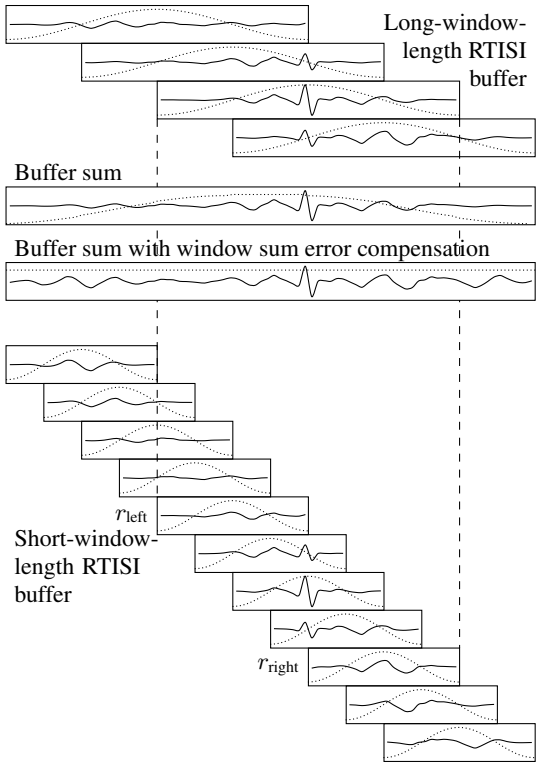

Fig. 2: *Two RTISI buffers with different window lengths. To transport audio data between the buffers, the algorithm calculates the buffer sum, compensates the window sum error, and windows the result for each target buffer row. The dotted lines denote the window function the audio data in the buffer are implicitly multiplied with.*

Let us assume now that the audio data for all rows except the last one have already been estimated. We can estimate the content for the last row (or improve its estimation) by the following procedure:

1. Calculate the sum of all buffer rows and limit it to the part covered by the last row.
2. As explained in [4] with more details, this sum is implicitly windowed with a sum of overlapping analysis *and* synthesis windows. This leads to inconsistencies between time and frequency representation. We call that phenomenon the window sum error. Multiply this sum with $\frac{w(n)}{\sum_{m=1}^{L/S} w^2(n-mS)}$ to compensate the error, so that the sum is implicitly windowed with $w(n)$.
3. Calculate the phase spectrum of the sum using an FFT.
4. Combine each phase of phase spectrum with the corresponding magnitude of the magnitude spectrum stored in the buffer (see [4] for details).
5. Transform this combination into the time domain (using an inverse FFT).
6. Window the result and store it into the last row.

These steps can be iterated several times. After a certain number I of iterations, we commit the frame stored in the last row and synchronize the buffer to the next frame. After synchronization, we check if a signal estimation for the next frame is available. If this is

the case, we initialize the last buffer row with this estimation (after windowing), otherwise with zeros. An overlap-add synthesis step assembles the final audio data from the committed frames.

An important extension for RTISI is the use of look-ahead frames. Here, after the estimation of the last row content, its predecessors are re-estimated backwards, until the number k of look-ahead frames is reached. The k^{th} row (counted from the last) is committed. This extension is explained in more detail in [4].

3.2. Extension to a dual time/frequency resolution

Section 3.1 shows how the phase of audio data is estimated when only one window length is used, i.e. no transients occur. If a transient has occured and been detected, the algorithm notices that via the spectrogram configuration (see Section 2). In this case, the phase estimation works as follows:

1. Copy the audio content from the long-window buffer into the short-window buffer. Within this process, compensate the window sum error for each target row.
2. Find the left and the right short-window buffer rows r_{left} and r_{right} which correspond to the current long-window buffer row that should be estimated in the non-transient case.
3. Estimate the audio data for the rows r_{left} up and including r_{right} as described in Section 3.1 using the short-window buffer.
4. Copy the estimated audio data from the short-window buffer back to the long-window buffer with window sum error compensation.

3.2.1. Buffer size

The audio copy process and the window sum error compensation are illustrated in Figure 2. Since both buffers store the same amount of audio data, the number of columns is the same in both buffers. Thus, the number of short-window buffer rows R_{short} can be calculated from the number of long-window buffer rows R_{long} as

$$R_{\text{short}} = N + (R_{\text{long}} - 1) \cdot \frac{S_{\text{long}}}{S_{\text{short}}}. \tag{4}$$

3.2.2. Audio transfer and window sum error compensation

The audio data stored in the buffer are represented by the sum of the buffer rows, column by column, so we can use the buffer sum to exchange the audio data between the buffers. Since the audio data in the buffer rows are windowed, the buffer sum is also implicitly windowed with the window sum. As described in [7], we compensate this error by dividing the buffer sum element-wise by the window sum. After that, we multiply the buffer sum for each destination row element-wise with the analysis *and* the synthesis window before storing the results in the destination buffer.

3.2.3. Short-window buffer edge row calculation

Let i be the index of the long-window buffer row to process. Then, the first and the last short-window buffer row indices r_{left} and r_{right} are calculated as

$$r_{\text{left}}(i) = i \cdot \frac{S_{\text{long}}}{S_{\text{short}}}, \tag{5}$$

$$r_{\text{right}}(i) = r_{\text{left}}(i) + N - 1. \tag{6}$$

4. EXAMPLE

To show the benefits of window switching also for magnitude-only spectrogram processing, we have mixed the castanet and the double bass arpeggio example from the EBU-SQAM library [8]: Castanets require a high temporal resolution, and double bass requires a high frequency resolution due to its low pitch. From a part of this mixture, we created the magnitude spectra and re-estimated the phase using RTISI with look-ahead with and without window switching. As fixed window lengths, we have chosen 512, 1024, and 2048 samples, respectively, with a sampling frequency of 48 kHz. For window switching, we have chosen window lengths of 512 and 2048 samples. We used a peak detection algorithm as described in [3] for transient detection. In all examples, we set the number of lookahead frames k to 3 and the number of iterations per frame estimation I to 50. As analysis and synthesis window, we used the Hamming window shape (Eq. 2). The results are presented in Figure 3. We can see that the window switching phase estimation approximates the original best. The 512-sample-window-length phase estimation causes distortions in the frequency domain, whereas the 2048-sample window leads to the well-known pre-echo artifacts [2] in the time domain. The most interesting comparison is window switching against the 1024-sample window. We can see that window switching leads to a better time *and* frequency resolution than the fixed medium window length.

To confirm these experiments, we have measured the STFT-magnitude signal-to-error ratio (SER) for these signals. This SER is defined as follows [4]:

$$\text{SER} = 10 \log \frac{\sum_{m=-\infty}^{\infty} \int_{\Omega=-\pi}^{\pi} |X(mS, \Omega)|^2 d\Omega}{\sum_{m=-\infty}^{\infty} \int_{\Omega=-\pi}^{\pi} (|X(mS, \Omega)| - |X'(mS, \Omega)|)^2 d\Omega} \tag{7}$$

This SER measure operates on STFT magnitudes and thus depends on its own STFT window length. The results are given in Table 4. It should be noted that, in principle, RTISI bases on the Griffin and Lim [6] iteration scheme which maximizes the SER function (7). As a consequence, the SER is best if the SER window length equals the window length used by RTISI. To take this effect into account, we additionally measured the examples with a SER window length L of 500, 1000, and 2000 samples additionally to 512, 1024, and 2048. In all cases, we set the hop size S for the SER measure to $L/4$. We can see that, in our example, window switching outperforms the fixed-window RTISI for all SER measure window lengths except 1024 and 2048, where the direct optimization effect occurs as described above.

Table 1: *SER measurements for the examples in Figure 3. For the switching experiments, the RTISI window lengths are 2048 (normal mode) and 512 (transient mode) samples. The SER measure window lengths of 500, 1000, and 2000 samples are chosen to compensate the direct SER optimization which RTISI performs if the SER measure window length equals the RTISI window length.*

SER measure window length (samples)	Spectrogram window length (samples)			Switched windows
	512	1024	2048	
500	16.27	13.48	11.69	**24.00**
512	18.37	13.86	11.88	**24.73**
1000	14.80	24.05	18.07	**24.87**
1024	14.79	**26.10**	18.23	24.72
2000	11.21	22.14	21.63	**22.35**
2048	11.14	22.18	**24.58**	22.60

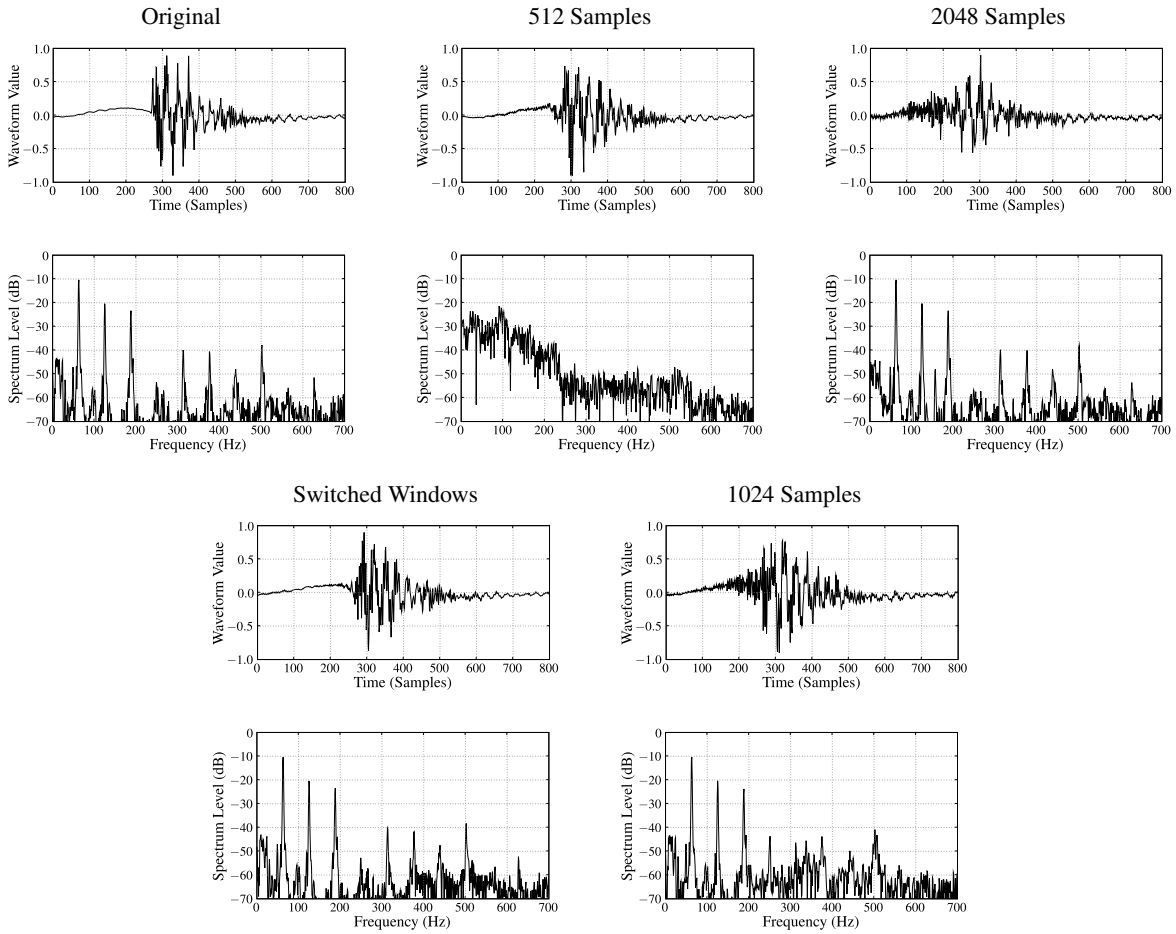

Fig. 3: *Waveforms and spectrum plots for a mix of double bass and castanets. For different window lengths, we calculated the magnitudes and estimated the phases. The lengths for the switched window example were 512 and 2048 samples. The signal part chosen for frequency analysis was one second long; from this second, an 800-sample long castanet beat is plotted in the time domain.*

5. CONCLUSIONS AND OUTLOOK

We have presented a method to invert spectrograms with different window lengths. This method allows to combine the advantage of window switching with algorithms that operate on magnitude spectra only. This opens a new way to improve the quality of applications working in the frequency domain, like denoising, time/pitch scale modification, or comb-filter free audio mixing.

The algorithm can easily be extended to more than two buffers, which allows the use of more than two window lengths. However, this would require a technique to determine the optimal time/frequency resolution optimum, which is an interesting topic for future research.

6. REFERENCES

[1] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, 2. edition, 1999.

[2] T. Painter and A. Spanias, "Perceptual Coding of Digital Audio," *Proceedings of the IEEE*, vol. 88, no. 4, pp. 451–513, April 2000.

[3] B. Edler, "Coding of Audio Signals with Overlapping Block Transform and Adaptive Window Functions," *Frequenz*, vol. 43, no. 9, pp. 252–256, 1989, in German.

[4] X. Zhu, G. Beauregard, and 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, pp. 1645–1653, 2007.

[5] M. Bosi, K. Brandenburg, S. Quackenbush, L. Fiedler, K Akagiri, H. Fuchs, M. Dietz, J. Herre, G. Davidson, and Y. Oikawa, "ISO/IEC MPEG-2 Advanced Audio Coding," *Journal of the Audio Engineering Society*, vol. 45, no. 10, pp. 789–814, 1997.

[6] D. Griffin and J. Lim, "Signal Estimation From Modified Short-Time Fourier Transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 2, pp. 236–243, 1984.

[7] V. Gnann and M. Spiertz, "Comb-filter Free Audio Mixing Using STFT Magnitude Spectra and Phase Estimation," in *Proc. Int. Conference of Digital Audio Effects DAFx* , 2008.

[8] European Broadcasting Union, "Sound Quality Assessment Material," Tech 3253, 1988.