Audio Engineering Society

Convention Paper 9416

Presented at the 139th Convention
2015 October 29–November 1 New York, USA

Lossless Audio Checker: A Software for the Detection of Upscaling, Upsampling and Transcoding in Lossless Musical Tracks

Julien Lacroix[1], Yann Prime[1], Alexandre Remy[1], and Olivier Derrien[2]

[1] *Independent developer*

[2] *University of Toulon / CNRS-LMA*

Correspondence should be addressed to Julien Lacroix and Olivier Derrien (julien_lacroix111@msn.com, derrien@lma.cnrs-mrs.fr)

ABSTRACT

Internet music dealers currently sell "CD-Quality" tracks, or even better ("Studio-Master"), thanks to lossless audio coding formats (FLAC, ALAC). However, a lossless format does not guarantee that the audio content is what it seems to be. The audio signal may have been upscaled (increasing the resolution), upsampled (increasing the sampling rate) or even transcoded from a lossy to a lossless format. In this paper, we describe a new software that analyzes lossless audio tracks and detects upsampling, upscaling and transcoding (only for AAC in this early version). Validation tests over a large music database (with groundtruth available) show that this method is fast and accurate: 100% of success for upscaling and transcoding, 91.3% for upsampling.

1. INTRODUCTION

In the last years, lossless musical tracks have become very popular on the Internet, thanks to lossless coding formats (FLAC [1], ALAC [2] ...). These can provide truly unaltered audio content, like PCM/WAVE [3], but with a significant compression ratio. Older lossy formats (MP3 [4], AAC [5] ...) can reach higher compression ratios, but the audio content is irreparably altered, although this is not always easily audible. Internet music dealers currently sell lossless "CD-Quality" tracks (typically 16 bits / 44.1 kHz) or even "Studio-Master" (typically 24 bits / 96 kHz). However, a lossless audio format does not guarantee that the audio content is a real CD-Quality or Studio-Master track. Indeed, it is easy to artificially increase the resolution (upscaling), the sample rate (upsampling) or even transcode from a lossy to a lossless format. This also potentially applies to CD/DVD Audio [6] and SACD [7]. So this is a real issue for consumers as well as for music dealers.

In this paper, we describe a new software called Lossless Audio Checker. It analyses any lossless audio track and is able to detect upscaling, upsampling and transcoding (only for AAC in this early version).

The paper is oganized as follows. In section 2, we present the key concepts of high-quality digital audio. In section 3, we describe our upscaling, upsampling and transcoding detection methods. In section 4, we report the results and validation tests. Finally, we conclude in section 5.

2. BACKGROUND

In this section, we recall the main concepts used in this paper.

Uncompressed audio: This means a digital audio signal as it could be observed at the output of an Analog to Digital Converter (ADC) [11]. It is also known as Pulse Coded Modulation (PCM), and the usual containers for such signals are WAVE and AIFF files.

Lossy audio: This corresponds to an uncompressed signal that have been transformed by a compression algorithm in order to reduce the size of the audio file. These algorithms are usually called perceptual coders because they generate a distortion which is supposed to be unperceptible or hardly perceptible. However, the signal is irreparably altered. The usual algorithms are MP3, AAC, WMA [8], AC3 [9], DTS [10].

Lossless audio: Similarly to lossy audio, the uncompressed audio have been transformed by a compression algorithm, but without distortion. Thus, the compression is totally reversible, and there is no signal alteration. The usual algorithms are: FLAC, ALAC, APE [12], WavPack [13], MLP [14].

Sampling rate: In uncompressed digital audio, the time-domain is quantized in a linear way. Assuming that the signal is made of N samples per seconds, the sampling rate (or frequency) is N Hz. The higher the sampling rate, the higher the bandwidth. With Compact Discs (CD), the sampling rate is 44.1 kHz. Another usual value is 48 kHz. "High-Definition" audio use sampling rates of 88.2 kHz, 96 kHz and 192 kHz.

Resolution: This corresponds to the bit-depth used to encode the samples. The higher the resolution, the higher the theoretical dynamics and the lower the theoretical background noise. The usual value is 16 bits per sample. "High-Definition" audio use 24 or 32 bits per sample.

Channels: A channel corresponds to a monophonic signal that will be sent to a single loudspeaker in the acoustic reproduction system. Stereo audio has 2 channels (Left / Right), surround audio has 6 or 8 channels (Front Left / Front Right / Center / Rear Left Effects / Rear Right Effects / Low Frequency Effects (LFE), and optional channels are Front Left Effects / Front Right Effects) [15]. Ususally, the LFE is noted after a dot. Thus, 2.0 denotes stereo audio without LFE, and 5.1 / 7.1 denote five-channel / seven-channel surround audio plus LFE.

Specification of the audio format: The notation that we use in this paper is as follows: Resolution / Sampling rate / Channels. For instance, CD audio is denoted 16 bits / 44.1 kHz / 2.0.

3. METHODS

Several transformations can be performed on audio files. Some do not degrade the audio quality, and some do. Increasing the resolution or the sampling rate never degrades the quality. For instance, converting a CD track (16 bits / 44.1 kHz / 2.0) to Studio Master (24 bits / 192 kHz / 2.0) will result in exactly the same audio quality. But this is possibly an issue, because the audio format allows high performance in terms of bandwidth and signal-to-noise ratio, and the effective performance are the same as with the original CD track. In other words, the audio quality is not as good as expected. The opposite situation is not an issue: Converting from Studio Master (24 bits / 192 kHz / 2.0) to CD quality (16 bits / 44.1 kHz / 2.0) will degrade the quality, but there is no mismatch between the final quality and the audio format.

Another important issue is the use of compression / decompression algorithms: Converting uncompressed audio to lossy audio will degrade the audio quality. Converting lossy audio to uncompressed audio will result in the same audio quality *with respect* *to the quality of the lossy audio file.* Thus, applying a double-conversion (uncompressed to lossy and lossy to uncompressed) will globally result in a degradation of the audio quality. This situation is clearly an issue: The audio file appears to be an original lossless file, whereas the audio quality was degraded by the compression algorithm.

The aim of this software is, given an uncompressed audio file, to detect all possible mismatches between the expected audio quality and the effective audio quality. In this section, we present the transformations that can cause such mismatches and explain the strategy to detect these transformations.

3.1. Upscaling

The upscaling process consists in converting a lossless audio track A bits / B kHz / C into another lossless audio track D bits / B kHz / C with D > A.

In an upscaling process, the number of possible quantization levels is increased, but some are never used. Thus, the set of actually-used quantization levels does not match the audio format. To detect upscaling, our algorithm checks the number of actually used quantization levels and compares it to the levels allowed by the resolution of the audio format. For instance, if the analyzed audio file has a resolution of 24 bits, we check if at least one sample belongs to the 24 bits scale and not to the 16 bits scale. Otherwise, the file could have been quantized with a lower bit depth.

3.2. Upsampling

The upsampling process consists in converting a lossless audio track A bits / B kHz / C into another lossless audio track A bits / D kHz / C with D > B.

In an upsampling process, some new samples are added between the original samples and a interpolation filter is applied to ensure smooth transitions. It is often easier to characterize upsampling in the Fourier domain because the signal bandwidth is half the sampling rate. Thus, increasing the sampling rate also increases the theoretical bandwith, but the Power Spectral Density (PSD) in the extended frequency-band is equal to the PSD of the quantization error, or eventually to a residual signal depending on the frequency response (FR) of the interpolation filter and the PSD of the original audio signal, if higher than the PSD of quantization error [16]. This can occur with very high resolutions (typically 32 bits float). To detect upsampling, our algorithm applies a windowed Short-Time Fourier Transform (STFT) and checks the energy in the high-frequency band. For instance, for a quantization on 16 bits and a sampling rate of 48 kHz, we check if the PSD between 22.05 kHz and 24 kHz is higher than the quantization-noise level corresponding to 16 bits per sample. If not, the file could have been sampled at a lower rate.

As an illutration, we present in figures 1, 2 and 3, the spectrograms (i.e. time-frequency energy images) of, respectively:

1. A 24 bits / 96 kHz / 2.0 synthetically generated white noise coded in PCM/WAVE: The full frequency range (48 kHz) is exploited (no traces of upsampling).

2. The previous signal downsampled to a 24 bits / 48 kHz / 2.0 file: The full frequency range (24 kHz) is also exploited.

3. The previous signal finally upsampled to a 24 bits / 96 kHz / 2.0 file: The full frequency range (48 kHz) is not exploited: the signal bandwidth is limited to 24 kHz. Above, the PSD is very low, approximately the PSD of the quantization noise at 24 bits. This is a typical case of upsampled signal.

3.3. Transcoding

The transcoding process consists in converting a lossy audio track into a lossless audio track by applying a decompression algorithm. Compression algorithms operating at low bitrate generate easily detectable artifacts (birdies and/or transient smearing). Compression can also generate zeros in the spectrogram which almost never occur with uncompressed audio. However, at medium and high bitrates, such artifacts are almost undetectable. Objective audio quality assessment methods are able to distinguish compressed audio for the original file, but the original file must be known, which is not the case here (it is a full-blind case), thus these methods do not apply.

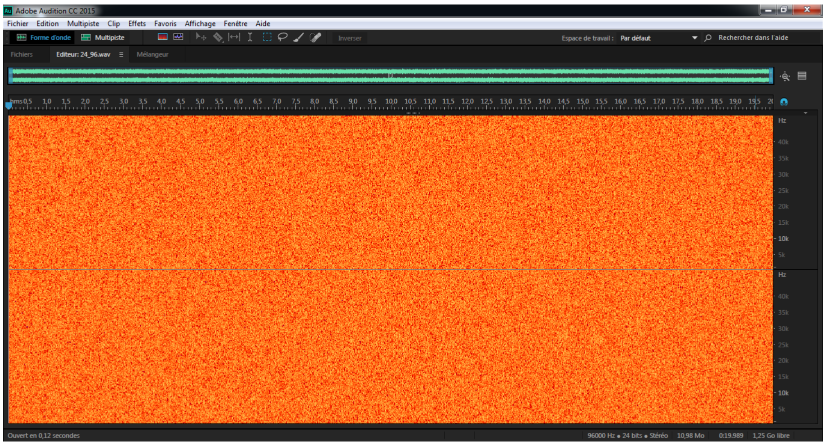

Fig. 1: White noise, 24 bits / 96 kHz / 2.0

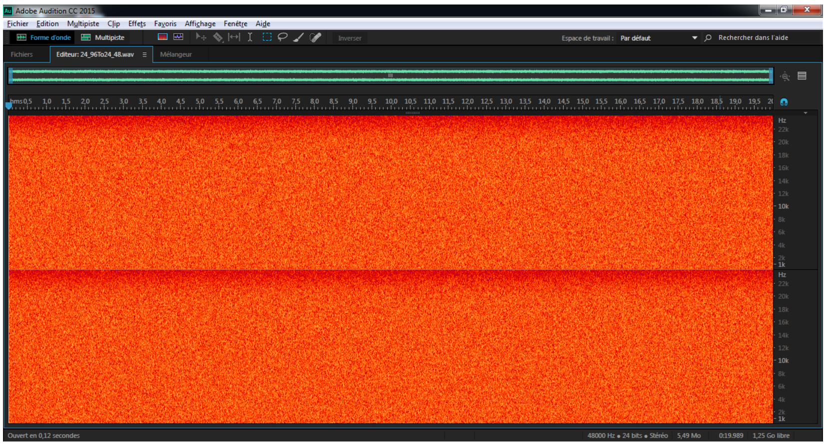

Fig. 2: White noise, 24 bits / 96 kHz / 2.0 to 24 bits / 48 kHz / 2.0

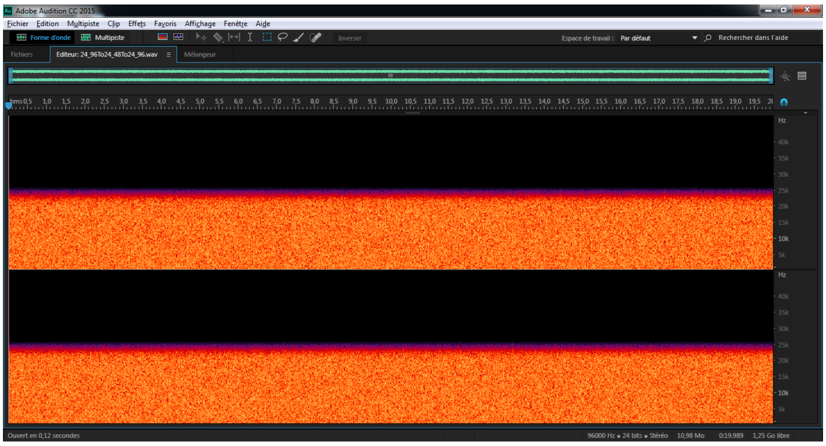

Fig. 3: White noise, 24 bits / 96 kHz / 2.0 to 24 bits / 48 kHz / 2.0 to 24 bits / 96 kHz / 2.0

So, we developed an algorithm that searches for the traces of re-quantization performed by the compression algorithm. This method can be very efficient, but is specific to one coding algorithm. If one wants to check for several coders, one algorithm must be applied for each codec. In this early version, we implemented the detection of MPEG-AAC coder (used for producing M4A and MP4 audio files). Note that AAC and MP3 are very similar, so the same method will be adapted for MP3 very soon.

More precisely, a lossy audio codec is mainly composed of a filterbank followed by a re-quantizer, which generates the distortion. The filterbank is defined by the coding standard, and the re-quantizer is tuned for each audio signal according to perceptual considerations (this implies a hearing model in the coder). For AAC and MP3, the re-quantizing parameters are called scale-factors, which are set for pre-defined frequency-subbands. These parameters are unknown in the full-blind case. Our algorithm applies the same analysis filterbank as in the coder we want to detect. This implies to re-synchronize both filterbanks (in the coder and in the detector): We test all possible time-onsets (1024 possible values for AAC), all analysis-window configurations (4 possibilities), and all stereo matrixing schemes (2 possibilities). Then, we perform a re-quantization of transform coefficients in each analysis window and each coding subband. If the audio signal was previously encoded and if the scalefactor in the detector is exactly the same as in the coder, the re-quantization error should be zero. In other cases, especially for uncompressed audio, the error is usually non-zero and has a specific statistical behavior than can be modeled. Practically, it is impossible to test all possible scalefactor values. Thus, we perform a random sampling on the most probable values for the scalefactors, and threshold the re-quantization error. The choice of the threshold value is made according to probabilistic considerations. When the thresholded error level is higher than a theoretical value corresponding to the uncompressed audio case, the file is marked “transcoded”. Note that transcoding detection only applies for sampling rates lower than or equal to 48 kHz. The workflow for this algorithm is presented on figure 4.

3.4. Detection workflow

Our software detection workflow is illustrated on fig-

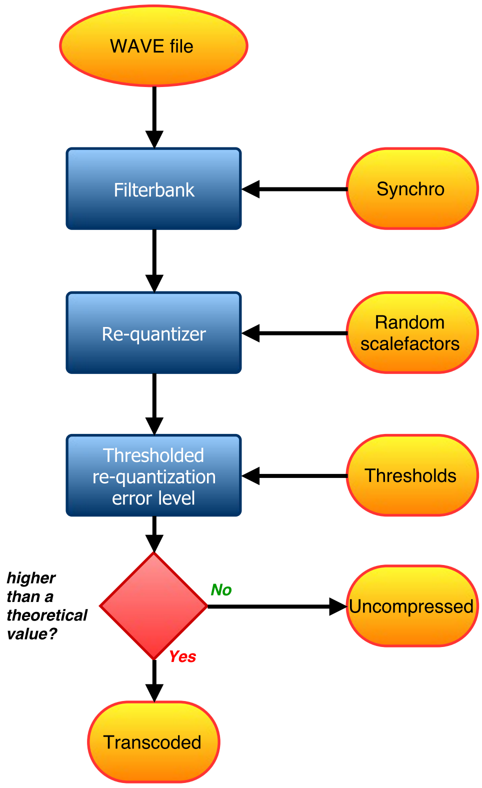

Fig. 4: Transcoding detection workflow

ure 5. It can be described as follows:

1. A WAVE file is taken as input,
2. If the number of channels is lower than or equal to 2 and if the sampling rate is lower than or equal to 48 kHz,
   (a) The transcoding detection is applied,
   (b) If the return value is "true", the file is tagged "transcoded".
3. The upscaling detection is applied,
4. If the return value is "true", the file is tagged "upscaled".
5. The upsampling detection is applied,
6. If the return value is "true", the file is tagged "upsampled".
7. Otherwise, the file is tagged "clean".

4. VALIDATION TESTS

4.1. Trusted pool

The first step in the validation of our detection algorithm consists in evaluating the effective values of false negatives (files that have been upscaled, upsampled or transcoded and that are marked "clean") and false positives (true lossless files that are marked "upscaled", "upsampled" or "transcoded"). Other cases correspond to accurate decisions. Our methodology is as follows: We generate true lossless audio files and then obtain upscaled, upsampled or transcoded versions by applying these transformations. In real life, it is impossible to be 100% sure that an audio file is truly lossless: Even if the recording, mixing and mastering stages have been performed with high-end equipment, it is always possible that one single component limits the performance of the overall audio chain (e.g. a microphone with a limited bandwidth). We found out that the most reliable way to get true lossless audio files is to generate test signals directly in the digital domain using a computer. Our files have been generated in floating point format at each sampling rate using Matlab R2013.a, and the quantization was performed during the WAVE export, using the "audiowrite" function.

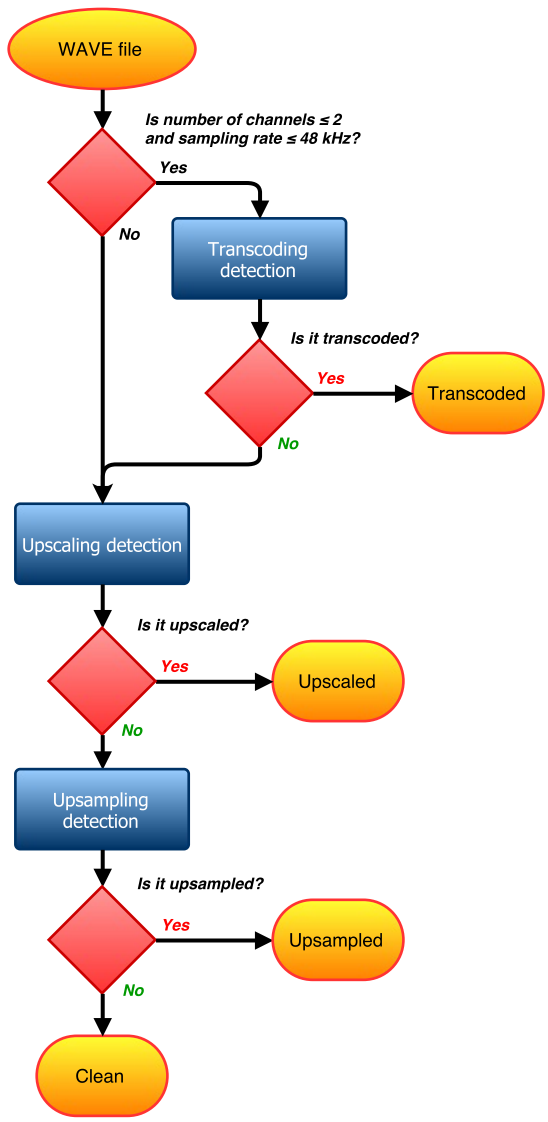

Fig. 5: Lossless Audio Checker detection workflow

4.1.1. Experimental protocol and results

Table 1 lists the 12 synthetic audio files that constitute the trusted pool. In each signal, the full bandwidth is exploited. The signal level is approximately -3 dB FS and the duration is approximately 20 seconds. We know that these files where not upscaled, upsampled nor transcoded.

Table 2 lists all the conversion schemes that we applied to the files in the trusted pool. Note that:

- No channels were removed nor added during the conversions.
- The final outputs were always PCM/WAVE files.
- For each input audio file, there are exactly two conversion schemes that do not change the signal: When the input and output resolution and sampling rate match and when no AAC compression is applied.
- The AAC compression does not change the sampling rate if the input is at 44.1 or 48 kHz. For higher input sampling rates, the output is limited to 48 kHz.
- Upscalings and upsamplings were performed using foobar2000 v1.3.8, while conversions to AAC-LC were performed using iTunes.

Table 3 depicts the final tagging matrix for each combination of input file in the trusted pool (identified in the first column) and conversion scheme (identified in the first line).

4.1.2. Discussion

One can observe that:

- When the transformation scheme does not change the signal, the output files are all tagged “clean”, which is correct.
- When the transformation scheme reduces the resolution or the sampling rate, the output files are all tagged “clean”, which is correct.
- When the transformation scheme applies an AAC compression, the output files are all tagged “transcoded”, which is correct.
- When the input files have a resolution lower than 32 bits float, upscaling and upsampling are correctly detected.
- When the input files have a resolution of 32 bits float, upsampling is not detected.

In other words, our method does not generate any false-positives on the files in the trusted pool. Some false-negatives are observed when the input signal has a resolution of 32 bits float, and when the transformation scheme performs an upsampling.

These cases of false-negatives can be explained as follows: The resolution of input and output files is very high, which means that the level of quantization error in the extended bandwidth is very low. Thus, after upsampling, the extended band is filled by a residual signal depending on the FR of the interpolation filter and on the PSD of the original signal [16]. However, in this case, it seems hard to make a correct decision for any audio file: How can one be sure that this residual signal does not correspond to true audio data?

The Matlab code used to generate synthetic files, the audio files in the trusted pool and the converted audio files are available online at:

`http://losslessaudiochecker.com/tests/AES_139_Trusted_Pool.zip`

4.2. Presumed trusted pool

The second step in the validation of our detection algorithm consists in evaluating the results on a pool composed of 892 lossless audio files (16 bits / 44.1 kHz / 2.0), corresponding to different music styles and genres: pop/rock, house/trance, chanson, folk, alternative and hard rock. We made sure that, when no transformation scheme is applied, all files are tagged ”clean”. Thus, these files are presumed to be authentic lossless audio files.

We applied 3 transformation schemes to the files in this pool:

1. Upscaling to 24 bits and 32 bits float.
2. Upsampling to 48, 88.2 and 96 kHz.
3. Coding to AAC (at 128, 160, 256 and 320 kbps) and transcoding back to PCM/WAVE.

ID	Filename	Signal	Specifications
A	PinkNoise_-3dBFS_20s_16_44.wav	Pink noise	16 bits / 44.1 kHz / 2.0
B	PinkNoise_-3dBFS_20s_16_96.wav	Pink noise	16 bits / 96 kHz / 2.0
C	PinkNoise_-3dBFS_20s_24_44.wav	Pink noise	24 bits / 44.1 kHz / 2.0
D	PinkNoise_-3dBFS_20s_24_96.wav	Pink noise	24 bits / 96 kHz / 2.0
E	PinkNoise_-3dBFS_20s_32f_44.wav	Pink noise	32 bits foat/ 44.1 kHz / 2.0
F	PinkNoise_-3dBFS_20s_32f_96.wav	Pink noise	32 bits float / 96 kHz / 2.0
G	WhiteNoise_-3dBFS_20s_16_44.wav	White noise	16 bits / 44.1 kHz / 2.0
H	WhiteNoise_-3dBFS_20s_16_96.wav	White noise	16 bits / 96 kHz / 2.0
I	WhiteNoise_-3dBFS_20s_24_44.wav	White noise	24 bits / 44.1 kHz / 2.0
J	WhiteNoise_-3dBFS_20s_24_96.wav	White noise	24 bits / 96 kHz / 2.0
K	WhiteNoise_-3dBFS_20s_32f_44.wav	White noise	32 bits float / 44.1 kHz / 2.0
L	WhiteNoise_-3dBFS_20s_32f_96.wav	White noise	32 bits float / 96 kHz / 2.0

Table 1: List of synthetic files in the trusted pool

ID	AAC compression	Output resolution	Output sampling rate	Filename suffix
1	No	16 bits	Same as input	To16_XX
2	No	24 bits	Same as input	To24_XX
3	No	32 bits float	Same as input	To32f_XX
4	No	Same as input	44.1 kHz	ToXX_44
5	No	Same as input	48 kHz	ToXX_48
6	No	Same as input	88.2 kHz	ToXX_88
7	No	Same as input	96 kHz	ToXX_96
8	No	Same as input	192 kHz	ToXX_192
9	128 kbps	16 bits	Same as input, $\leq$ 48 kHz	ToAAC_128To16_XX
10	160 kbps	16 bits	Same as input, $\leq$ 48 kHz	ToAAC_160To16_XX
11	256 kbps	16 bits	Same as input, $\leq$ 48 kHz	ToAAC_256To16_XX
12	320 kbps	16 bits	Same as input, $\leq$ 48 kHz	ToAAC_320To16_XX

Table 2: List of conversion schemes performed on the files in the trusted pool

<table>
<tr><th>ID</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th></tr>
<tr><td>A</td><td>Clean</td><td colspan="2">Upscaled</td><td>Clean</td><td colspan="4">Upsampled</td><td colspan="4">Transcoded</td></tr>
<tr><td>B</td><td>Clean</td><td colspan="2">Upscaled</td><td colspan="4">Clean</td><td>Upsampled</td><td colspan="4">Transcoded</td></tr>
<tr><td>C</td><td colspan="2">Clean</td><td>Upscaled</td><td>Clean</td><td colspan="4">Upsampled</td><td colspan="4">Transcoded</td></tr>
<tr><td>D</td><td colspan="2">Clean</td><td>Upscaled</td><td colspan="4">Clean</td><td>Upsampled</td><td colspan="4">Transcoded</td></tr>
<tr><td>E</td><td colspan="3">Clean</td><td>Clean</td><td colspan="4">Clean</td><td colspan="4">Transcoded</td></tr>
<tr><td>F</td><td colspan="3">Clean</td><td colspan="4">Clean</td><td>Clean</td><td colspan="4">Transcoded</td></tr>
<tr><td>G</td><td>Clean</td><td colspan="2">Upscaled</td><td>Clean</td><td colspan="4">Upsampled</td><td colspan="4">Transcoded</td></tr>
<tr><td>H</td><td>Clean</td><td colspan="2">Upscaled</td><td colspan="4">Clean</td><td>Upsampled</td><td colspan="4">Transcoded</td></tr>
<tr><td>I</td><td colspan="2">Clean</td><td>Upscaled</td><td>Clean</td><td colspan="4">Upsampled</td><td colspan="4">Transcoded</td></tr>
<tr><td>J</td><td colspan="2">Clean</td><td>Upscaled</td><td colspan="4">Clean</td><td>Upsampled</td><td colspan="4">Transcoded</td></tr>
<tr><td>K</td><td colspan="3">Clean</td><td>Clean</td><td colspan="4">Clean</td><td colspan="4">Transcoded</td></tr>
<tr><td>L</td><td colspan="3">Clean</td><td colspan="4">Clean</td><td>Clean</td><td colspan="4">Transcoded</td></tr>
</table>

Table 3: Tagging matrix for each combination of input file in the trusted pool (identified in the first column) and conversion scheme (identified in the first line)

Checking each file takes 2 minutes in the worst case on a three-years-old laptop (4 threads, operating at 2 GHz).

The results are as follows:

1. 100 % of upscaled files were correctly tagged "upscaled".
2. 91.3 % of upsampled files were correctly tagged "upsampled" (i.e. 8.7 % of false-negatives).
3. 100 % of transcoded files were correctly tagged "transcoded", for any bitrate.

Again, our algorithm exhibits a small ratio of false-negatives on the detection of upsampling. This time, the high resolution of the output format cannot explain these errors. According to us, the explanation lies in the design of the upsampler, and especially in the design of the interpolation filter: On some signals, it can produce a residual signal which has a PSD higher than the quantization noise.

Our results and summary for trusted and presumed trusted pools are available online at:

`http://losslessaudiochecker.com/tests/AES_139_Results.zip`

5. CONCLUSION

The Lossless Audio Checker software is totally novel in the sense that it is the first software which allows to measure the real quality of a lossless music track, in a fast and reliable way. Tested on a wide audio database, it appears that the detection of upscaling and transcoding (for AAC codec) is totally reliable, even for high bitrates, and the detection of upsampling is mostly reliable (91.3 %). It seems that these detection errors are related to the design of the upsampler, which is not optimal. However, this problem seems very hard to solve. In a future version, note that other lossy codecs (MP3) will also be considered.

6. REFERENCES

[1] Josh Coalson, "Free Lossless Audio Codec (FLAC) – Format Specification", available online at `http://www.xiph.org/flac/format.html`.

[2] Apple Inc., "Apple Lossless", available online at `http://www.applelossless.com`.

[3] Marina Bosi and Richard E. Goldberg, "Introduction to Digital Audio Coding and Standards – Part I: Audio Coding Methods – Chapter 1: Introduction", Kluwer Academic Publishers, 2003.

[4] ISO/IEC JTC1/SC29/WG11 MPEG, IS11172–3, "Information Technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s – Part 3: Audio", 1993.

[5] ISO/IEC JTC1/SC29/WG11 MPEG, IS13818–7, "Information Technology – Generic Coding of Moving Pictures and Associated Audio Information – Part 7: Advanced Audio Coding", 2006.

[6] DVD Forum, "DVD Specifications for Read-Only Disc – Part 4: Audio Specifications – Version 1.2", Tokyo, Japan, March 2001.

[7] Philips and Sony, "Super Audio CD System Description", Philips Licensing, Eindhoven, The Netherlands, 2002.

[8] Microsoft, "Windows Media Technologies 4.0", available on Microsoft Developer Network at `http://msdn.microsoft.com`, April 1999.

[9] Mark F. Davis, "The AC-3 Multichannel Coder", in *Proceedings of the 95th Convention of the Audio Engineering Society, Inc.*, Reprint Publication No. S93/9951, New York, NY, October 1993.

[10] The Digital Theater System, "DTS Audio Codec Overview", available online at `http://www.dts.com`.

[11] Mark Kahrs and Karlheinz Brandenburg, "Applications of Digital Signal Processing to Audio and Acoustics – Chapter 5: Digital Audio System Architecture", Kluwer Academic Publishers, 1998.

[12] Matt Ashland, "Monkey's Audio – A Fast and Powerful Lossless Audio Compressor", available online at `http://www.monkeysaudio.com`.

[13] "WavPack – Hybrid Lossless Audio Compression", available online at `http://www.wavpack.com`.

[14] Michael A. Gerzon, Peter G. Craven, J. Robert Stuart, Malcolm J. Law and Rhonda J. Wilson, "The MLP Lossless Compression System", in *Proceedings of the AES 17th International Conference on High-Quality Audio Coding*, Florence, Italy, August 1999.

[15] Andreas Spanias, Ted Painter and Venkatraman Atti, "Audio Signal Processing and Coding – Chapter 10.3: Multichannel Surround Sound", John Wiley & Sons, Inc., 2007.

[16] Andreas Spanias, Ted Painter and Venkatraman Atti, "Audio Signal Processing and Coding – Chapter 2.8: Review of Multirate Signal Processing", John Wiley & Sons, Inc., 2007.