Using Noise Substitution for Backwards-Compatible Audio Codec Improvement*

Colin Raffel
Experimentalists Anonymous
craffel@gmail.com

April 11, 2011

Abstract

A method for representing error in perceptual audio coding as filtered noise is presented. Various techniques are compared for analyzing and re-synthesizing the noise representation. A focus is placed on improving the perceived audio quality with minimal data overhead. In particular, it is demonstrated that per-critical-band energy levels are sufficient to provide an increase in quality. Methods for including the coded error data in an audio file in a backwards-compatible manner are also discussed. The MP3 codec is treated as a case study, and an implementation of this method is presented.

1 Introduction

Since their adoption in the 1990s, perceptual audio codecs have become a vital and nearly ubiquitous way to reduce an audio file's size without dramatically affecting its perceived quality. Despite their widespread use, many of the most common codecs are criticized for their low-quality and have been superseded by formats which use improved compression schemes. One example is the highly-pervasive MP3, which is far and above the most common format for audio files, but has technology which is relatively outdated [1]. Unfortunately, most audio codecs leave little room for backwards-compatibility, and each generally requires its own specialized decoder.

This paper discusses the technique of using a perceptually-shaped representation of the coding error to improve existing audio codings. The method recommended involves coding the error as per-critical-band noise levels. Noise substitution is a relatively recent method to improve audio codings and can have a very high perceptual improvement for a very low increase in bit rate [2] [3] [4]. In order to make a true improvement to an audio codec, the

*This paper was originally published in the Proceedings of the 129th Convention of the Audio Engineering Society, San Francisco, CA, 2010. This version was edited in a few places for clarity.

sound must be widely accepted as perceptually "better" without increasing the data rate - otherwise, the codec could just be set to a higher bit rate. Methods for analyzing and re-synthesizing the coding error are compared.

2 Coding Error

Generally speaking, perceptual audio codecs throw out spectral information in an audio file by quantizing frequency domain values until it is possible to be represented at a user-defined data rate [5]. The perceptual model used is formulated to first get rid of information that is difficult for the typical human auditory system to hear. In particular, bits are typically allocated to the portions of the spectrum that humans are most sensitive to. Many codecs also make use of effects such as masking to decide what information to include. Lower bit rate files are more likely to throw away high frequency information as it is generally harder to perceive. Figure 1 shows an example of the spectrum of an uncoded audio file and a 64 kilobit per second, or 32 kilobit per second per channel, MP3 file. In this case, the codec (here, LAME[1] was used) got rid of a large amount of spectral information above 10 kHz, and in particular between 10 kHz and 13 kHz.

A very direct way to obtain the error in a coded audio is to align and subtract the coded file from the original file. Then, to recreate the original material, the error and coded file can be added. This error signal tends to be highly noisy as the portions of the audio which are left out by the coding can change in an uncorrelated way from frame to frame. This can also be deduced by observing that the sample autocorrelation of the coding error tends to be highly impulsive, as we would expect for a noisy signal [6]. A comparison of the sample autocorrelation for the coding error and the original audio file can be seen in Figure 2. This fact, combined with the notion that humans have difficulty perceiving small spectral envelope differences within a critical band [7], suggests that it may be possible to improve audio codecs by including information about the spectrum of the coding error in the coded audio file itself.

Representing the error on a per-critical-band basis is also smart from a data standpoint. For example, this scheme would require that we include 25 (one for each Bark band) values per frame, per channel. If the frame size was 1024 samples, and we coded the level in each critical band as an 8-bit number, for a 44.1 kHz sampling rate file we would add about 8.6 kbps per channel. This figure could be made smaller by using a different number representation such as a block floating point scheme, or data compression techniques such as Huffman coding [8] [5]. With this in mind, we will now focus on methods of determining and re-synthesizing the per-critical-band colored noise representation of the coding error.

[1] `http://lame.sourceforge.net/`

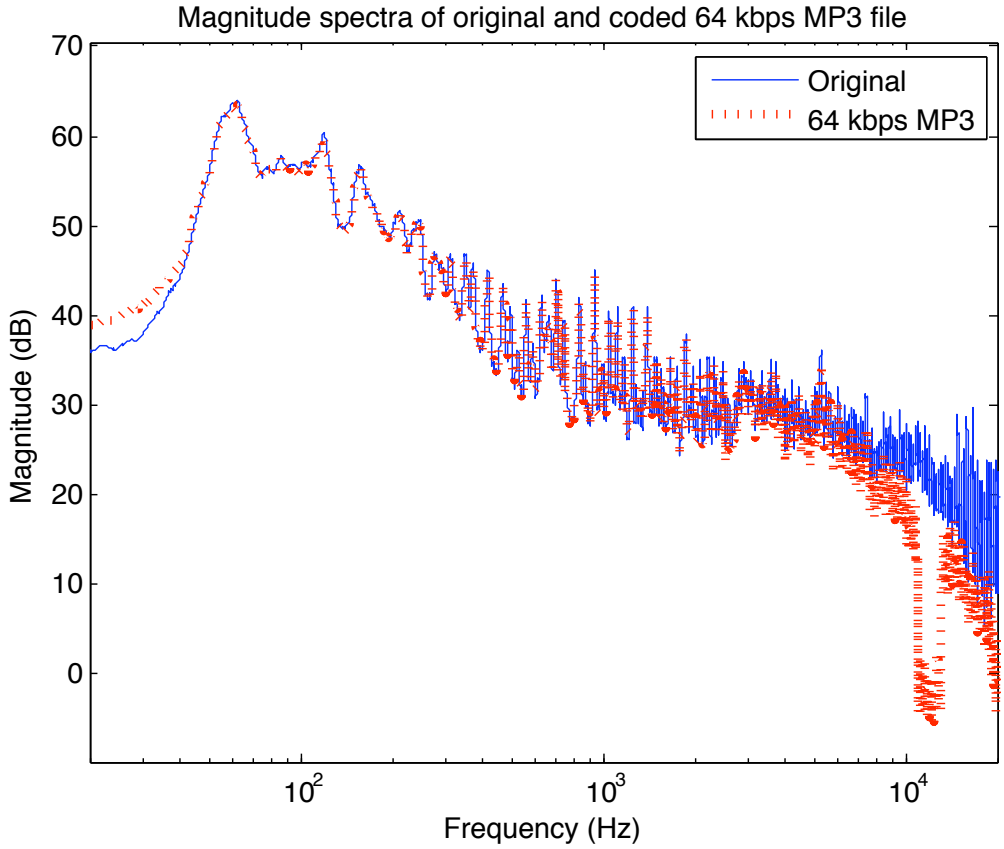

Figure 1: Demonstration of the spectral effects of the MP3 audio codec at 64 kbps.

3 Analysis

In order to obtain a perceptually accurate error representation, we need a method for determining the coloring of the noisy component of the coding error signal. Separating audio signals into sinusoidal and noise components is an effective and well-studied technique [9] [10]. Normally a peak-finding and tracking algorithm is used to extract the tonal components, and the residual is treated and modeled as colored noise. This residual is similar to the error in perceptual audio coding. The peak-finding technique can be made to be very effective for typical audio signals, but our situation is unique because the error signals being modeled have almost no stationary (that is, sinusoidal) components. For this reason, attempting to do peak finding and tracking would be mostly ineffective. Two alternate methods are proposed based on spectral flux and cepstral smoothing.

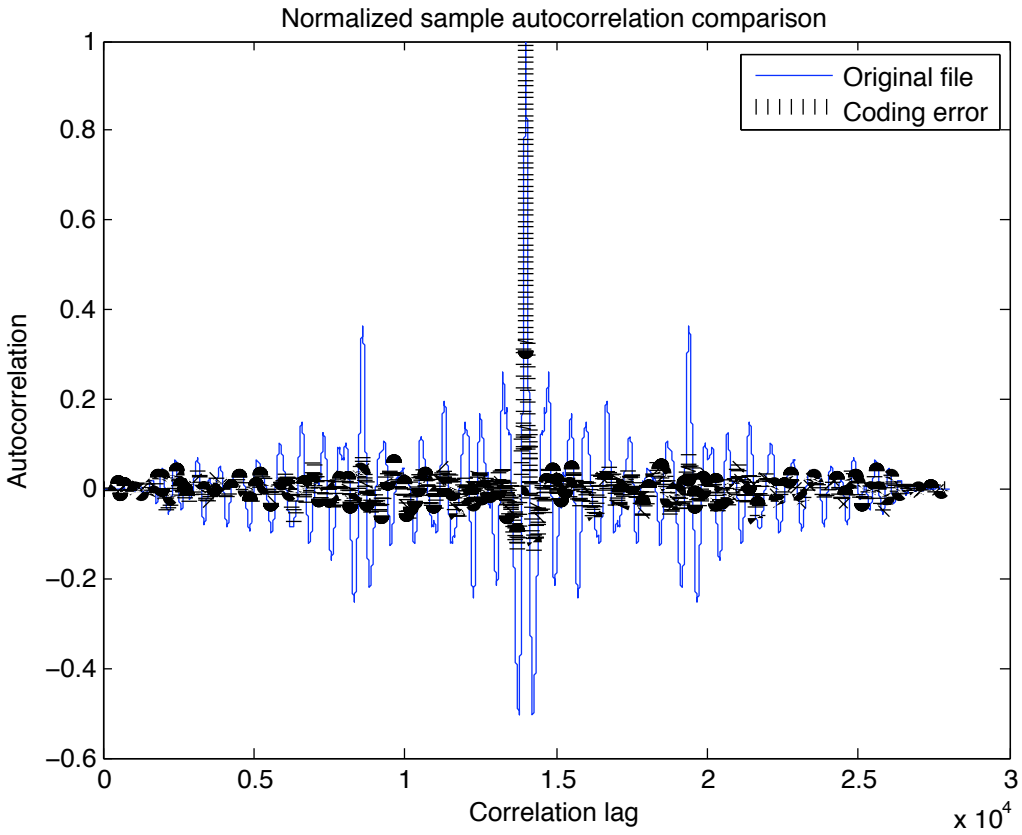

Figure 2: Demonstration that the normalized sample autocorrelation of the coding error signal tends to be significantly more impulsive than that of the original audio file.

3.1 Spectral Flux

One very simple way of finding the level of the noisy part of a spectrum is to find its spectral flux. This measure is typically used to compare the change in energy between each N-sample frame of audio, and is commonly used for onset detection [11]. The spectral flux is typically defined as the 2-norm of successive magnitude spectra, and can be calculated by

$$\mathrm{SF}(n) = \sqrt{\sum_{k=0}^{N-1} (|X[n,k]| - |X[n-1,k]|)^2} \tag{1}$$

where $X[n,k]$ is the kth frequency bin of the spectrum of the nth length-N frame of a signal [12]. Occasionally, the power spectrum is used in place of magnitude spectra, and

some implementations omit the square root [13]. Also, the half-wave rectification of the successive frame difference is sometimes used in order to measure only positive changes in energy [11]. In the context of estimating the stochastic component of a spectrum, the spectral flux is useful because it removes stationary components by subtracting bins in consecutive spectra. This ensures that sinusoids that are constant in level and remain in the same bin from frame to frame (in other words, are stationary in amplitude and frequency) will be removed.

The spectral flux is also useful for our purposes because for a Gaussian noise signal it is proportional to the signal's RMS level, which can be shown as follows: For analysis purposes, we can assume that the coding error is zero-mean Gaussian noise with variance σ^2 (in practice, it more closely resembles a two-sided exponential distribution). This implies that the first DFT bin, which is the sum of the signal across the length-N frame, will have a variance of $N\sigma^2$ because it is the sum of N independent normally distributed random variables. The rest of the bins will be complex random variables with sample variance $N\sigma^2$ because the DFT kernel is a unit-magnitude complex sinusoid. These random variables will be independent as long as a rectangular window is used. For simplicity, we will define the spectral flux for a frame n as

$$\mathrm{SF}(n) = \sqrt{\sum_{k=0}^{N-1} (\Re(X[n,k]) - \Re(X[n-1,k]))^2} \tag{2}$$

For the complex frequency domain random variables, the real part has half of the variance of the complex DFT bin value itself [14]. The subtraction in the spectral flux calculation will have twice the variance of a single DFT bin because it is the linear combination of two Gaussian random variables, which are uncorrelated as long as there is no overlap between frames. In other words, we have

$$\begin{aligned}
&\mathrm{Var}(\Re(X[n,k]) - \Re(X[n-1,k])) && (3)\\
&= 2\mathrm{Var}(\Re(X[n,k])) && (4)\\
&= 2\left(\frac{N\sigma^2}{2}\right) && (5)\\
&= N\sigma^2 && (6)
\end{aligned}$$

Now, the RMS of a signal is defined as

$$\mathrm{RMS}(y[k]) = \sqrt{\frac{\sum_{k=0}^{N-1} y[k]^2}{N}} \tag{7}$$

So if we define

$$y[k] = \Re(X[n,k]) - \Re(X[n-1,k]) \tag{8}$$

then it is clear that the spectral flux is calculating a $\sqrt{N}$-scaled RMS of N Gaussian random variables with variance $N\sigma^2$. The RMS of these random variables will then be $\sqrt{N}\sigma$. This implies that

$$\begin{aligned} &\mathrm{SF}(n) && (9)\\ &= \sqrt{N} RMS[\Re(X[n,k]) - \Re(X[n-1,k])] && (10)\\ &= \sqrt{N}(\sqrt{N}\sigma) && (11)\\ &= N\mathrm{RMS}(x[n]) && (12) \end{aligned}$$

In summary, we have shown that for Gaussian noise, the spectral flux definition given in Equation (2) calculated over non-overlapping rectangular-windowed frames is proportional to the RMS of the signal.

Because the coding error is not specifically Gaussian-distributed noise, and because there is some correlation from frame to frame, this proportionality is not strictly true in practice. Furthermore, we have found that the half-wave-rectifying spectral flux definition used in [11] achieves more accurate results in comparison with the definition used in Equation (2). However, we have found experimentally that a proportionality holds for all spectral flux definitions discussed herein, so this relation serves as a general guideline that makes this technique useful. A graph showing the RMS and spectral flux of the coding error using a frame size of 1024 samples is shown in Figure 3.

To generate critical band levels based on the spectral flux approach, we simply summed the consecutive spectral difference across only those bins which fell in each band. In other words, the spectral flux for frame n and critical band B is given by

$$\mathrm{SF}(B,n) = \sqrt{\sum_{k \in B} (|X[n,k]| - |X[n-1,k]|)^2} \quad (13)$$

where $k \in B$ denotes the relation that the frequency corresponding to the kth bin is within critical band B. This method proved to be fairly robust in representing the error with based solely on critical band levels. When synthesizing colored noise based on these levels, we found that the results were perceptually similar but tended to change too rapidly, resulting in a "fluttering" sound. This is likely due to the fact that the spectral flux caused the estimate to be intentionally and overly uncorrelated. To combat this effect, we implemented a leaky integrator scheme which prevented the level estimates from changing too rapidly. This helped with the fluttering character to a limited degree. A comparison between the spectrum of noise synthesized with this technique and the actual error spectrum is shown in Figure 4.

3.2 Smoothed Cepstrum

To further explore methods of generating perceptually equivalent representations of the coding error, we focused on techniques which attempt to find the spectral envelope directly.

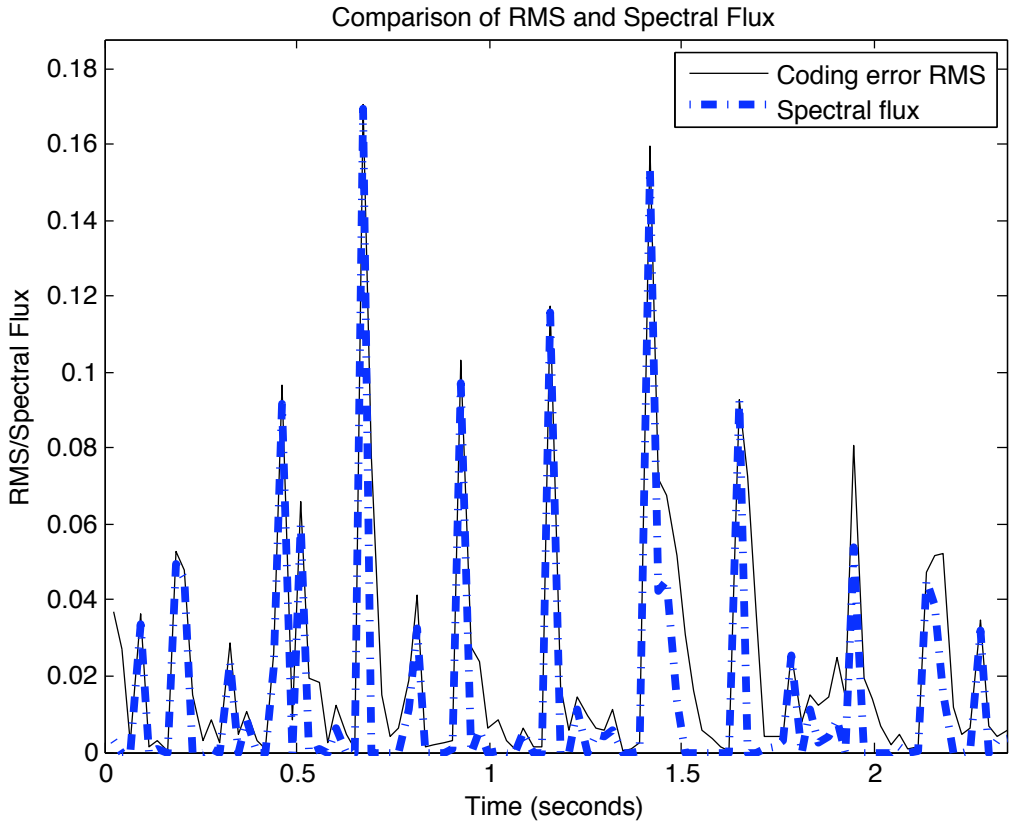

Figure 3: RMS and spectral flux levels for the coding error of a 64 kilobit per second MP3 file.

One such method is cepstral smoothing. The real cepstrum is defined as the inverse DFT of the log of a signal's spectrum [15]. It can be calculated by

$$C[n] = \frac{1}{N} \sum_{k=0}^{N-1} \log(|X(k)|) e^{j2\pi nk/N} \tag{14}$$

where $X(k)$ is the kth bin of the length-N DFT of a signal and $C[n]$ denotes the nth sample of the cepstrum. To obtain a spectral envelope, we can window the real cepstrum in the time domain and take its Fourier transform [6], which results in a smoothing of the original signal's spectrum. We used a Hamming window of length ≈7 ms. This method is very effective for determining the envelope of a relatively peak-free spectrum like those of the coding error. One frame of the coding error and its smoothed cepstrum-generated

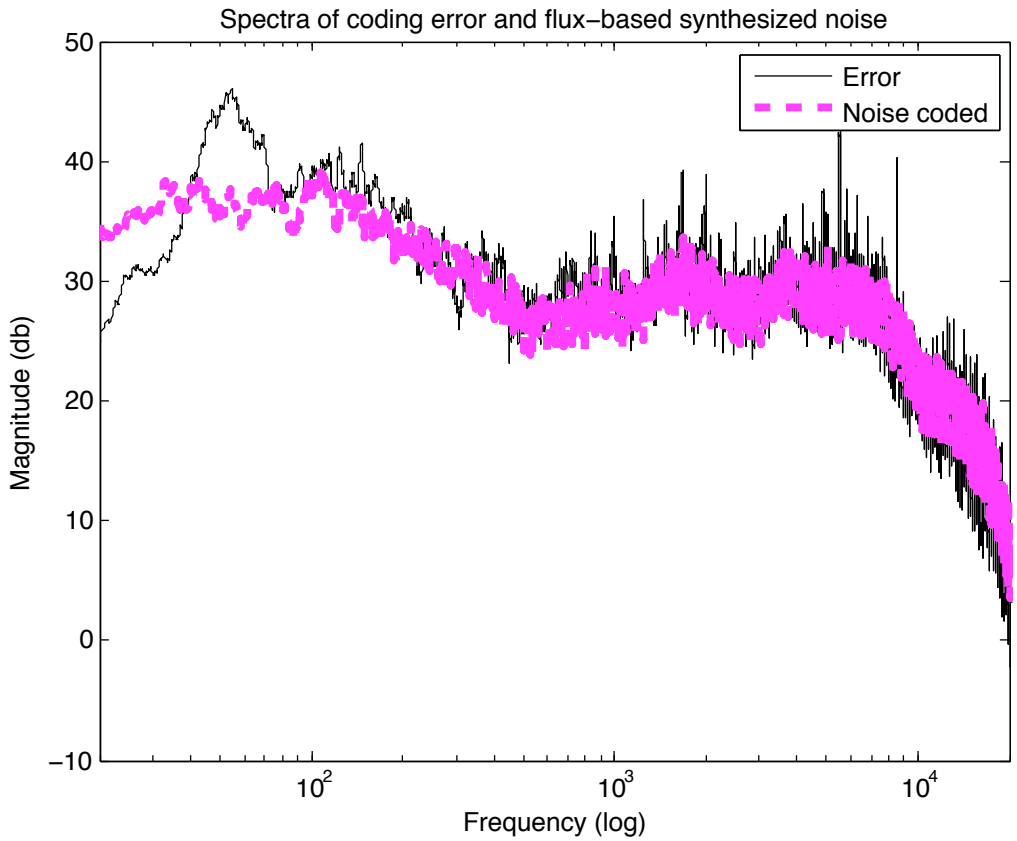

Figure 4: Results of representing the coding error with a spectral flux-based per-critical-band noise estimate.

envelope is shown in Figure 5.

With this envelope, we can generate the per-critical-band noise level by simply finding the smoothed cepstrum's mean in each band. This provides an accurate metric that does not vary quickly as with the spectral flux-based technique, and was generally smoother in each frame. It is worth noting that finding the mean of the cepstrum-based spectral envelope achieves somewhat similar results to finding the mean of the magnitude spectrum itself. The cepstral smoothing method is also significantly more computationally complex. However, we found that cepstral smoothing resulted in a significantly more accurate spectral envelope estimate, which produced more perceptually accurate error representations.

The resulting critical band estimates for the spectral flux, smoothed cepstrum, and per-band spectral mean methods is shown in Figure 6 (using the same spectral frame shown

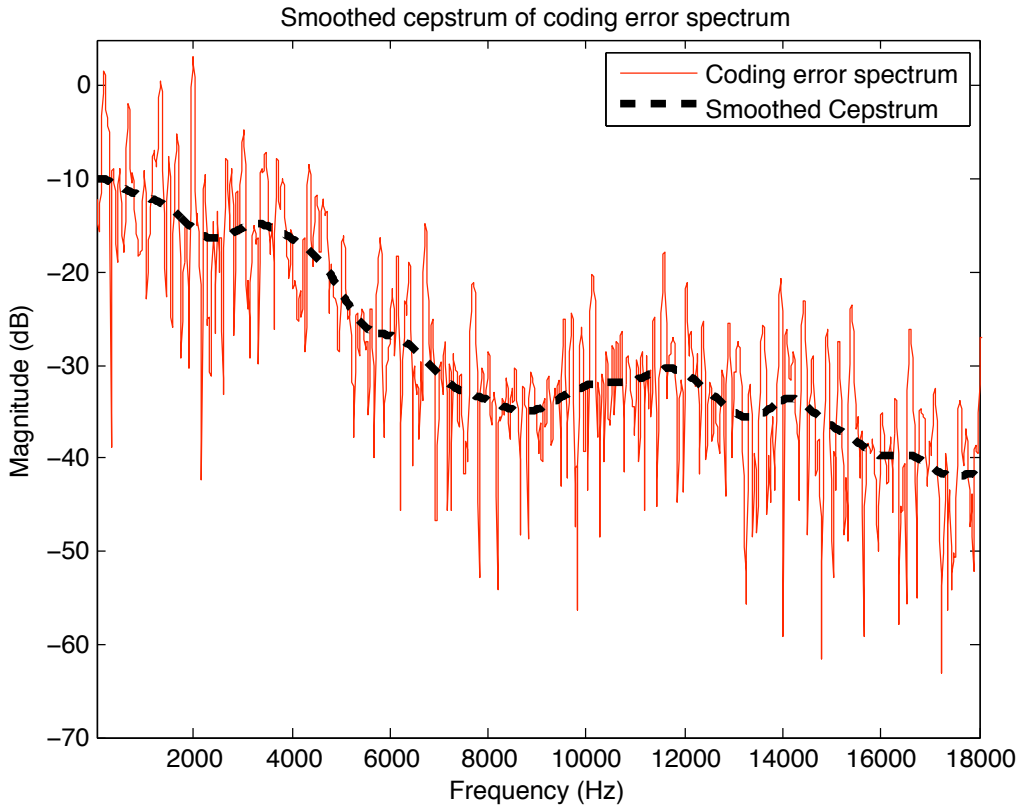

Figure 5: Example of spectral envelope derived from cepstral smoothing of the coding error signal.

in Figure 5). Clearly, the spectral flux does not produce as accurate of a representation due to the fact that it omits correlated components, which do arise in the coding error. As mentioned, however, it is convenient due to its relation to the RMS value of the signal and its relative ease of computation. Additionally, the spectral flux method causes the colored noise to solely model the "noisy" part of the error signal. However, we have found that the smoothed cepstrum method generally produces an envelope which sounds more perceptually accurate and does not change dramatically from frame to frame.

4 Synthesis

The most straightforward way to synthesize a colored noise signal from the calculated critical band weightings is to generate a random spectrum (that is, a frame-length of complex

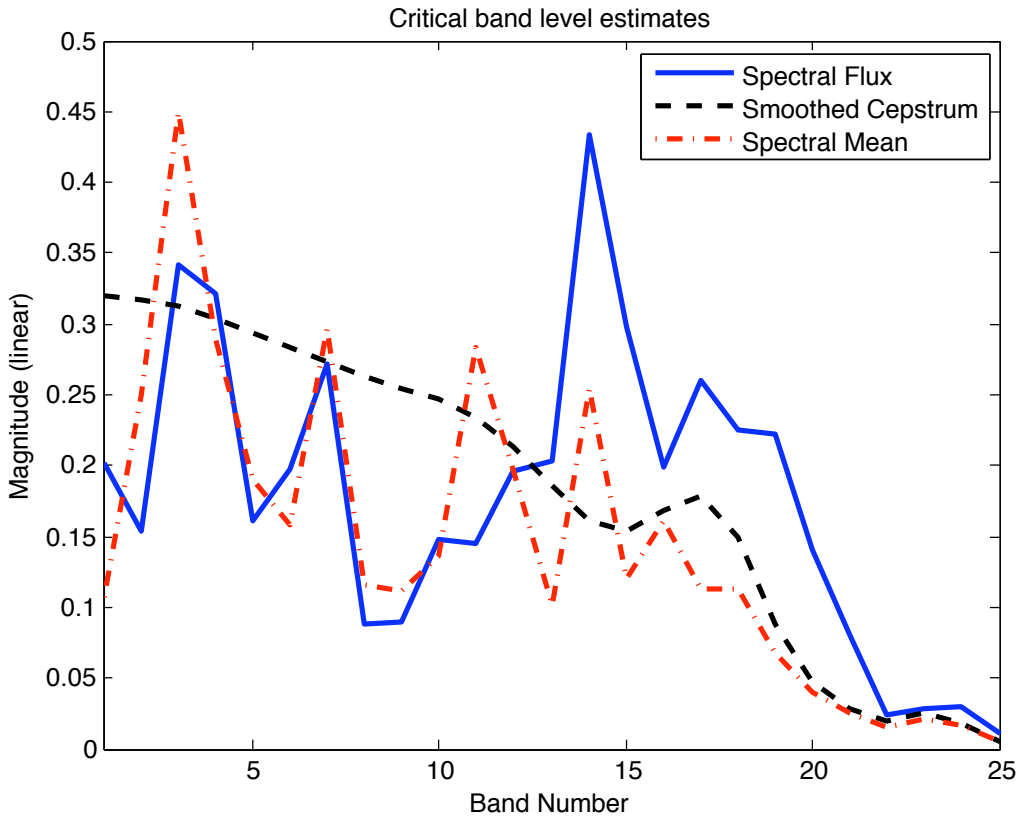

Figure 6: Comparison of methods used for calculating the critical band levels for the coding error signal.

numbers) and scale each bin magnitude according to the level in the corresponding band. One difficulty with this technique which came up immediately was that the level discontinuities between each band created perceptually inaccurate colorings. This is easily fixed with interpolation on a bin-by-bin basis over the band levels. In Figure 6, linear interpolation is used to generate a smoother spectral weighting. Another source of discontinuity came from the frame-by-frame difference in noise coloration. One nice characteristic of generating noise is that once the spectral weighting is found, a colored noise sequence of arbitrary length can be created. So, rather than generating a frame's worth of noise, each spectral weighting is used to generate two frames of random complex numbers. Half of each frame is then crossfaded with its neighboring frames with an overlap-add window technique. This method makes the transition between frames considerably less abrupt and noticeable. In our implementation, we found that a sinusoidal window [6] sounded best in

terms of reducing flutter between frames.

4.1 Transients

One great difficulty with our noise detection techniques comes from the way they treat transients. Because both an impulse and white noise will result in a flat spectrum, it is easy for our spectral envelope estimations to confuse a transient for a large amount of noise. Furthermore, it is common for a large amount of a transient to be left out of an audio coding, so impulsive signals in our source material were often also partially found in the coding error. In the worst case, this would cause our system to generate a frame's worth of white noise in response to an impulse. Based on early listening tests, this was the aspect of our system which bothered subjects most.

Some methods of modeling signals with sinusoids and noise also involve the modeling of transients [10]. Typically, impulsive sounds are not modeled in any particular way, and the sines and noise are simply left out while the unmodified transient is played back. To mimic this approach, we tried a number of transient detection schemes and used them to determine when to not synthesize any noise. This technique was somewhat effective, but we had difficulty accurately and consistently finding and characterizing transients. Furthermore, the best technique for treating an impulsive signal was not generally consistent. For example, in some instances it would sound better to fade out the noise momentarily, while in others it was smarter to simply generate an extra frame of the previous colored noise weighting.

The best method for treating transients was found based on the observation that the coding error generally followed a similar amplitude envelope to the coded audio. In other words, rather than simply using the coded error levels to determine the noise's amplitude on a per-frame basis, we can scale the synthesized noise per-sample by the coded audio's amplitude envelope. Here, we simply define a signal $x[n]$'s level over a frame of size N as

$$\text{Level}(x[n]) = \frac{1}{N} \sum_{n=0}^{N-1} |x[n]| \tag{15}$$

or, in other words, the average of the signal's absolute value over the frame. The combination of the coded audio's envelope and the overall noise level in each frame allowed us to better match the instantaneous error level without encoding any additional information. More importantly, this technique helps silence the noise representation near transients so that the problematic noise-during-transient frames were less apparent.

The main drawback to this approach is that it tended to perceptually over-emphasize the time-domain envelope, but this can be avoided by creating a weighting for the per-frame noise amplitude level and the calculated coded audio envelope. This can be expressed as

$$y[n] = (1 - \alpha + \alpha L[n])x[n] \tag{16}$$

where $x[n]$ is the synthesized noise signal, $L[n]$ is the coded audio file envelope, α is the mix amount, and $y[n]$ is the resulting modulated residual representation. We achieved generally better results near transients with $\alpha \approx .2$. A comparison of the desired coding error envelope, coded noise envelope, and coded audio envelope-modulated ("matched") noise with $\alpha = .2$ is shown in Figure 7.

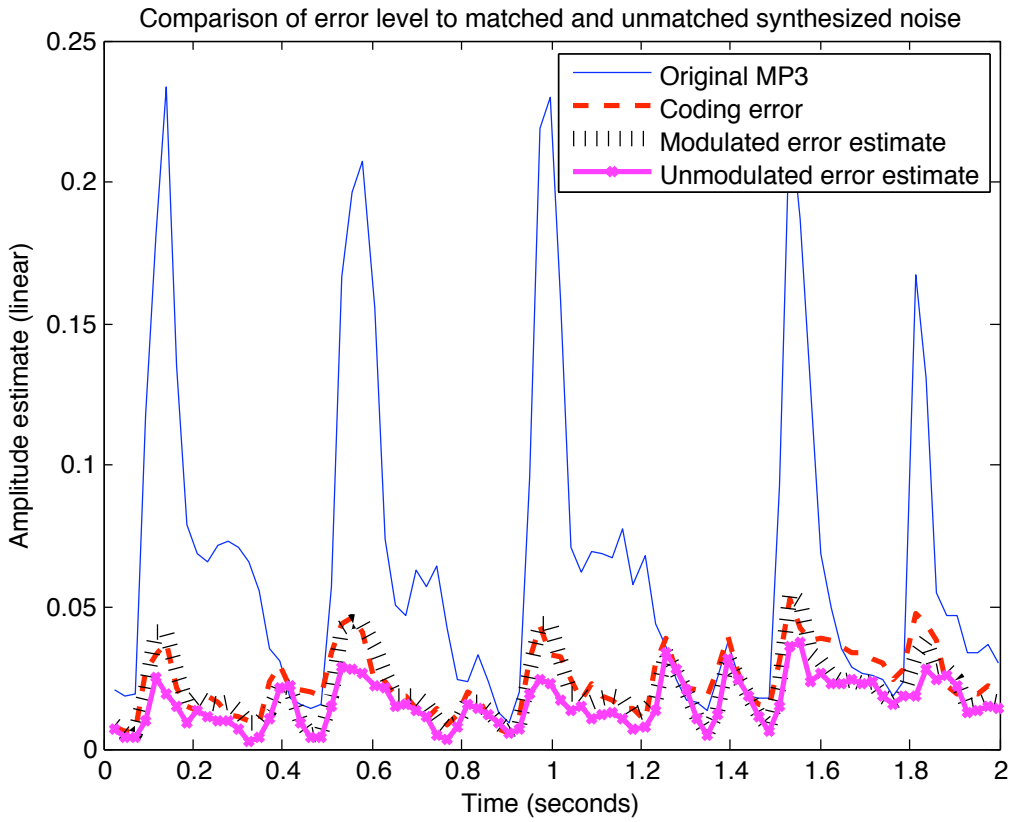

Figure 7: Demonstration of the accuracy improvement possible by modulating the error estimate with the coded audio's envelope.

5 Implementation

To test these approaches, we implemented an audio codec called "row-mp3" which uses the ID3 tags in an MP3 file to store per-frame noise level information [16]. ID3 is a metadata

container format implemented in the vast majority of MP3 players.[2] The tags are generally used to give text descriptors of the content such as the artist or song title. Fortunately, if an MP3 player is not able to parse an ID3 tag, it simply ignores it. In this way, players which are not row-mp3 enabled would ignore the information, making it backwards compatible with the common MP3. Most audio codecs have similar support for arbitrary metadata.

We created row-mp3 files based on the spectral flux level estimate for a variety of musical genres and audio files. We found that the approximately 60 test subjects tended to rate the row-mp3 files about 150% better than the mp3 file of the corresponding bit rate for low-quality mp3 codings.[3] For higher quality codings, there was no real statistical difference. The frame-by-frame critical band levels were compressed using Huffman Coding, which allowed us to keep the data rate increase very low relative to the MP3 file size. These results suggest that the noise substitution technique discussed herein has promising applications in improving low-quality audio codings in a backwards compatible manner without dramatically increasing the data rate.

6 Conclusion

We have shown that the error in perceptual audio codings can be effectively and cheaply modeled by colored noise. Some techniques for measuring the per-critical-band noise levels were discussed and difficulties with each method were addressed. Specifically, we showed that the spectral flux provides a theoretically-sound estimate but that a smoothed cepstrum technique works better in practice. The generation of discontinuity-free, transient-safe and amplitude-matched colored noise based on these levels was also described. In particular, we took advantage of the unique aspects of the coded audio and coding error to generate a more perceptually accurate noise coding. Early tests show that these techniques can be used to improve the perceived quality of audio codecs.

Because our system simply defines a framework for representing coding error as critical band levels, it will be easy to improve upon our analysis and synthesis processes in a backwards-compatible manner. For example, if thousands of files are created with an old spectral envelope analysis method, they will still work (albeit relatively poorly) when a new analysis technique is used, as long as the data format doesn't change. This also allows for different implementations of this system to be created, which can use differing techniques, allowing the end-user to pick their favorite analysis and synthesis schemes. In this way, the codec improvement method discussed herein blurs the distinction between a codec and an audio enhancement, in that it can be interpreted as an attempt to make poor-quality audio sound "better".

[2] `http://www.id3.org/`

[3] The test our subjects took can be found at `https://ccrma.stanford.edu/~craffel/etc/mp3challenge`

7 Acknowledgements

The author would like to thank Isaac Wang and Jieun Oh for their collaboration on implementing the row-mp3 codec, Prof. Marina Bosi for her instruction in the field of audio coding, and Prof. Julius Smith for helpful discussions on topics in this paper.

References

[1] John Borland, "MP3 losing steam?," *CNET News*, Oct. 2004, `http://news.cnet.com/MP3-losing-steam/ 2100-1027_3-5409604.html`.

[2] Donald Schulz, "Improving audio codecs by noise substitution," *J. Audio Eng. Soc*, vol. 44, no. 7/8, pp. 593–598, 1996.

[3] Jürgen Herre and Donald Schulz, "Extending the MPEG-4 AAC codec by perceptual noise substitution," in *Audio Engineering Society Convention 104*, May 1998, pp. 4720–4734.

[4] Tony S. Verma and Teresa H. Y. Meng, "A 6kbps to 85kbps scalable audio coder," in *Proceedings of the 2000 IEEE International Conference On Acoustics, Speech, and Signal Processing*, Washington, DC, USA, 2000, pp. 877–880.

[5] Marina Bosi and Richard E. Goldberg, *Introduction to Digital Audio Coding and Standards*, Kluwer Academic Publishers, Norwell, MA, USA, 2002.

[6] Julius O. Smith, *Spectral Audio Signal Processing, October 2008 Draft*, `http://ccrma.stanford.edu/~jos/sasp/`, accessed September 1, 2010, online book.

[7] Eberhard Zwicker and Hugo Fastl, *Psychoacoustics: Facts and Models*, Springer, 2nd updated edition, April 1999.

[8] David A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, January 2007.

[9] Xavier Serra, *A System for Sound Analysis/Transformation/Synthesis based on a Deterministic plus Stochastic Decomposition*, Ph.D. thesis, Stanford University, 1989.

[10] Scott Levine, *Audio Representations for Data Compression and Compressed Domain Processing*, Ph.D. thesis, Stanford University, 1998.

[11] Simon Dixon, "Onset detection revisited," in *Proc. of the Int. Conf. on Digital Audio Effects (DAFx-06)*, Montreal, Quebec, Canada, Sept. 18–20, 2006, pp. 133–137.

[12] Eric Scheirer and Malcolm Slaney, "Construction and evaluation of a robust multi-feature speech/music discriminator," in *Proceedings of the 1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Washington, DC, USA, 1997, pp. 1331–1334.

[13] Tao Li, "Musical genre classification of audio signals," in *IEEE Transactions on Speech and Audio Processing*, 2002, pp. 293–302.

[14] Fabien Milloz and Nadine Martin, "Estimation of a white Gaussian noise in the short time fourier transform based on the spectral kurtosis of the minimal statistics: application to underwater noise," in *Proceedings of the 2010 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Dallas, TX, USA, 2010.

[15] John R. Deller Jr., John G. Proakis, and John H. Hansen, *Discrete Time Processing of Speech Signals*, Prentice Hall, Upper Saddle River, NJ, USA, 1993.

[16] Colin Raffel, Jieun Oh, and Isaac Wang, "Row.mp3 encoder," `https://ccrma.stanford.edu/~craffel/ software/rowmp3/rowmp3.pdf`, 2010.