Overview of lossless audio codecs

Luka Železnik[1], Damjan Strnad[1], Bogdan Lipuš[1], Josef Kohout[2], David Podgorelec[1]

[1] University of Maribor, Faculty of Electrical Engineering and Computer Science
[2] University of West Bohemia, Faculty of Applied Sciences
E-mail: luka.zeleznik@student.um.si

Overview of lossless audio codecs

In this paper, we overview existing lossless audio formats, their way of predicting audio samples and encoding differences between the samples and predicted values. The considered algorithms are classical FLAC, MPEG-4 ALS, and neural net enhanced Monkey's Audio and LINNE. Then we test and compare the compactness, encoding, and decoding times of mentioned codecs using various types of audio.

1 Introduction

Data compression is currently changing with advancements in artificial intelligence. Previous mesh-based standard procedures are being updated with novel approaches that push data compactness further than ever. Lossless audio is the leading type of compression of this change in approach, for its simple and intuitive way of translating the problem to a neural network (NN).

Lossless audio compression is also becoming more popular with the wide adoption of high-speed audio streaming over the internet. That enables larger but higher quality, lossless music and other forms of audio to reach more and more listeners. With increasing awareness of potencial data loss, the demand for data backup services has increased, for which compact lossless compression is required.

In this overview, we review traditional and novel lossless audio compression methods and compare their compactness, encoding, and decoding times.

2 FLAC

Free Lossless Audio Codec (FLAC) [1] started as a project by Josh Coalson in 1999 as a free and completely open alternative to the lossless audio codecs of the day. Main ideas were taken from the now deprecated SHORTEN [2] algorithm but were expanded and made more robust for different audio consumption ways, mainly streaming. FLAC is now considered the standard for lossless audio compression.

In 2011 Xiph.Org Foundation[1] took stewardship over the open-source project, and now the source code is available on GitHub[2].

But despite the takeover, the goals of the FLAC codec remain the same. None of the used algorithms can have patents, FLAC must stay completely lossless (lossy compression shouldn't even be an option, even encoding floating point samples should be excluded), and the format should be error resistant, streamable, with flexible metadata, and should support fast sample-accurate seeking. But most importantly decoding must be done in real time even on modest hardware.

2.1 Sample prediction

In all modern audio codecs a mechanism called blocks is used so that decoding can be done in real time even for long audio streams. The default block size for audio with a sampling rate of 44.1 kHz is 4608 samples. Blocks are comprised of as many subblocks as there are channels. So, a typical block of stereo audio has 2*4608 = 9216 samples. Each block has an 8-bit cyclic redundancy check (CRC) code for redundancy.

After the audio is partitioned into blocks, it is decorrelated by default, which can be specified as a codec parameter. Left and right channels of the stereo audio stream are typically similar, so FLAC exploits that inter-channel correlation. New channels, called middle and side, are calculated so that *middle = (left + right)/2* and *side = left – right*. Effectively we are encoding the average in the middle and details in the side channel, as seen in Haar wavelet transform [3]. When decoding inverse transformation is performed.

As most modern lossless codecs, FLAC uses predictive approximation functions $f(t)$ of the sample $s(t)$ and encodes the residual $e(t)$ as a part of the data stream, as seen in equation (1).

$$e(t) = s(t) - f(t) \qquad (1)$$

FLAC employs four such functions, though only the last two provide actual compression [4]:

- **Verbatim**: The prediction is a zero audio sample. This means that the residual is the actual sample. This prediction doesn't use variable length codes for compression.
- **Constant**: This encodes digital silence (a stream of constant samples). Such subblocks are compressed using run-length encoding.
- **Fixed linear predictor**: A simple polynomial predictor of orders 0, 1, 2, 3, or 4. Side information of 3 bits is included to determine which order is used. The verbatim prediction is effectively a zeroth-order polynomial prediction, but here the residual is compressed. We can see five possible predictors $\hat{s}$ below in equation (2), where $s(t-x)$ are previous samples in the audio stream:

  0. $\hat{s}_0(t) = 0,$
  1. $\hat{s}_1(t) = s(t-1),$
  2. $\hat{s}_2(t) = 2s(t-1) - s(t-2),$ (2)

[1] https://xiph.org/

[2] https://github.com/xiph/flac

3. $\hat{s}_3(t) = 3s(t-1) - 3s(t-2) + s(t-3)$,
4. $\hat{s}_4(t) = 4s(t-1) - 6s(t-2) + 4s(t-3) - s(t-4)$.

- **FIR Linear prediction**: A more sophisticated method that is also referred to as general linear predictive coding (LPC) [5] (see section 2.2).

Residuals are encoded using Rice codes [8]. These descendants of Golomb codes [9] are variable-length codes (VLC) for encoding integers. Rice codes are very fast to encode and decode, and optimal for the Laplace distribution [10], which is expected for residuals from this type of prediction.

2.2 Linear predictive coding

Linear predictive coding [5] is the primary prediction method used for FLAC. It is somewhat slower than a fixed linear predictor but usually results in 5-10% smaller files, depending on the concrete audio character. Maximum polynomial order can be set as a parameter within the range of 1 and 32 – larger number grants better predictions but slows the encoder. Generally, increasing the maximum order over 9 brings diminishing returns [4].

The basic approach to finding the best polynomial fit is finding a set of predictor coefficients to minimize the mean-squared prediction error over a short waveform segment [5].

To find an optimal coefficient for the polynomial fit of sampled audio $(\alpha_1 \dots \alpha_p)$, a system of p equations (3) must be solved, where ϕ are Pearson autocorrelation coefficients [6] of the system. The reference encoder uses Levinson-Durbin recursion [7] to solve this system, but other methods can also be used.

$$\begin{bmatrix} \phi[0] & \phi[1] & \phi[2] & \dots & \phi[p-1] \\ \phi[1] & \phi[0] & \phi[1] & \dots & \phi[p-2] \\ \phi[2] & \phi[1] & \phi[0] & \dots & \phi[p-3] \\ \dots & \dots & \dots & \ddots & \dots \\ \phi[p-1] & \phi[p-2] & \phi[p-3] & \dots & \phi[0] \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \\ \alpha_p \end{bmatrix} = \begin{bmatrix} \phi[1] \\ \phi[2] \\ \phi[3] \\ \vdots \\ \phi[p] \end{bmatrix} \quad (3)$$

The Levinson–Durbin algorithm determines by recursion the optimum ith-order predictor from the optimum $(i-1)$th-order predictor, so, as part of the process, all optimal order predictors, from zeroth to p-th order, are calculated.

3 MPEG-4 Audio Lossless Coding (ALS)

MPEG-4 ALS [11] is a lossless audio standard adopted by the MPEG committee first and later by ISO. Because of this, its formal designation is ISO/IEC 14496-3:2005/Amd 2:2006 international standard [4].

As a competitor of FLAC, MPEG-4 ALS also uses inter-channel decorrelation (see five static linear predictors in equation (1) and linear prediction codes in section 2.2), even though it is expanded up to 2^{16} channels. The innovations are a novel long-term predictor, random access frames, a different option for encoding residuals, and support of IEEE 32-bit floating point [12] samples.

Most audio signals have a basic expected frequency, called the fundamental frequency. In music, it coincides with the musical scale the song is written in and depends on the instruments, and in speech, on the pronunciation of phonemes. So, over a short audio excerpt, it can be predicted that some sine waves will be more prevalent than others. We can also expect multiples of the fundamental frequency, called harmonics. They contribute to the richness of the audio to our ears [13].

A long-term predictor (LTP) uses this correlation to correct a short-term LPC (section 2.2) prediction, as seen in equation (4). The $\hat{s}(t)$ is a short-term prediction, γ are five different quantized gain values, and r is a lag. It is calculated based on the fundamental frequency and sampling rate. For example, musical note C has a frequency of 261 Hz. So an audio recording sampled at 48 kHz can expect some correlation (lag) at 48,000/261 = 184 audio samples [4].

$$\varepsilon(t) = \hat{s}(t) - \sum_{j=-2}^{2} \gamma_{r+j} \cdot \hat{s}(t - r + j) \qquad (4)$$

Random access frames are another quality-of-life improvement. A practical audio codec should allow users to skip forward or backwards in an audio stream at their leisure. Usually, an audio frame (a user-controlled number of samples) depends on previous frames in a sequence. This approach, called progressive predictions [4], is used so predictions can improve with more data about the audio stream. The random access frames can be decompressed without any context and effectively restart the progressive prediction.

Residuals are then encoded with one of two methods. One option is to use Golomb-Rice codes. The first step to creating this VLC is transforming residuals into non-negative integers with equation (5). Then the residuals are encoded as in FLAC, with an exception that GolombRice codes only support positive integers per (5).

$$r' = \begin{cases} 2r & if\ r \geq 0 \\ -2r - 1 & if\ r < 0 \end{cases} \qquad (5)$$

The other option is a hybrid method using block Gilbert-Moore (BGM) [14], Golomb-Rice, and fixed length codes. It achieves a better compression rate on account of better efficiency for encoding small residuals, but is about 15-25% slower. A central region between $[-x, x]$ is encoded using BGM and the tail of the distribution is using Golomb-Rice codes.

4 Monkey's Audio

Monkey's Audio [15] is a well known NN-based algorithm and file format for lossless audio data compression. Unlike FLAC, Monkey's audio is not open source. It is freeware, and its software development kit (SDK) can be used for applications free of charge, but the distribution of modified versions is not allowed. This codec typically performs slightly better than FLAC (see the experiments in Section 6), but the cost is a dramatic increase in hardware requirements, especially on the decoding end.

The encoder is comprised of three steps. Firstly, the same process of inter-channel decorrelation is performed, as described in 2.1. The next step is the prediction. Monkey's audio uses a fixed-first-order predictor followed by multiple adaptive offset filters. Then the results are filtered up to 3 times using convolutional NN filters, dependent on the compression level, of which five exist. A high compression rate is achieved via sample-by-sample stochastic gradient descent [16]. Lastly, residuals are compressed using a range-style adaptive arithmetic encoder [4, 17].

5 LINNE

Linear-predictive NN encoder (LINNE) [16] is a novel and relatively unknown approach to audio compression that uses NN processing as a main predictor for the algorithm.

Firstly, the channels are decorrelated per section 2.1. Then multiple first-order LPC filters are applied to simplify and boost the NN's performance.

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_L \end{bmatrix} \rightarrow \begin{bmatrix} 0 & \cdots & 0 \\ x_1 & \ddots & \vdots \\ \vdots & \ddots & 0 \\ x_{L-p} & \vdots & x_1 \\ \vdots & \ddots & \vdots \\ x_{L-1} & \cdots & x_{L-p} \end{bmatrix} h \qquad (6)$$

This codec uses a convolution neural network (CNN) [18], which inputs a 2D matrix, not a 1D vector that can easily be interpreted as a block of audio. Equation (6) shows the transformation of an L-sized block with order p, where h is a division coefficient vector for that layer.

Then the popular 34 layer CNN ResNet [19] is used for prediction. After adjusting the weights they are quantized to an 8-bit signed integer, which results in negligible predictive performance degradation. Without this step, the coefficients represent a significant amount of data at around 0.73% of the data stream.

Finally, the residuals are encoded using recursive Golomb-Rice codes [4], more suited for heavily tailed distribution than original Golomb-Rice codes.

6 Comparison

The codecs were tested with 61 previously uncompressed 16-bit PCM stereo audio files from different genres, including electronic, folk, pop, rock, metal, and film soundtracks. We also tested two excerpts from audiobooks and audio from two films, which were previously compressed with lossy methods. The respective versions of the codecs and used parameter flags were:

- FLAC 1.3.4 (--best),
- MPEG-4 Audio Lossless Coding (ALS), Reference Model Codec Version 23 (-b),
- Monkey's Audio 10.16 (-c4000),
- LINNE 0.0.2 (-m 0).

The results were collected on a computer with an Intel Core i5-8300H processor and 8 GB of memory.

Figure 1 shows the encoding times of all 65 audio samples depending on their length. We can see that they are very predictably following a linear trend, with LINNE being the most demanding. Note that both axes are on a logarithmic scale for clarity. On the contrary Figure 2 shows decoding times, and we encounter no such predictability. FLAC is the fastest codec in both figures.

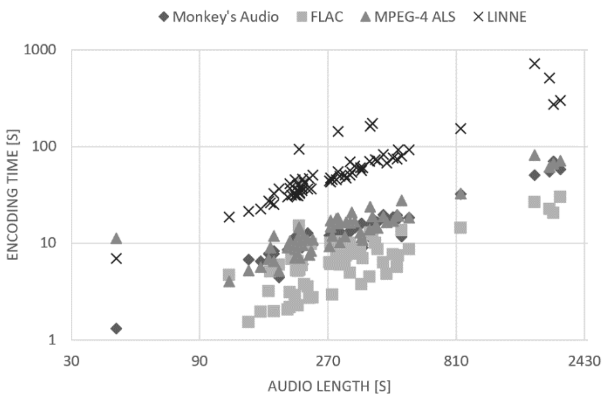

Figure 1: Encoding times depending on audio length

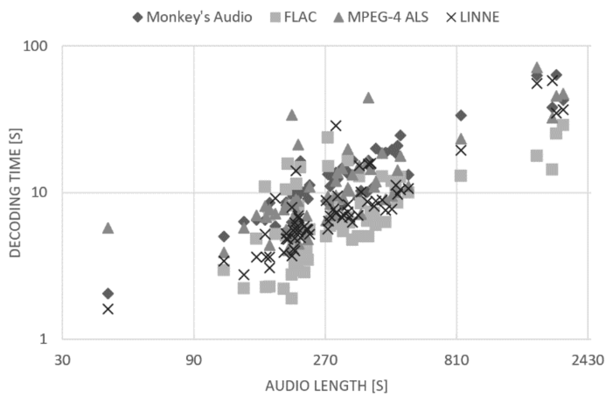

Figure 2: Decoding times depending on audio length

Table 1 shows the tested samples' average compression ratios, compression, and decompression times. Compression ratios are calculated with $CR = \frac{Uncompressed\ size}{Compressed\ size}$, so a higher ratio indicates better compression.

Table 1: Average results

	Compression ratio	Encode time (s)	Decode time (s)
FLAC	1.794	7.94	7.99
MPEG-4 ALS	1.830	16.44	13.01
Monkey's Audio	1.865	14.30	14.20
LINNE	1.815	79.27	9.98

Generally, we can expect the best compression from Monkey's Audio. The exceptions are audiobooks and podcasts, because Monkey's Audio prediction methods are not well suited for encoding speech. This compactness though comes at a cost of encoding and decoding times. Even with this caveat, we can deduce that Monkey's Audio is still the most appropriate codec for archiving large amounts of audio [16].

MPEG4-ALS decoding and encoding times were also relatively slow (Figure 1, Figure 2), with above-average compression ratios. Notably, here we used its improved residual coding, which should slow down the encoder by 15-25% but achieve better compression [11]. We observed no such difference in performance but a worse compression ratio using Rice codes only.

The newcomer prototype LINNE encoder boasts a fast decoding time, at least compared to other NN alternatives, but very slow encoding (Figure 1). Because of that, we used the fastest encoding option because the codec is by far the most resource intensive. The highest encoding mode (flag -m 7) increases (Table 1) compression times tenfold, decoding times by 50%, and improves compression ratios by less than one per cent. But those are still worse than Monkey's Audio, but now with even more untenable compression times.

Lastly, FLAC compressed files significantly the worst. It justifies its wide adoption by end users with solid encoding and decoding times, as is the goal of its creators. But for archiving, better options exist.

7 Conclusion

Neural net enhanced codec Monkey's Audio reigns supreme with excellent encoding but long decoding times. LINNE tries to bridge the gap in terms of decompression times, but loses out in terms of compactness.

For the average user, classical encoders, which use linear predictive coding, are still popular because of their wide adoption, hardware, and software support. Especially FLAC's simple design and openness contribute to its success. But codecs based on new advances in artificial intelligence have a future in environments with more computing power. On a large scale, we can expect a sizable gain in efficiency storing lossless audio data, when encoding times are not as important. Decoding times are typically more significant, but their value is diminished when audio is being stored in a backup service. Currently, Monkey's Audio is still the preferred choice with the best compromise of encoding/decoding times and an attractive, although not completely open source, license.

Acknowledgement

The project (Data compression paradigm based on omitting self-evident information, J2-4458) was financed by Slovenian Research and Innovation Agency and Czech Science Foundation (project No. 23-04622L) from the state budget.

References

[1] J. Coalson, "FLAC - Free Lossless Audio Codec". Accessed: Jun. 29, 2022. [Online]. https://xiph.org/flac/

[2] T. Robinson, "SHORTEN: Simple lossless and near-lossless waveform compression". University of Cambridge, Department of Engineering, 1994.

[3] D. Zhang and D. Zhang, "Wavelet transform", Fundamentals of image data mining: Analysis, Features, Classification and Retrieval, pp. 35–44, 2019.

[4] D. Salomon and G. Motta, Handbook of Data Compression, 5th ed. Springer Publishing Company, Incorporated, 2009.

[5] L. R. Rabiner and R. W. Schafer, "Introduction to Digital Speech Processing," Foundations and Trends® in Signal Processing, vol. 1, no. 1–2, pp. 1–194, 2007, doi: 10.1561/2000000001.

[6] I. Cohen et al., "Pearson correlation coefficient," Noise reduction in speech processing, pp. 1–4, 2009.

[7] B. Iser, W. Minker, and G. Schmidt, Bandwidth extension of speech signals. Springer, 2008.

[8] R. F. Rice, "Some practical universal noiseless coding techniques," 1979.

[9] S. Golomb, "Run-length encodings (Corresp.)," IEEE Transactions on Information Theory, vol. 12, no. 3, pp. 399–401, 1966, doi: 10.1109/TIT.1966.1053907.

[10] W. J. Reed, "The normal-Laplace distribution and its relatives," in Advances in distribution theory, order statistics, and inference, Springer, 2006, pp. 61–74.

[11] T. Liebchen, "MPEG-4 ALS – The Standard for Lossless Audio Coding," Journal of the the Acoustical Society of Korea, vol. 28, no. 7, pp. 618–629, Oct. 2009.

[12] "IEEE Standard for Binary Floating-Point Arithmetic," ANSI/IEEE Std 754-1985, pp. 1–20, 1985, doi: 10.1109/IEEESTD.1985.82928.

[13] J. H. Ginsberg (auth.), Acoustics-A Textbook for Engineers and Physicists: Volume I: Fundamentals, 1st ed. Springer International Publishing, 2018.

[14] E. N. Gilbert and E. F. Moore, "Variable-Length Binary Encodings," Bell System Technical Journal, vol. 38, no. 4, pp. 933–967, 1959.

[15] M. T. Ashland, "Monkey's Audio-a fast and powerful lossless audio compressor". 2011.

[16] T. Mineo and H. Shouno, "A lossless audio codec based on hierarchical residual prediction," in 2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2022, pp. 123–130.

[17] M. Schindler, "A fast renormalisation for arithmetic coding," in Data Compression Conference, IEEE Computer Society, 1998, pp. 0572–0572.

[18] M. A. Nielsen, Neural networks and deep learning, vol. 25. Determination press San Francisco, CA, USA, 2015.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.