Vol.2021-MPS-134 No.1
2021/7/27

NARU: Sign Algorithmの自然勾配法に基づくロスレス音声コーデック

峰尾 太陽[1,a)]　庄野 逸[1,b)]

概要：ロスレス音声圧縮では，エントロピー符号化を適用するにあたり音声信号の残差をスパースにすることが重要である．残差の絶対値を最小化する適応アルゴリズムとして Sign Algorithm (SA) が知られているが，他のアルゴリズムと比べ収束性能が悪い．SA に自然勾配法を適用した Natural Gragient SA (NGSA) が提案され，人口データに対して収束性能の改善が示されたが，実応用研究はされていない．本研究では NGSA に基づく新しいロスレス音声コーデック”NARU” (Natural-gradient AutoRegressive Unlossy Audio Compressor) を提案する．実装は C 言語でオープンソースである．NARU と既存のコーデックの圧縮率とデコード速度比較を行い，圧縮率において優れた性能を示し，デコード速度についても実用的であることを確かめた．

キーワード：ロスレス音声圧縮，適応アルゴリズム，Sign algorithm，自然勾配法，自己回帰モデル

NARU: Natural-gradient AutoRegressive Unlossy Audio Compressor

Taiyo Mineo[1,a)]　Hayaru Shouno[1,b)]

Abstract: In lossless audio compression, it is essential for predictive residuals to remain sparse when applying entropy codings. The sign algorithm (SA) is a conventional method for minimizing the magnitude of residuals; however, it exhibits poor convergence performance compared with the other adaptive algorithms. Although the natural gradient sign algorithm (NGSA) exhibited better convergence performance than SA, its practical applications were not provided yet. This paper proposes a novel lossless audio codec based on the NGSA, called Natural-gradient AutoRegressive Unlossy Audio Compressor (NARU). Its implementation was written by C and open-source. We compared the NARU with existed well-known codecs and showed better compression performance.

Keywords: Lossless audio codec, adaptive algorithm, sign algorithm, natural gradient method, autoregressive model

1. Introduction

デジタル音声コンテンツの高級化に伴って，その記録に必要な記憶容量は増大を続けている [14]．これに伴い，音声を無劣化に圧縮するロスレス音声コーデックは，音声の配信，編集，記録を行うにあたり必須の技術となっている．図 1 に一般的なロスレス音声コーデックの構成を示す [9]．

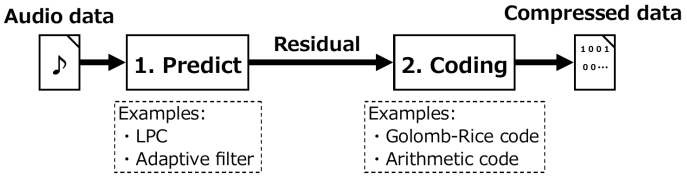

図 1　一般的なロスレス音声コーデックの構成

まず，コーデックは音声信号を予測モデルにより残差信号に変換し，続いてエントロピー符号化によって短いビット列を割り当てる．ここで，モデルの予測が正確であれば残差はスパースになり，高い圧縮率が達成できる．Shoten [18] の提案を契機として，図 1 の構成に従った多くのコーデッ

[1] 電気通信大学
The University of Electro-Communications
[a)] m2040009@uec.ac.jp
[b)] shouno@uec.ac.jp

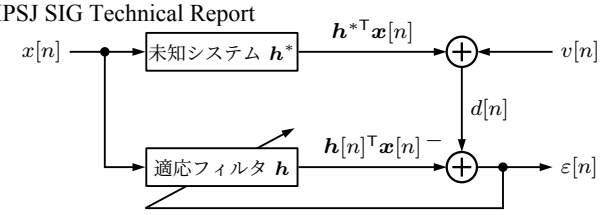

図 2 適応フィルタ

クが開発されている．例えば，MPEG4-ALS [15]，FLAC [12] は線形予測符号化 (Linaer predictive coding, LPC) を予測モデルとして使用しており，WavPack [5]，TTA [21] は適応フィルタを使用している．エントロピー符号化は Golomb-Rice 符号 [17] が一般に使われている．この符号は残差がラプラス分布に従うときに最適な符号化が実現できる．しかし，上記の予測モデルは一般的に残差がガウス分布に従うと仮定して定式化される．この問題を解決するために，[13] は LPC をラプラス分布の仮定の上で定式化した．一方，適応フィルタにおいて残差がラプラス分布に従うアルゴリズムとして Sign Algorithm (SA) [6] が知られているが，収束が遅いことが知られている [7]．これに対処するために SA に自然勾配法を適用した Natural gradient sign algorithm (NGSA) が提案されている．

本研究では，NGSA に基づく新しいロスレス音声コーデック NARU (Natural-gradient AutoRegressive Unlossy) Audio Compressor を提案する．本コーデックの実装は MIT ライセンスの元で公開されている [1]．NARU は FLAC，WavPack，TTA，MPEG-ALS といった既存のコーデックよりも高い圧縮率を示した．Monkey's Audio [3] より圧縮率は劣るが，より高速なデコード速度を実現した．

本稿は以下，次のように構成される．2 節では NARU の数理的基礎を述べ，3 節は NARU のコーデック構成を示す．コーデックの性能比較結果を 4 節で示し，5，6 節で議論と結論を述べる．

2. 数理的背景

2.1 適応フィルタ

適応フィルタの構成を図 2 に示す．ここで，入力信号 $x[n]$ と観測雑音 $v[n]$ は離散時間系列信号である．また本研究では $x[n]$ は弱定常でありエルゴート過程に従うと仮定する．本研究では Finite impulse response (FIR) 型の適応フィルタを考える．すなわち，$\boldsymbol{h}[n] = [h_1[n], ..., h_N[n]]^{\mathsf{T}}$（$\mathsf{T}$: 転置）を適応フィルタの係数としたとき，フィルタの出力を $\boldsymbol{h}[n]^{\mathsf{T}}\boldsymbol{x}[n]$ と表せる．ここで $\boldsymbol{x}[n] = [x[n-N+1], ..., x[n]]^{\mathsf{T}}$ である．未知システムのフィルタ係数を $\boldsymbol{h}^*$ と表す．

フィルタの適応は観測信号

$$d[n] := \boldsymbol{h}^{*\mathsf{T}}\boldsymbol{x}[n] + v[n], \tag{1}$$

と残差信号

$$\varepsilon[n] := d[n] - \boldsymbol{h}[n]^{\mathsf{T}}\boldsymbol{x}[n]. \tag{2}$$

を元に係数 $\boldsymbol{h}[n]$ を更新することで実現される．

2.2 SA と NGSA

Sign Algorithm (SA) は残差 $\varepsilon[n]$ がラプラス分布に従う仮定の元，最尤推定問題を解く過程で得られる．SA の係数適応則を式 (3) に示す．

$$\boldsymbol{h}[n+1] = \boldsymbol{h}[n] + \mu\,\mathrm{sgn}(\varepsilon[n])\boldsymbol{x}[n], \tag{3}$$

ここで，$\mu > 0$ はステップサイズ，$\mathrm{sgn}(\cdot)$ は符号関数である．自然勾配は Fisher 情報行列の逆行列 $\boldsymbol{F}^{-1}$ に損失関数勾配を乗じることで得られる [2]．SA は，

$$\boldsymbol{F} \propto \boldsymbol{R} \tag{4}$$

を満たす．ここで，$\boldsymbol{R}$ は入力信号の自己相関行列である．従って，Natural gradient sign algorithm (NGSA) が以下のように得られる．

$$\boldsymbol{h}[n+1] = \boldsymbol{h}[n] + \mu_{\mathrm{NGSA}}\,\mathrm{sgn}(\varepsilon[n])\boldsymbol{R}^{-1}\boldsymbol{x}[n], \tag{5}$$

ここで，μ_{NGSA} はステップサイズである．

2.3 自己回帰モデル

p 次の自己回帰モデルを AR(p) と表す．このモデルは離散時間信号 s について以下の関係が成立する．

$$s[n] = \sum_{i=1}^{p} \psi_i s[n-i] + \nu[n],\ \psi_i \in \mathbb{R}\ (i = 1, ..., p), \tag{6}$$

ここで，$\nu[n]$ は標準正規分布に従う雑音である．AR(p) 過程に従う信号の自己共分散行列の逆行列 $\boldsymbol{K}_p^{-1}$ の i 行 j 列の成分は以下で陽に計算できる [20]．

$$(\boldsymbol{K}_p^{-1})_{ij} = \begin{cases} \sum_{k=1}^{j} \psi_{i-k}\psi_{j-k} & 1 \le i \le p+1 \\ \sum_{k=1}^{L-i+1} \psi_{L-i+1-k}\psi_{L-j+1-k} & L-p \le j \le L \\ 0 & i \ge j+p+1 \\ \sum_{k=i-p}^{j} \psi_{i-k}\psi_{j-k} & \text{otherwise} \end{cases}, \tag{7}$$

ここで，$i \ge j$，$\psi_0 = 1$ であり，L は $L > 2p$ を満たす行列サイズである．

2.4 効率的な自然勾配更新

入力信号は AR(p) 過程に従うとすると，時刻 n における自然勾配 $\boldsymbol{m}[n] = [m_1[n], ..., m_N[n]]^{\mathsf{T}} := \boldsymbol{K}_p^{-1}\boldsymbol{x}[n]$ は下式で更新できる．

$$\boldsymbol{K}_p^{-1}\boldsymbol{x}[n+1] = \begin{bmatrix} m_2[n] \\ m_3[n] \\ \vdots \\ m_N[n] \\ 0 \end{bmatrix} + m_1[n] \begin{bmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_p \\ \mathbf{0}_{N-p} \end{bmatrix} - m_N[n+1] \begin{bmatrix} \mathbf{0}_{N-p-1} \\ \psi_p \\ \vdots \\ \psi_1 \\ -1 \end{bmatrix},$$
$$m_N[n+1] = x[n+1] - \sum_{i=1}^{p} \psi_i x[n+1-i], \tag{8}$$

ここで，$\mathbf{0}_N$ は $N \times 1$ サイズのゼロベクトルである．式 (8) は $3p$ 回の積和演算で実行できるため，自然勾配を $\mathcal{O}(p)$ のオーダーで更新できる．

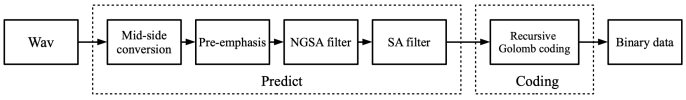

図 3 NARU のエンコーダ構成

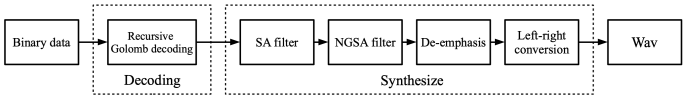

図 4 NARU のデコーダ構成

3. コーデック構成

3.1 エンコーダ

NARU のエンコーダ構成を図 3 に示す．本節ではエンコーダの各要素について説明する．

3.1.1 Mid-side (MS) 処理

Mid-side (MS) 処理はステレオ音声信号のチャンネル間の相関を除去する目的で行われ，下式で表される．

$$M = (L + R)/2,\ S = L - R, \tag{9}$$

ここで，L, R, M, S はそれぞれ左，右，Mid，Side チャンネルを表している．

3.1.2 プリエンファシス (Pre-emphasis)

プリエンファシスは次の FIR フィルタで実現される．

$$y[n] = x[n] - \eta\, x[n-1], \tag{10}$$

ここで η は $\eta \approx 1$ を満たす定数であり，$x[n], y[n]$ はそれぞれ時刻 n におけるフィルタの入力と出力である．このフィルタは入力信号の直流成分を除去し，$\boldsymbol{R}$ が悪条件になるのを防ぐ効果がある [16]．本研究では $\mu = 31/32 = 0.96875$ と設定した．これは，除算が 5-bit の右シフトで実現できるためである．

3.1.3 NGSA フィルタ

NGSA フィルタは NARU の中心となる予測モデルで，最も高い次数（$N \leq 64$）を持つ FIR フィルタで構成される．ここで，式 (5) に従ってフィルタ係数を適応し，式 (2) で $d[n] := x[n+1]$ と設定し入力信号に対し適応を行う．

3.1.4 SA フィルタ

圧縮性能を高めるために NGSA フィルタの後段に SA フィルタをカスケード接続 [10] している．このフィルタは NGSA フィルタよりも小さい次数（$N \leq 8$）を持つ．

3.1.5 再帰的ゴロム符号

再帰的ゴロム符号 [19] によって残差信号に短いビット列を割り当てる．この符号は Golomb-Rice 符号に修正を施したものであり，WavPack と TTA で採用実績がある．

3.2 デコーダ

NARU のデコーダ構成を図 4 に示す．デコードはエンコードの逆順に処理を行う．SA フィルタと NGSA フィルタはエンコードと全く同一の予測を行うため，残差信号から入力信号を完全に再構成できる．また，プリエンファシスの逆のデエンファシス (De-emphasis) 処理は

$$x[n] = y[n] + \eta\, x[n-1], \tag{11}$$

によって行われ，M, S を L, R に復元する Left-right 変換は次式で実現される．

$$L = M + (S/2),\ R = M - (S/2). \tag{12}$$

3.3 実装

速度と移植性を重視するため，コーデックは C 言語 [11] で実装した．実装は MIT ライセンスの元で公開した [1]．エンコード/デコード処理は信号を完全に復元するため，固定小数点演算により実現している．固定小数点数は小数部の幅を 15-bit とし，演算時の桁あふれを防ぐために 32-bit 符号付き整数型により保持している．現在は実装を簡潔にするため，16-bit のリニア PCM で符号化された Wav (Waveform Audio File Format) ファイルのみサポートしている．より高いビット深度への対応は将来の課題である．

4. 評価

4.1 実験設定

圧縮性能を評価するための実験設定を以下に示す．比較対象のコーデックの設定を以下に示す．

FLAC version 1.3.2 highest compression (`-8`)
WavPack version 5.4.0 very high quality (`-hh`)
TTA version 2.3 デフォルト設定
Monkey's Audio version 6.14 extra high (`-c4000`)
MPEG4-ALS RM23[4] デフォルト設定．最適化オプション (`-7`) はエンコードに非現実的な時間を要するため使用していない．
NARU NGSA フィルタの次数は 64，AR 次数は 1，SA フィルタの次数は 8

比較にあたり以下の指標を使用した．

$$\text{圧縮率} = \frac{\text{圧縮後のサイズ [byte]}}{\text{元の Wav ファイルサイズ [byte]}} \times 100\ [\%] \tag{13}$$

$$\text{デコード速度} = \frac{\text{デコード時間 [sec]}}{\text{Wav ファイルの音声長 [sec]}} \times 100\ [\%] \tag{14}$$

入力音声は RWC 音楽データセット [8] の一部データを使用した．本データは全て 16-bit/サンプル，サンプリングレート 44100[Hz]，ステレオチャンネルの形式の Wav ファイルで配布されている．データセットの内容と，各カテゴリの音量を Root mean square (RMS) 基準で計測した結果を表 1 に示す．比較実験は Windows 10 OS 上で行った．CPU は Intel(R) Core(TM) i7-9750H 2.6 [GHz] であり，RAM は 32 [GB] である．

4.2 実験結果

圧縮率の比較結果を表 2 に，デコード速度の比較結果を

表 1 データセットの概要

カテゴリ	トラック数	合計サイズ [MB]	平均 RMS [dB]
クラシック	34	2157.0	−25.4
音楽ジャンル	48	1862.3	−19.3
ジャズ	50	2260.0	−19.6
ポピュラー音楽	64	2587.9	−13.0
著作権切れ音源	15	331.9	−15.5
合計	211	9199.0	−18.2

表 2 平均圧縮率比較

カテゴリ ＼ コーデック	FLAC	WavP	TTA	MPEG4	Monkey	NARU
クラシック	44.22	43.91	43.96	43.67	**41.28**	43.30
音楽ジャンル	58.43	57.64	58.20	58.17	**56.21**	57.69
ジャズ	48.12	47.43	47.64	47.51	**45.51**	47.27
ポピュラー音楽	67.91	67.18	67.72	67.70	**65.71**	67.54
著作権切れ音源	61.03	60.47	61.00	60.82	**58.59**	60.25
合計	56.76	56.10	56.49	56.39	**54.32**	56.07

表 3 平均デコード速度比較

カテゴリ ＼ コーデック	FLAC	WavP	TTA	MPEG4	Monkey	NARU
合計	**0.11**	0.41	0.31	0.43	1.06	0.68

表 3 に示す.

5. 議論

表 2，3 から見れるように，Monkey's Audio (Monkey) は最も高い圧縮率を示したが，デコード速度が最も遅い．これは Monkey's Audio が予測に畳み込みニューラルネット，符号に算術符号を使用しており，圧縮率を重視した設計になっているためである．一方，FLAC は圧縮率が低いがデコード速度は最も早く，逆の傾向を示した．

NARU は FLAC，WavPack (WavP)，TTA，MPEG4-ALS よりも高い平均圧縮率を示した．NARU はクラシックとジャズカテゴリの音声に対して高い圧縮率を示す一方，ポピュラー音楽に対しては WavPack が優れた圧縮率を示した．この傾向と表 1 から，NARU は音量の低い音源に対して高い圧縮率を示すことが示唆された．

NARU のデコード速度は Monkey's Audio を除き他のコーデックよりも遅かった．この原因として，フィルタの予測と係数更新に CPU 時間を取られていることを確認している．現在 NARU の実装は最適化をほとんど施していないため，ループ展開や SIMD 命令の積極的な使用により更に高速化を行えるものと見ている．

6. 結論

本研究では自然勾配法に基づくロスレス音声コーデック NARU を提案した．FLAC, WavPack, TTA, MPEG4-ALS よりも優れた圧縮率を示すことを確認した．将来の課題として，高ビット深度への対応や，ハードウェア対応を含めたさらなる最適化が挙げられる．

参考文献

[1] aikiriao: Natural-gradient AutoRegressive Unlossy Audio Compressor, `https://github.com/aikiriao/NARU`.
[2] Amari, S.-I.: Natural gradient works efficiently in learning, *Neural computation*, Vol. 10, No. 2, pp. 251–276 (1998).
[3] Ashland, M.: Monkey's Audio - a fast and powerful lossless audio compressor, `https://monkeysaudio.com`.
[4] Berlin, T.: Forschung: MPEG-4 Audio Lossless Coding (ALS), `https://www.nue.tu-berlin.de/menue/research/research_topic/compression_and_transmission/mpeg_4_audio_lossless_coding_als/parameter/en/#c230252`.
[5] Bryant, D.: WavPack Audio Compression, `http://www.wavpack.com`.
[6] Diniz, P. S.: *Adaptive filtering*, Springer (1997).
[7] Gersho, A.: Adaptive filtering with binary reinforcement, *IEEE Transactions on Information Theory*, Vol. 30, No. 2, pp. 191–199 (1984).
[8] Goto, M., Hashiguchi, H., Nishimura, T. and Oka, R.: RWC Music Database: Popular, Classical and Jazz Music Databases., *Ismir*, Vol. 2, pp. 287–288 (2002).
[9] Hans, M. and Schafer, R. W.: Lossless compression of digital audio, *IEEE Signal processing magazine*, Vol. 18, No. 4, pp. 21–32 (2001).
[10] Huang, H., Franti, P., Huang, D. and Rahardja, S.: Cascaded RLS–LMS prediction in MPEG-4 lossless audio coding, *IEEE transactions on audio, speech, and language processing*, Vol. 16, No. 3, pp. 554–562 (2008).
[11] ISO: Programming Language C, *ISO/IEC 9899 : 1990* (1990).
[12] Josh Coalson, X. F.: FLAC - Free Lossless Audio Codec, `https://xiph.org/flac/`.
[13] Kameoka, H., Kamamoto, Y., Harada, N. and Moriya, T.: A Linear Predictive Coding Algorithm Minimizing the Golomb-Rice Code Length of the Residual Signal, *The Transactions of the Institute of Electronics,Information and Communication Engineers. A*, Vol. 91, pp. 1017–1025 (2008).
[14] Konstantinides, K.: An introduction to super audio CD and DVD-audio, *IEEE signal processing magazine*, Vol. 20, No. 4, pp. 71–82 (2003).
[15] Liebchen, T.: MPEG-4 ALS-the standard for lossless audio coding, *the Journal of the Acoustical Society of Korea*, Vol. 28, No. 7, pp. 618–629 (2009).
[16] Markel, J. E. and Gray, A. H.: *Linear Prediction of Speech*, Springer-Verlag, Berlin, Heidelberg (1982).
[17] Rice, R. F.: Practical universal noiseless coding, *Applications of Digital Image Processing III*, Vol. 207, International Society for Optics and Photonics, pp. 247–267 (1979).
[18] Robinson, T.: SHORTEN: Simple Lossless and Near-Lossless Waveform Compression, *Technical Report, Cambridge Univ., Eng. Dept.* (1994).
[19] Salomon, D.: *Data compression - The Complete Reference, 4th Edition.* (2007).
[20] Siddiqui, M.: On the inversion of the sample covariance matrix in a stationary autoregressive process, *The Annals of Mathematical Statistics*, Vol. 29, No. 2, pp. 585–588 (1958).
[21] Software, T.: TTA Lossless Audio Codec - True Audio Compressor Algorithms, `http://tausoft.org/wiki/True_Audio_Codec_Overview`.