複数のオーディオエフェクトが適用された楽音に対するエフェクトチェイン推定と原音復元*

☆武 伯寒（東大院・情報理工），渡邉 研斗，中塚 貴之，Tian Cheng，中野 倫靖，
後藤 真孝（産総研），高道 慎之介，猿渡 洋（東大院・情報理工）

1 はじめに

音響信号を加工するツールであるオーディオエフェクト (AFX) は, 音楽制作や楽器演奏のサウンドデザインにおいて必要不可欠である [1]. 具体的には Fig. 1のように, 楽器の演奏音（楽音）に対して Delay や Reverb などの複数の AFX を順番に適用することで, 楽曲・演奏に馴染んだ楽音を作り出すことが多い. この AFX の列はエフェクトチェインと呼ばれており (本稿では AFX chain とする), AFX の選択とその内部パラメータ値の設定, AFX を適用する順番で定義される. サウンドデザイナーは所望の楽音を作り出すために, AFX chain が適用された既存楽曲中の楽音を参考にしながら, どの AFX をどんなパラメータでどのような順番で適用するかを試行錯誤して調整することがある. もし, 既存楽曲中の楽音から, AFX の種類, 内部パラメータ値, 適用順, そして AFX chain 適用前の原音の自動推定が可能であれば, サウンドデザイン作業の効率化や, サウンドデザインに不慣れな初学者に対する AFX の特性理解支援が実現できる.

そこで本稿では, AFX chain が適用された楽音信号 (wet signal) から, chain 中の各 AFX の種類と内部パラメータ値を推定して, AFX chain を適用する前の原音信号 (dry signal) を復元するタスク（**AFX chain 推定・原音復元タスク**）に対して, その解決手法を提案する. 具体的には, 最後に適用された単一の AFX の種類と内部パラメータ値を推定し, その AFX が適用される前の楽音信号 (AFX-bypassed signal) へ復元する深層学習 (DNN) モデルを提案する. このようにして復元された AFX-bypassed signal に対して, 繰り返し提案モデルを適用しながら, 適用された AFX の総数を別途推定することで, AFX chain 全体の推定と, 各段階の AFX が適用される前の AFX-bypassed signal と, AFX chain が適用される前の原音信号の復元を行う.

評価実験では, 複数の AFX を用いたサウンドデザインが頻繁に行われる楽器の一つであるギターのフレーズ演奏音源を対象にした AFX chain 推定・原音復元タスクに取り組む. 具体的には, 提案手法が入力した楽音信号の AFX chain の推定と原音復元を達成できるか, また復元した原音信号に対して推定した AFX chain を再び適用した結果が入力した楽音信号をどれだけ再現するかについて検証する.

2 関連研究

サウンドデザインにおける AFX の重要性から, DNN モデルを用いた AFX の関連研究が発表されてきた. 例えば, 既存の AFX 製品の入出力関係を模倣する研究や [2, 3], AFX chain を構成する AFX の設定情報を自動で制御する研究が行われている [4].

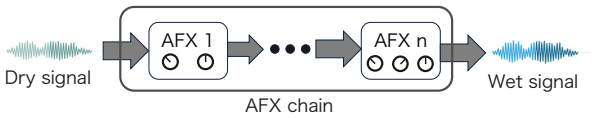

Fig. 1 Process of applying an AFX chain to a signal.

楽音に適用された AFX の設定情報を DNN モデルで推定する研究として, 1 つの AFX の設定情報を推定する手法 [5] や, 複数の AFX を適用する過程と各内部パラメータ値を推定する手法 [6] が提案されている. これらの手法では複数の AFX が適用される前の原音は復元しない. そのため推定した複数の AFX の設定情報を用いて楽音を再現するために, 真の原音が既知である必要がある.

楽音から特定の AFX 一つが適用される前の原音を DNN モデルで復元する手法も研究されている. 具体的には Distortion [7] や Limiter [8] を対象とした手法が提案されている. AFX chain が適用された楽音に対しては, 適用された AFX を全て検出した後, 対応する特定の DNN モデルを一つずつ適用することで原音を推定する手法が提案されている [9]. この手法では AFX chain の適用順を推定していないため, 線形時不変でない加工を施す AFX が含まれる AFX chain が適用された際の原音復元性能の低下が欠点である.

3 エフェクトチェインの推定と原音復元

3.1 問題の定式化

入力の原音信号に対して単一の AFX を適用し, 出力信号を得る過程を次のように定式化する：

$$\boldsymbol{y} = f_{c,\boldsymbol{p}_c}(\boldsymbol{x}). \tag{1}$$

ここで, 原音信号 $\boldsymbol{x} \in \mathbb{R}^{2\times T}$ と出力信号 $\boldsymbol{y} \in \mathbb{R}^{2\times T}$ はいずれも固定サンプル長 T のステレオ音響信号である. $f_{\cdot}$ は AFX の適用を示す関数であり, AFX の設定情報である AFX の種類を表す離散ラベル c と, 対応する内部パラメータ値 $\boldsymbol{p}_c$ によってその関数形が決定される. 内部パラメータの数は AFX の種類 c ごとに異なる. 本稿では内部パラメータは全て連続値とする. AFX chain は様々な AFX を複数適用する合成関数 F として表すことができる. 原音信号 $\boldsymbol{x}$ に適用して出力信号 $\boldsymbol{y}$ を得る過程は次のように定式化する：

$$\boldsymbol{y} = F(\boldsymbol{x}) = f_{c_n,\boldsymbol{p}_{c_n}^{(n)}} \circ \cdots f_{c_2,\boldsymbol{p}_{c_2}^{(2)}} \circ f_{c_1,\boldsymbol{p}_{c_1}^{(1)}}(\boldsymbol{x}). \tag{2}$$

ここで n は AFX chain の長さであり, 原音信号に適用される AFX の総数を表す. AFX chain 推定・原音復元タスクでは, 出力信号 $\boldsymbol{y}$ から F を推定し, F を適用する前の原音信号 $\boldsymbol{x}$ を復元する. $F, \boldsymbol{x}$ の推定量をそれぞれ $\hat{F}, \hat{\boldsymbol{x}}$ と記す. 実際に $\hat{F}$ を決定する際は, これを構成する AFX それぞれの種類 c_i と内部パラメー

*Estimation of audio effect chain and recovery of dry signal from multi-effect-applied musical signal by TAKE, Osamu (The University of Tokyo), WATANABE, Kento, NAKATSUKA, Takayuki, CHENG, Tian, NAKANO, Tomoyasu, GOTO, Masataka (National Institute of Advanced Industrial Science and Technology (AIST)), TAKAMICHI, Shinnosuke, SARUWATARI, Hiroshi (The University of Tokyo)

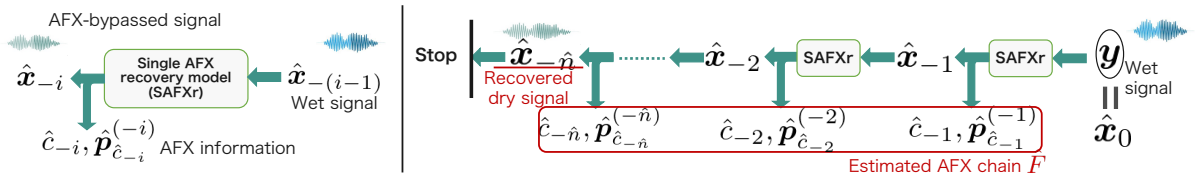

Fig. 2 (Left) Overview of the proposed Single AFX recovery model (SAFXr). (Right) Overview of our proposed method that iteratively applies SAFXr for AFX chain estimation and dry signal recovery.

タ値 $\boldsymbol{p}_{c_i}^{(i)}$, そして AFX chain の長さ n を推定する.

3.2 提案手法：DNN 推定モデルの繰り返し適用

提案手法では Fig. 2に示すように, wet signal $\boldsymbol{y}$ を起点として, 音響信号生成時に最後に適用した AFX の設定情報, そして推定した AFX をバイパスした[1] AFX-bypassed signal を復元する DNN モデルである **単一 AFX 推定・バイパスモデル** (Single AFX recovery model, SAFXr) を繰り返し適用する. SAFXr は Bypassed Signal Estimator (BSEst) g と AFX Info Estimator (AFXIEst) h に分かれ, AFX を一つずつ推定してバイパスした信号を復元する：

$$\hat{c}_{-i}, \hat{\boldsymbol{p}}_{\hat{c}_{-i}}^{(-i)} = h(\hat{\boldsymbol{x}}_{-(i-1)}),\ h_{-i} = f_{\hat{c}_{-i}, \hat{\boldsymbol{p}}_{\hat{c}_{-i}}^{(-i)}}, \quad (3)$$

$$\hat{\boldsymbol{x}}_{-i} = g(\hat{\boldsymbol{x}}_{-(i-1)}, h(\hat{\boldsymbol{x}}_{-(i-1)})). \quad (4)$$

ただし, $\hat{\boldsymbol{x}}_0 = \boldsymbol{y}$ である. AFX chain の長さを別途 $\hat{n}$ として推定した上で, (3)(4) 式は $i = 1, 2, \cdots, \hat{n}$ に対して昇順で計算する. $\hat{\boldsymbol{x}} = \hat{\boldsymbol{x}}_{-\hat{n}}$, $\hat{F} = h_{-1} \circ \cdots \circ h_{-(\hat{n}-1)} \circ h_{-\hat{n}}$ とすることで, AFX chain 推定と原音復元が達成される.

原音復元のみを扱うタスクにおいて, 複数の AFX を一括で除去するのではなく, 推定した AFX を一つずつ除去することでより高い性能を実現したことが報告されている [9]. よって, SAFXr を繰り返し適用して 1 つずつ AFX を推定する提案手法も, F を一挙に推定する手法に比べて学習効率や性能が向上すると考えられる. また, 提案手法の応用上の特長として, 原音信号 $\hat{\boldsymbol{x}}_{-\hat{n}}$ だけでなく, 途中過程で推定された復元信号 $\hat{\boldsymbol{x}}_{-i}$ が得られる点が挙げられる. 例えば, サウンドデザイナーが途中の復元信号 $\hat{\boldsymbol{x}}_{-i}$ を試聴することで, その後の AFX の種類や個数 n を変更する試行錯誤を実現可能とするなど, AFX chain の再利用・再構成によるインタラクティブで新しいサウンドデザインの可能性が切り拓かれる.

AFX chain の長さ $\hat{n}$ の推定 本稿では, AFX が何も適用されていない信号が SAFXr に入力された際, 復元前後の入出力 $\hat{\boldsymbol{x}}_{-(i-1)}$ と $\hat{\boldsymbol{x}}_{-i}$ の間で波形がほぼ変動しないと仮定する. この仮定に基づき, SAFXr 入出力前後の差が閾値より小さい場合, 入力した楽音信号には AFX が適用されていないと判定して適用を終了し, 直前までに推定された AFX とその数 $\hat{n}$ により $\hat{F}$ を構成する.

3.3 単一 AFX 推定・バイパスモデル

提案手法において用いる SAFXr の概要を Fig. 3に示す. BSEst は AFX を扱った関連研究 [7, 9] において高い性能を実現した波形入力/波形出力のモデル [10, 11] に Transformer エンコーダを組み込んだ

[1] AFX 適用前後の楽音は dry-wet を用いて表されることが多いが, 本稿ではその中間で AFX 一つが適用される前の楽音信号を表現するためにバイパス (bypassed) という単語を用いる. 詳しくは Fig. 2, 3を参照されたい.

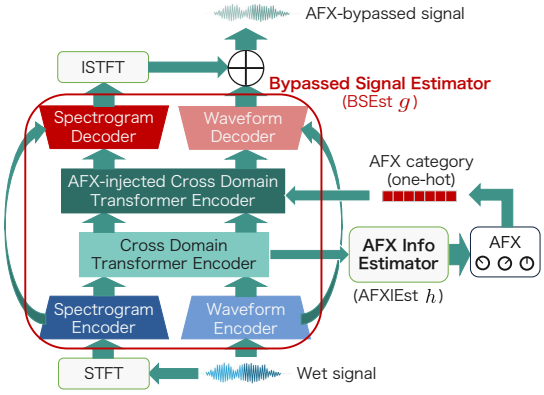

Fig. 3 Architecture of Single AFX recovery model (SAFXr) based on Hybrid Transformer Demucs [12].

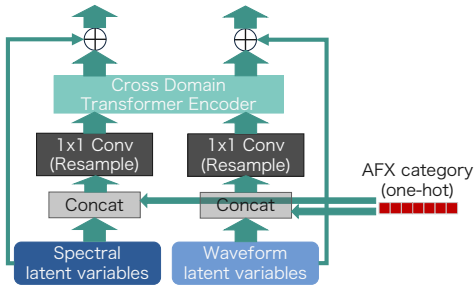

Fig. 4 Architecture of AFX-injected Cross Domain Transformer Encoder to condition SAFXr using AFX category.

Hybrid Transformer Demucs (HTDemucs) [12] を, さらに拡張したモデルである. 具体的には, BSEst のボトルネック部に存在する Cross Domain Transformer Encoder (CDTEnc) を AFX の設定情報で条件付けるため, Fig. 4に示す AFX-injected Cross Domain Transformer Encoder (AFXCDTEnc) を追加する. AFXIEst は, CDTEnc の出力する潜在表現を入力として AFX の設定情報を推定する. このとき, モデルの学習のために AFXIEst は推定した AFX の種類 $\hat{c}_{-i}$ に対応する内部パラメータ値 $\hat{\boldsymbol{p}}_{\hat{c}_{-i}}^{(-i)}$ に限らず, 扱う全ての AFX の種類に対し, それぞれが適用された場合の内部パラメータ値 $\hat{\boldsymbol{p}}^{(-i)}$ も全て推定して出力しておく. AFXCDTEnc では, 直前の CDTEnc から得られた潜在表現と, AFX の種類の one-hot ベクトルを結合し, 内部にある別の CDTEnc に渡す. 実験で学習した SAFXr の詳細は 4.2 節で述べる.

SAFXr の学習プロセス SAFXr の学習は 2 段階に分けることにより, 安定した学習を行う. まず第 1 段階の学習では, AFX の設定情報の推定は行わず, BSEst のみを学習する. この時, c は真の値を用いる. この学習における目的関数 $\mathcal{L}_1$ は真の AFX-bypassed signal $\boldsymbol{x}_{-i}$ と SAFXr によって復元した $\hat{\boldsymbol{x}}_{-i}$ から次のように計算する：

$$\mathcal{L}_1 = \mathcal{L}_{\mathrm{MSE}}(\hat{\boldsymbol{x}}_{-i}, \boldsymbol{x}_{-i}) + 0.05\mathcal{L}_{\mathrm{mrstft}}(\hat{\boldsymbol{x}}_{-i}, \boldsymbol{x}_{-i}). \quad (5)$$

ここで，$\mathcal{L}_{\mathrm{MSE}}$ は平均二乗誤差，$\mathcal{L}_{\mathrm{mrstft}}$ は Multi-resolution STFT loss [13] を表す．その実装は auraloss [14] を用いた．なお，(5) 式の係数 0.05 は予備実験に基づき決定した．次に第 2 段階の学習では，BSEst のパラメータを固定し，AFXIEst のみ学習する．この学習における目的関数 $\mathcal{L}_2$ は，AFXIEst の出力した AFX の設定情報 $(\hat{c}_{-i}, \hat{\boldsymbol{p}}^{(-i)}_{\cdot})$ と $(c_{-i}, \boldsymbol{p}^{(-i)}_{c_{-i}})$ から次のように計算する：

$$\mathcal{L}_2 = \mathcal{L}_{\mathrm{ce}}(\hat{c}_{-i}, c_{-i}) + \mathcal{L}_{\mathrm{MAE}}(\hat{\boldsymbol{p}}^{(-i)}_{c_{-i}}, \boldsymbol{p}^{(-i)}_{c_{-i}}). \quad (6)$$

ここで，$\mathcal{L}_{\mathrm{MAE}}$ は平均絶対誤差，$\mathcal{L}_{\mathrm{ce}}$ は cross entropy loss を表す．

4 評価実験

4.1 データセットとその作成

本実験ではデータセットとして，ギターのフレーズ演奏に関するデータセットである GuitarSet [15]，IDMT-SMT-Guitars [16] と，Slakh2100 [17] に収録されているギターのトラックを用い，ランダムに 10 秒の有音区間を切り出して原音信号とした．原音信号のサンプリング周波数は全て 48 kHz に変換した．原音信号について，振幅の正規化は行わなかった．

AFX chain を構成する AFX は Python の音情報処理用ライブラリである pedalboard [18] に内蔵されている中から，関連研究 [9] や実際のギターのサウンドデザインでよく用いられる Distortion, Delay, Chorus, Reverb の 4 つを選んだ．本稿では，推定する内部パラメータとして連続値のパラメータのみを選択した．各原音信号ごとに，4 種類の AFX から高々 1 回は現れる AFX chain として，適用順も考慮した $\sum_{k=1}^{4} {}_4P_k = 64$ 通りの AFX chain を生成し，それぞれを正解の AFX chain とするデータを生成した．内部パラメータの値域は事前知識に基づき経験的に設定した．SAFXr の学習データでは内部パラメータの最小値を 0，最大値を 1 とする正規化を行った．この設定に基づき AFX 内部パラメータの値をランダムに振って AFX chain を原音信号に適用することにより，データセットを作成した．

データセットは互いに原音信号区間が重複しないように train/valid/test 用に分割した．train/valid/test はそれぞれ 287424/58368/36608 個の wet signal $\boldsymbol{y}$，最後に適用された AFX の設定情報 $c_{-1}, \boldsymbol{p}^{(-1)}_{c_{-1}}$ とその AFX 適用前の AFX-bypassed signal $\boldsymbol{x}_{-1}$ の組で構成した．これを用いて，SAFXr は AFX chain 適用後の wet signal $\boldsymbol{y}$ から $c_{-1}, \boldsymbol{p}^{(-1)}_{c_{-1}}$ を推定し，$\boldsymbol{x}_{-1}$ を復元するよう学習する．

4.2 モデルパラメータと学習条件

本実験で学習した SAFXr のパラメータや学習条件は文献 [12] で用いられた値をベースとした．ただし，BSEst において Spectrogram Encoder への入力に用いる STFT の窓長は 8192 サンプルとした．Waveform/Spectrogram Encoder の第 1 層の出力チャネル数は 32 とした．前後段の Encoder/Decoder は 6 層で揃えて構成した．また，CDTEnc, AFXCDTEnc はそれぞれ 3 層，2 層の Transformer レイヤーで構成した．このモデルを cdt-3-afxcdt-2 と呼ぶ．比較手法として，AFX の設定情報でモデルを条件付けることの有効性を検証するために，AFXCDTEnc が含まれず，CDTEnc が 5 層の Transformer レイヤーで構成されるモデルである cdt-5-afxcdt-0 を学習した．また，CDTEnc の Transformer レイヤーの存在による性能の変化を観察するために，CDTEnc, AFXCDTEnc がそれぞれ 0 層，5 層の Transformer レイヤーで構成されるモデルである cdt-0-afxcdt-5 を学習した．

AFXIEst は 3 層の畳み込みブロックと 3 層の全結合ブロックにより構成した．畳み込みブロックは時系列方向の 1 次元畳み込み層と 1 次元バッチ正規化，ReLU 関数により構成され，最終層出力を時間方向に平均・最大をとってその和を全結合ブロックに入力した．全結合部は AFX の種類と内部パラメータそれぞれについて構成した．全結合ブロックは全結合層とバッチ正規化，ReLU により構成され，全結合層の隠れ次元数は全て 512 とした．また，AFX の種類と内部パラメータ値を推定する全結合部にそれぞれ確率 0.2, 0.05 の dropout を学習時に適用した．cdt-3-afxcdt-2 と cdt-5-afxcdt-0 において，AFXIEst への入力は CDTEnc の出力，また cdt-0-afxcdt-5 においては AFXCDTEnc に入力される潜在表現とした．

モデルの学習において，バッチサイズは 32，学習率は 5×10^{-5} とした．SAFXr の第 1 段階の学習では 400 エポックの学習を行って BSEst のパラメータを更新し，第 2 段階ではそのパラメータを固定して 150 エポック学習して AFXIEst のパラメータを更新した．

Table 1 Results of AFX-bypassed signal $\hat{\boldsymbol{x}}_{-1}$ recovery.

Model	**SI-SDR**(↑)	**MR-STFT**(↓)
Wet $\boldsymbol{y}$	10.57 dB	2.32
cdt-3-afxcdt-2	**13.63 dB**	**0.68**
cdt-0-afxcdt-5	13.30 dB	0.69
cdt-5-afxcdt-0	9.74 dB	1.06

4.3 評価指標

提案手法の原音復元性能と AFX chain 適用後の wet signal $\boldsymbol{y}$ の再現性能を測る指標として，本稿では関連研究 [9] に倣い，波形での誤差を測る Scale-Invariant SDR (SI-SDR) [14, 19] と，周波数領域での誤差を測る Multi-resolution STFT loss (MR-STFT) を採用した．AFXIEst の性能を測る指標としては，最後に適用された AFX の種類 c_{-1} 推定の正解率 (AFX Acc.) と，内部パラメータ値推定の平均二乗誤差 (Param. MSE) を採用した．内部パラメータ値については，正規化した値同士の誤差を調べた．

4.4 単一 AFX 推定・バイパスモデルの評価実験

本節では SAFXr により AFX-bypassed signal $\boldsymbol{x}_{-1}$ 復元と，最後に適用された AFX の設定情報推定の性能を検証する実験を行った．Table 1に学習した BSEst による $\boldsymbol{x}_{-1}$ の復元性能の各評価指標を示す．ここで，ベースラインである Wet とは入力となる wet signal $\boldsymbol{y}$ に何もモデルを適用しない場合の性能を示す．この評価において，AFXCDTEnc は真の最後に適用した AFX の設定情報で条件付けた．この結果から，cdt-5-afxcdt-0 以外については，いずれの評価指標でもモデルを適用しない Wet に比べて評価指標の値が改善していることがわかった．よって，BSEst は AFX の設定情報で条件付けるモデルを用いることで初めて，入力の信号と比較して正解である $\boldsymbol{x}_{-1}$ に近い楽音信号が得られることがわかった．

Table 2　Results of AFX info. estimation.

Model	AFX Acc.(↑)	Param. MSE(↓)
cdt-3-afxcdt-2	0.695	0.034
cdt-0-afxcdt-5	0.724	**0.030**
cdt-5-afxcdt-0	**0.728**	0.038

Table 3　Results of dry signal recovery.

Model	SI-SDR(↑)	MR-STFT(↓)
Wet $\boldsymbol{y}$	1.26 dB	5.67
MAE threshold		
cdt-3-afxcdt-2	**6.25 dB**	2.07
cdt-0-afxcdt-5	5.75 dB	2.23
SDR threshold		
cdt-3-afxcdt-2	5.09 dB	1.95
cdt-0-afxcdt-5	4.54 dB	**1.88**

次に, Table 2にて学習した AFXIEst の AFX 設定情報推定結果を示す．この結果から, cdt-3-afxcdt-2 は少し性能が劣るものの, いずれもモデルも同程度の推定性能を達成できていることがわかった．これらの実験結果から, 以後の実験では SI-SDR と MR-STFT の双方の指標のもとで入力の信号と比較した時に原音復元を行える, cdt-3-afxcdt-2 と cdt-0-afxcdt-5 について評価する．

4.5　提案手法による原音復元性能の評価

本節の実験では, 4.4節で学習した SAFXr を用いてデータセット内の $\boldsymbol{y}$ を生成した F を推定し, $\boldsymbol{x}$ を復元する実験を行った．この時, 長さ $\hat{n}$ の推定のための入出力 $\hat{\boldsymbol{x}}_{-(i-1)}, \hat{\boldsymbol{x}}_{-i}$ 間の差分として, MAE threshold Δ_{ℓ_1} と SDR threshold Δ_{SDR} の 2 つを用いて比較した：

$$\Delta_{\ell_1} = \mathcal{L}_{\mathrm{MAE}}(\hat{\boldsymbol{x}}_{-(i-1)}, \hat{\boldsymbol{x}}_{-i}) / ||\hat{\boldsymbol{x}}_{-(i-1)}||_1, \quad (7)$$

$$\Delta_{\mathrm{SDR}} = |\mathrm{SDR}(\boldsymbol{y}, \hat{\boldsymbol{x}}_{-(i-1)}) - \mathrm{SDR}(\boldsymbol{y}, \hat{\boldsymbol{x}}_{-i})|. \quad (8)$$

予備実験により, 閾値は それぞれ 0.2, 1 dB とした．評価実験の結果を Table 3に示す．

この結果から, 提案した AFX chain 推定・原音復元モデルにより, 出力信号 (Wet $\boldsymbol{y}$) と比較して原音信号を推定できていることがわかった．SI-SDR により評価した場合は MAE threshold, MR-STFT の場合は SDR threshold を用いたモデルの方が性能が高かった．次に, SAFXr として cdt-3-afxcdt-2 を用いた時の, test データセットにおける $\hat{F}$ の長さの平均を調べた．その結果, 正解の AFX chain F の長さ n の平均が 3.06 であったのに対し, SDR threshold により推定した $\hat{F}$ は 1.89, MAE threshold を用いた場合は 2.41 であった．以上の結果により, SAFXr の平均的な適用回数がより大きかった MAE threshold を用いると, 波形での評価では原音復元性能は改善し, 周波数領域での評価では僅かに劣化すると考えられる．

4.6　エフェクトチェイン適用後信号再現性能の評価

本節では推定 AFX chain $\hat{F}$ を用いて, AFX chain 推定・原音復元手法に入力した wet signal $\boldsymbol{y}$ を再現できるかを評価する実験を行った．4.5節の SI-SDR による原音復元の評価結果を重視し, 提案手法の停止判断のための差分の指標として MAE threshold を選択した．復元した原音信号 $\hat{\boldsymbol{x}}$ と真の原音信号 $\boldsymbol{x}$ それぞれに $\hat{F}$ を適用し, 得られた楽音信号の各評価指標値の改善量 SI-SDRi ($\mathrm{SI\text{-}SDR}(\boldsymbol{y}, \hat{F}(\cdot)) - \mathrm{SI\text{-}SDR}(\boldsymbol{y}, \cdot)$), MR-STFTi ($\mathcal{L}_{\mathrm{mrstft}}(\boldsymbol{y}, \cdot) - \mathcal{L}_{\mathrm{mrstft}}(\boldsymbol{y}, \hat{F}(\cdot))$) を用いて性能を調べた．その結果を Table 4に示す．なお, 入力楽音に AFX が適用されていない（$\hat{F}$ = id, 恒等写像, $\hat{n} = 0$）と提案手法が推定したケースにおいて, SI-SDRi, MR-STFTi はそれぞれ 0 とした．

Table 4　Results of wet signal reproduction using estimated AFX chain $\hat{F}$.

Model	SI-SDRi(↑)	MR-STFTi(↑)
Reproduced from recovered signal $\hat{\boldsymbol{x}}$		
cdt-3-afxcdt-2	**1.53 dB**	**0.71**
cdt-0-afxcdt-5	−0.09 dB	0.58
Reproduced from ground-truth signal $\boldsymbol{x}$		
cdt-3-afxcdt-2	**2.77 dB**	**0.33**
cdt-0-afxcdt-5	2.47 dB	0.32

この結果から, cdt-3-afxcdt-2 を用いた手法では $\hat{\boldsymbol{x}}$ に推定した AFX chain $\hat{F}$ を適用する場合は SI-SDRi が正であったため, 入力した wet signal $\boldsymbol{y}$ を再現することが可能であることがわかった．一方で, cdt-0-afxcdt-5 では SI-SDRi は負, すなわち $\boldsymbol{y}$ の再現に失敗していることがわかった．$\boldsymbol{x}$ に $\hat{F}$ を適用した場合は, いずれの SAFXr を用いても SI-SDRi は正であり, $\boldsymbol{y}$ の再現が可能であることがわかった．

5　おわりに

本稿では AFX chain 推定・原音復元タスクを提唱し, このタスクに対して単一 AFX 推定・バイパスモデルを繰り返し適用することで AFX chain を推定する手法を提案した．ギターのフレーズ演奏音源を用いた評価実験により, 提案手法が楽音の原音復元と, 推定した AFX chain による wet signal の再現を行うのに有効であることを確認した．また比較実験により, 提案手法では, 推定した AFX の設定情報で条件付ける構造が性能向上につながることがわかった．今後はより一般的な楽音信号や AFX に対応できる AFX chain 推定・原音復元モデルの構築や, 提案手法を実装したサウンドデザイン支援ツールの作成・検証を行っていく予定である．

謝辞 本研究は科研費 21H04900, 22H03639，23H03418, JST 創発的研究支援事業 JP23KJ0828，ムーンショット JPMJPS2011 の助成を受けたものです．

参考文献

[1] T. Wilmering *et al.*, Appl. Sci., 10(3), 791, 2020.
[2] M. A. Martinez–Ramirez *et al.*, Appl. Sci., 10(2), 638, 2020.
[3] C. J. Steinmetz, J. D. Reiss, Proc. 152nd Convention of the AES, 1–9, 2022.
[4] C. J. Steinmetz *et al.*, JAES, 70(9), 708–721, 2022.
[5] M. Comunitá *et al.*, JAES, 69(7, 8), 594–604, 2021.
[6] S. Lee *et al.*, Proc. ICASSP, 1–5, 2023.
[7] J. Imort *et al.*, Proc. ISMIR, 218–225, 2022.
[8] C. Jeon, K. Lee, Proc. WASPAA, 1–5, 2023.
[9] M. Rice *et al.*, Proc. WASPAA, 1–5, 2023.
[10] A. Défossez *et al.*, 10.48550/arXiv.1911.13254, 1–16, 2019.
[11] A. Défossez, Proc. MDX Workshop, 1–13, 2021.
[12] S. Rouard *et al.*, Proc. ICASSP, 1–5, 2023.
[13] R. Yamamoto *et al.*, Proc. ICASSP, 6199–6203, 2020.
[14] C. J. Steinmetz, J. D. Reiss, Proc. DMRN+15, 14, 2020.
[15] Q. Xi *et al.*, Proc. ISMIR, 453–460, 2018.
[16] K. Christian *et al.*, Proc. DAFx, 219–226, 2014.
[17] E. Manilow *et al.*, Proc. WASPAA, 45–49, 2019.
[18] S. Peter, 10.5281/zenodo.7817838, 2021.
[19] J. Le Roux *et al.*, Proc. ICASSP, 626–630, 2019.