

# ニューラルネットワークによる 音色の変換

教養学部後期課程学際科学科総合情報学コース

陶山大輝

学籍番号：08-192021

指導教官：金子知適准教授

# 目次

	Page
<b>第 1 章 はじめに</b>	<b>4</b>
<b>第 2 章 背景：音</b>	<b>5</b>
2.1 音の定義	5
2.1.1 音響信号	5
2.1.2 楽音	5
2.2 音の表現	6
2.2.1 一次元データと二次元データ	6
2.2.2 STFT	6
2.2.3 CQT	7
<b>第 3 章 背景：ニューラルネットワーク</b>	<b>8</b>
3.1 教師あり学習	8
3.1.1 教師あり学習の目的	8
3.1.2 教師あり学習モデルの学習と汎化	8
3.2 MLP	8
3.2.1 人工ニューロンの構造と定式化	8
3.2.2 MLP の構造	9
3.2.3 MLP の定式化	10
3.2.4 MLP の学習	10
3.3 CNN	11
3.3.1 CNN と視覚野の関係	11
3.3.2 CNN の畠み込み層	11
3.3.3 CNN のプーリング層	11
3.4 GAN	12
3.4.1 GAN の概要	12
3.4.2 GAN の定式化	12
3.5 Pix2pix	13
3.5.1 Pix2pix の定式化	13
3.5.2 Pix2pix の生成モデル	14
3.5.3 Pix2pix の識別モデル	14
<b>第 4 章 実験</b>	<b>15</b>
4.1 提案手法	15
4.1.1 提案手法の目標	15
4.1.2 提案モデルの構造	15
4.1.3 生成モデルと識別モデル	16
4.2 実験方法	17
4.2.1 データセット	17

4.2.2 実験 1：提案モデルの表現力の評価実験	17
4.2.3 実験 2：提案モデルの汎化能力の評価実験	17
4.2.4 データ拡張	17
4.2.5 評価方法	17
4.3 実験結果	18
4.3.1 実験 1：提案モデルの表現力の評価実験	18
4.3.2 実験 2：提案モデルの汎化能力の評価実験	18
4.3.3 課題	19
<b>第 5 章 まとめ</b>	<b>21</b>
5.1 実験結果のまとめ	21
5.2 展望	21
5.2.1 楽器の重ね合わせ	21
5.2.2 音の重ね合わせ	21
5.2.3 音の繋ぎ方	21
<b>謝辞</b>	<b>22</b>
<b>参考文献</b>	<b>23</b>
<b>付録 A Adam</b>	<b>25</b>
<b>付録 B 学習時のパラメータ</b>	<b>26</b>
<b>付録 C データセットの分割</b>	<b>27</b>
<b>付録 D 実験結果</b>	<b>28</b>
D.1 提案モデルの表現力の評価実験	28
D.2 提案モデルの汎化能力の評価実験	31

# 図目次

	Page
<b>第 2 章 背景：音</b> .....	<b>5</b>
図 2.1 音波 .....	5
図 2.2 音波の拡大図 .....	5
図 2.3 音響信号 .....	6
図 2.4 STFT .....	6
図 2.5 メルスペクトログラム .....	7
図 2.6 CQT .....	7
図 2.7 Rainbowgram .....	7
<b>第 3 章 背景：ニューラルネットワーク</b> .....	<b>8</b>
図 3.1 MLP の人工ニューロン .....	9
図 3.2 MLP のネットワーク .....	9
図 3.3 CNN の畠み込み層 .....	11
図 3.4 CNN のプーリング層 .....	11
図 3.5 GAN の全体図 .....	12
図 3.6 Pix2pix のスタイル変換の例 .....	13
図 3.7 Pix2pix の全体図 .....	13
図 3.8 Pix2pix の生成モデルと識別モデル .....	14
<b>第 4 章 実験</b> .....	<b>15</b>
図 4.1 提案モデルの全体図 .....	15
図 4.2 本研究の生成モデルと識別モデル .....	16
図 4.3 88 鍵のピアノの鍵盤 .....	17
図 4.4 実験 1：ハープの音を表現できた音波 .....	18
図 4.5 実験 1：ハープの音を表現できなかった音波 .....	18
図 4.6 実験 2：ハープの音を表現できた音波 .....	19
図 4.7 実験 2：ハープの音を表現できず音の高さも維持できなかった音波 .....	19
図 4.8 実験 2：ハープの音を表現できず音の高さを維持できた音波 .....	19
図 4.9 課題：音の減衰の表現 .....	20
図 4.10 課題：音の大きさの維持 .....	20
図 4.11 課題：音波の滑らかさの表現 .....	20
図 4.12 課題：データセットの不安定さ .....	20

# 第 1 章 はじめに

音楽は世界中で楽しまれ、その作成方法は技術発展により多様化している。近年注目されている音楽の作成方法としてリミックスと呼ばれるものがある。リミックスは既存の曲に音響操作を加えて新しい曲を作成する方法であり、1970 年代のディスコの発達とともに世界的に普及した。当時はディスコでの DJ による即興のパフォーマンスとして行われていたが、近年のパソコンなどのデジタル機器の発達により音楽の作成方法として一般的なものとなった。

しかし、音の編集などを行うソフトウェアアプリケーション (DAW) の操作がリミックスには必要であり、音楽作成を円滑に行うためには一定の経験が必要となる。したがって、コンピュータプログラムによるリミックスの補助が音楽作成に役立つと考えられる。また、本研究では、リミックスの方法の一つである音色の変換に注目し、ニューラルネットワークによる音色の変換手法を提案する。

そして、本研究では、画像のスタイル変換を行う Pix2pix [1] を元に作成した提案モデルを用いてギターの単音からハープの単音へ音色を変換する実験を行った。その結果、音の高さを維持したまま変換を行うことができたが、ハープの音を表現できたのは一部の音のみであり、音色の変換におけるいくつかの課題が浮かび上がった。

なお、本論文では、音のデータセットを作成する際に楽譜作成ソフトの MuseScore<sup>\*1</sup>を利用し、音波の波形画像を作成する際に音声編集ソフトの Audacity<sup>\*2</sup>を利用した。

---

<sup>\*1</sup> <https://musescore.org/>

<sup>\*2</sup> <https://www.audacityteam.org/>

# 第 2 章 背景：音

本章では、音の定義を行った後、音の表現方法を紹介する。

## 2.1 音の定義

音とは、弾性体中を伝播する波により起こされる音波が聴覚により感じられるもののことである。

### 2.1.1 音響信号

音は時間方向の変化量であるため、音響信号と呼ばれる。音響信号は連続的な信号であるが、コンピュータで扱う際には離散的な信号に変換される。また、変換の際には標本化（サンプリング）と量子化が必要である。まず、サンプリングは一定の時間を空けて離散的に測定を行うことであり、1秒あたりのサンプリング回数をサンプリング周波数と呼ぶ。そして、量子化は信号の大きさを離散的に表現することであり、信号の大きさを表現するビット数を量子化ビット数と呼ぶ。

### 2.1.2 楽音

楽音は周期性のある音波を持つ音のことであり、音楽で用いられる。本論文では、楽音としての音を生成することを目標とする。また、楽音は長さ、大きさ、高さ、音色の四要素を持つ。

#### 2.1.2.1 音の長さと大きさ

音の長さは音波の時間長により決まり、音の大きさは音波の振幅により決まる（図 2.1）。音波の時間長が長いほど音の長さは長くなり、音波の振幅が大きいほど音の大きさは大きくなる。

#### 2.1.2.2 音の高さと音色

音の高さと音色は直感的にはそれぞれ音波の周期構造の長さと形により決まる（図 2.2）。また、音の高さは音波の周波数により決まり、周波数の高い音ほど音の高さは高くなる。そして、音の長さと大きさと高さが同じ時の音の違いを音色と呼び、それぞれの楽器は異なる音色を持つ。

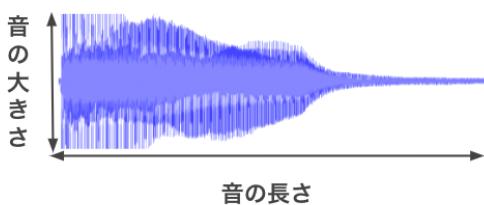


図 2.1: 音波

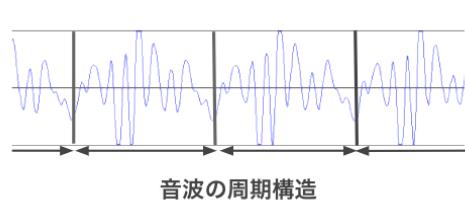


図 2.2: 音波の拡大図

## 2.2 音の表現

音をニューラルネットワークで扱うためには、楽音としての特徴を学習するための適切な表現を得る必要がある。また、本節は [2] の 3.2 節及び [3] の 2.1 節を参考に作成し、本節の図は [2] の Figure 4 と [3] の Figure 2 を利用した。

### 2.2.1 一次元データと二次元データ

音響信号をニューラルネットワークでは一次元データとして扱うが（図 2.3）、素朴な音の表現である一次元データから楽音としての特徴を学習するのは難しい。したがって、[4] のような階層型のネットワークを用いるなどの工夫が必要となる。また、音響信号は複数の周波数成分の合成波であるため、音響信号を周波数成分ごとに分解して二次元データに変換することが多い。ここで、二つの次元としては時間と周波数を選ぶことが一般的で、この時に得られる二次元データを時間-周波数表現と呼ぶ。

### 2.2.2 STFT

STFT (Short Time Fourier Transform) は、同じバンド幅の中間周波数を利用したバンドパスフィルターを用いて音響信号を周波数成分ごとに分解する手法である。また、STFT を用いて生成できる時間-周波数表現を可視化した画像をスペクトログラムと呼び、振幅の大きさを明度などにより表現する（図 2.4）。ただし、通常のスペクトログラムには位相の情報は保持されないため、音響信号からスペクトログラムへの変換は不可逆である。

そして、離散信号に対しての STFT は式 (2.1) により定式化される。ここで、 $m$  は時刻、 $w_k$  は周波数、 $w$  は 0を中心とした窓関数、 $x(n)$  は時間  $n$  における信号  $x$  の値、である。

$$\text{STFT}(m, \omega_k) = \sum_{n=-\infty}^{\infty} x(n)w(n-m)e^{-i\omega_k n} \quad (2.1)$$

また、STFT は人間の聴覚系の周波数分解能に合わせて作られたものではないため、楽音の解析の手法として用いる際はスペクトログラムに何らかの工夫を加えることが多い。ここで、周波数方向の圧縮方向を施した一般的なスペクトログラムとしてはメルスペクトログラムがあげられる（図 2.5）。メルスペクトログラムでは、人間の感じる音の高さの差が等幅になるように調整した尺度であるメル尺度 [5] を用いて周波数方向での圧縮を行う。

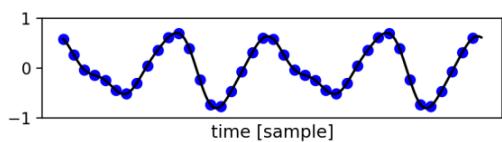


図 2.3: 音響信号

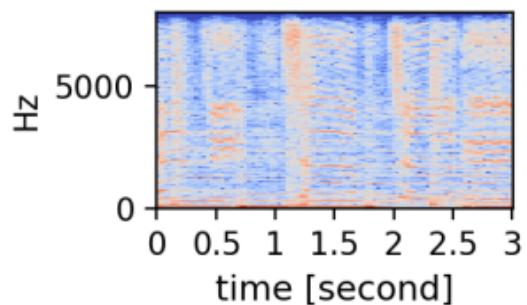


図 2.4: STFT

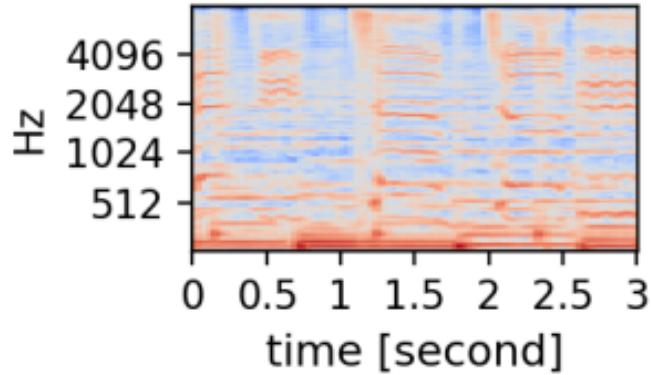


図 2.5: メルスペクトログラム

そして、メル尺度は周波数  $f$  に対して式 (2.2) として定式化される [6]。また、メル尺度には既定のものはないが、いずれでも対数を用いる点は変わらない。

$$mel(f) = 1127.01048 \log\left(\frac{f}{700} + 1\right) \quad (2.2)$$

### 2.2.3 CQT

CQT (Constant Q Transform) [7] は STFT と同様に周波数成分ごとに分解する手法であるが、楽音の解析に適するような二つの工夫を行う。まず、定バンド幅のバンドフィルターを持つ STFT には低周波数成分のサンプル数が少なくなり分解能が低下するという問題があるため、CQT では低周波数成分ほどバンド幅を広げることで分解能を一定にする工夫を行う。そして、人間の聴覚系の分解能は線形よりも対数に近いため、CQT では対数振幅を利用する工夫を行う。

また、CQT により対数振幅を用いたスペクトログラムを生成することができる (図 2.6)。さらに、位相の時間微分をカラーコードにより表現した Rainbowgram も音波の生成時の可視化による評価に用いるスペクトログラムとして [8] にて提案され、[9] や [3] で用いられる (図 2.7)。

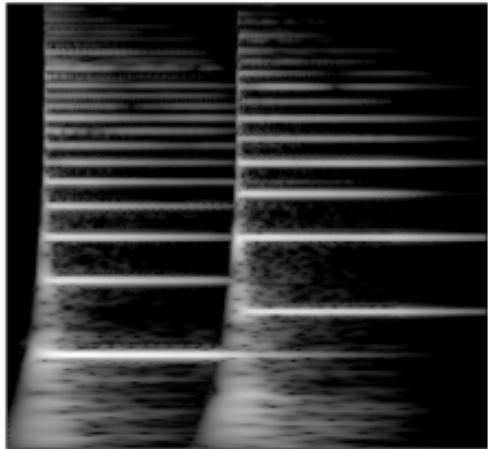


図 2.6: CQT

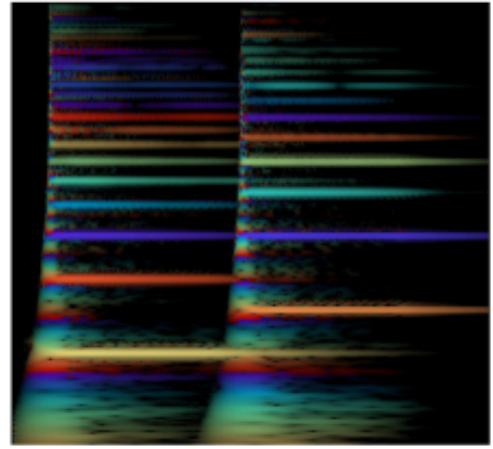


図 2.7: Rainbowgram

# 第 3 章 背景：ニューラルネットワーク

本章では、教師あり学習の説明を行った後、本論文で用いるニューラルネットワークの説明を行う。また、GAN の変換対象となる鞄の画像では [1] の Figure 1 を利用した。

## 3.1 教師あり学習

教師あり学習とは、学習データとして説明変数と対応するべき目的変数のペアが与えられる機械学習の手法である。また、機械学習とは、学習データに含まれる特徴をコンピュータプログラム（モデル）が自動で学習し、学習したモデルを用いて何らかの問題を解く手法のことである。

### 3.1.1 教師あり学習の目的

$X, Y$  をそれぞれ説明変数と目的変数の集合とすると、 $f : X \rightarrow Y$  のうち任意の  $x \in X$  について正しい値  $y \in Y$  を出力する関数  $f'$  を表現するモデルを作成することが教師あり学習の目的である。

また、 $f(x)$  の  $f'(x)$  への近似の程度を評価する関数を損失関数  $L$  と呼ぶ。損失関数  $L$  は  $L : Y \times Y \rightarrow \mathbb{R}^+$  として定義され、値が小さいほど近似の程度が良い関数である。 $\mathbb{R}^+$  は非負の実数を表す。

そして、モデルのパラメータを  $\theta$  とすると、教師あり学習は損失関数の期待値を最小化する  $\theta'$  を求める最適化問題であり、 $\theta'$  により決まる関数  $f$  が  $f'$  となる。

### 3.1.2 教師あり学習モデルの学習と汎化

任意の  $x$  と対応する  $y$  の組を用意することは現実的には難しい。したがって、学習データのみでの最適化問題を解いて最適解として  $\hat{\theta}$  を求めることが教師あり学習モデルの学習の目標である。

しかし、 $\hat{\theta}$  により定まる  $\hat{f}$  は  $f'$  に一致するとは限らない。従って、 $\hat{f}$  の  $f'$  への近似の程度を評価する必要がある。そこで、学習データとは別に評価データを用意し、評価データについての損失関数の期待値を求めることなどにより未知のデータへのモデルの性能（汎化性能）を評価する必要がある。

## 3.2 MLP

MLP (Multilayer perceptron) は教師あり学習に用いられる順伝播型のニューラルネットワークである。また、ニューラルネットワークは神経細胞間のシナプス結合を通じて電気信号により実現される脳の機能に類似した数理モデルの総称であり、神経細胞を人工ニューロンとして表現する。

### 3.2.1 人工ニューロンの構造と定式化

MLP の人工ニューロンは図 3.1 として表現される。まず、 $m$  個の入力  $x_i$  は樹状突起からの入力を表し、重み  $W_i$  は入力  $x_i$  に対応する神経細胞との結合強度を表す。そして、 $b$  は活動電位の発生の閾値を調整するバイアス項であり、 $\theta$  は活動電位の非線形な発生を表す活性化関数である。

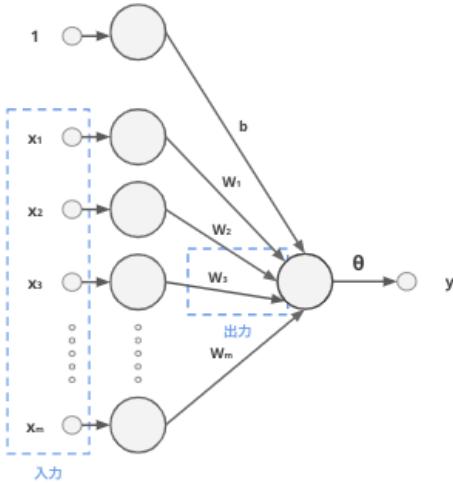


図 3.1: 人工ニューロン

この時、他の神経細胞への出力  $y$  は、 $x_i$  の  $W_i$  を重みとした重み付き和に  $b$  を加えて活性化関数を作用させた値である。したがって、MLP の人工ニューロンは式 (3.1) として定式化される。

$$y = \theta\left(\sum_{i=1}^m W_i \times x_i + b\right) \quad (3.1)$$

### 3.2.2 MLP の構造

MLP は、図 3.1 の人工ニューロンが複数並んだ層を一層とし (図 3.2a) 、順伝播型の階層構造のニューラルネットワークを形成する (図 3.2b) 。また、入力層と中間層と出力層の三つの層構造を持ち、入力層と出力層はそれぞれ一層のみである。ここで、層を構成する人工ニューロンの全ての出力が次の層の全ての人工ニューロンの入力として渡されるため、MLP を構成する層を全結合層と呼ぶ。

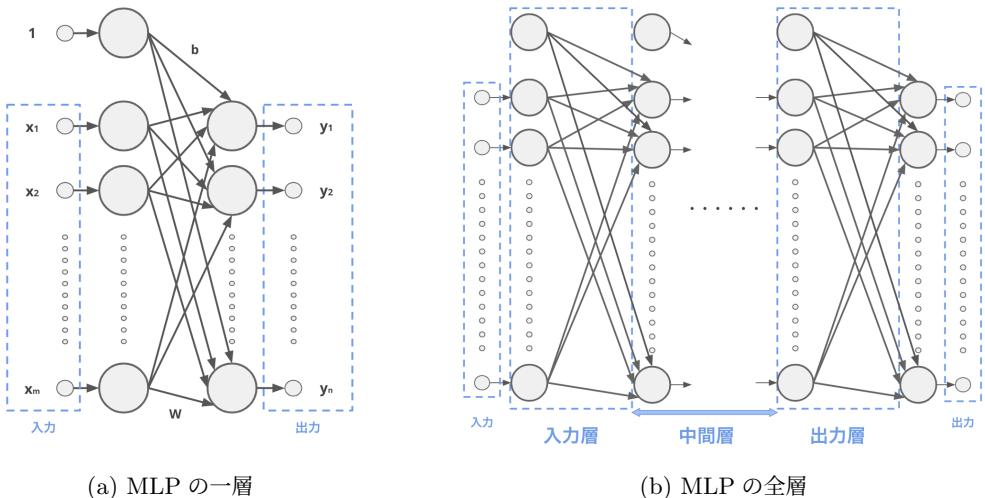


図 3.2: MLP のネットワーク

### 3.2.3 MLP の定式化

式 (3.1) より一層の  $j$  番目の人工ニューロンの出力は式 (3.2) となる。したがって、入力と出力とバイアス項をそれぞれベクトル  $\mathbf{x}, \mathbf{y}, \mathbf{b}$  で表し、重みを行列  $W$  で表すことで、MLP の一層の出力は式 (3.3) と求まる。また、MLP は全結合層のみで構成されるため、何番目の層であるかを下付きの数字で表すことで、 $n$  層の MLP は式 (3.4) として定式化される。

$$y_j = \theta\left(\sum_{i=1}^m W_{ji} \times x_i + b\right) \quad (3.2)$$

$$\mathbf{y} = \theta(W\mathbf{x} + \mathbf{b}) \quad (3.3)$$

$$\mathbf{y} = \theta_n(W_n(\theta_{n-1}(W_{n-1} \cdots (\theta_1(W_1\mathbf{x} + \mathbf{b}_1)) \cdots + \mathbf{b}_{n-1})) + \mathbf{b}_n) \quad (3.4)$$

### 3.2.4 MLP の学習

MLP では実行可能解であるパラメータ  $W_i, \mathbf{b}_i$  の更新を繰り返すことで最適解を求める。この時、損失関数  $L$  の勾配を用いた最適化アルゴリズムである勾配降下法 [10] を用いて更新するのが一般的である。また、本節の手法は次章以降の CNN などの他のニューラルネットワークにおいても用いられる。

#### 3.2.4.1 勾配降下法

勾配降下法の中で単純なアルゴリズムである最急降下法を紹介する。最急降下法では、最も急な降下方向である勾配の反対方向へパラメータの更新を行うため、式 (3.5) として定式化される。

$$W_i, \mathbf{b}_i \leftarrow W_i - \eta \frac{\partial L}{\partial W_i}, \mathbf{b}_i - \eta \frac{\partial L}{\partial \mathbf{b}_i} \quad (3.5)$$

ここで、 $\eta$  は学習率と呼ばれる正の実数であり、更新の程度を指定する。また、本研究では勾配降下法として Adam [11] を用いるため、その挙動を付録 A に示す。

#### 3.2.4.2 誤差逆伝播法

勾配降下法ではそれぞれのパラメータにおいて損失関数の勾配を求める必要があり、高速に勾配を求める手法として誤差逆伝播法が一般に用いられる。また、誤差逆伝播法は大きく二つの段階に分けることができる。一つ目の段階では、与えられた学習データを元に入力層から出力層の方向にそれぞれの人工ニューロンの出力を求める。二つ目の段階では、出力層における損失関数の勾配を求めた後、勾配を誤差として出力層から入力層の方向に伝えつつ連鎖律を使用した勾配の計算をそれぞれのパラメータについて行う。また、誤差逆伝播法と連鎖律についてはそれぞれ [12] の 5.3 節と [13] の 4.4 節に詳しい。

#### 3.2.4.3 バッチ処理

3.2.4.1 節で紹介した最急降下法では、パラメータの更新のたびに全ての学習データの期待値として誤差関数の値を計算する。このように全てのデータをひとまとめに扱う手法をバッチ処理と呼び、並列計算による高速化を行うことができる。しかし、パラメータの更新のたびに全データを扱うために、時間計算量と空間計算量のいずれも悪い手法となる。そこで、ミニバッチ処理と呼ばれる手法を用いるのが一般的である。ミニバッチ処理では、パラメータの更新のたびに指定したサイズ (バッチサイズ) のサブセットのみを用いる。これにより、時間計算量と空間計算量のいずれも改善しつつ並列計算による高速化を享受することができる。

### 3.3 CNN

CNN (Convolution Neural Network) は、全結合層だけでなく畳み込み層とプーリング層を用いた順伝播型のニューラルネットワークである。2012年のILSVRCで2位以下に10%以上の画像の認識精度の差をつけて優勝したAlexNet [14] はその著名な例である。また、本章では二次元データの画像でのCNNの説明を行うが、CNNについて詳しくは [15] を参考にされたい。

#### 3.3.1 CNNと視覚野の関係

CNNは脳の視覚野の機能を模倣するニューラルネットワークであり、Neocognitron [16] を起源に持つ。また、脳の視覚野には主に二つの機能がある。一つ目は刺激の位置により異なった位置のニューロンの活動電位を発生させることであり、二つ目は視覚情報の特徴を段階的に受容することである。

畳み込み層とプーリング層を用いることで二つの機能をCNNは表現する。まず、畳み込み層とプーリング層のいずれも局所領域間の位置関係を変えないため、前者の機能を模倣する。そして、畳み込み層では局所領域の特徴を抽出し、プーリング層では局所領域の特徴を集約するため、後者の機能を模倣する。

#### 3.3.2 CNNの畳み込み層

畳み込み層では、カーネルと呼ばれるフィルターを用いて入力の行列に畳み込み演算を行い、行列を出力する(図3.3)。カーネルは重みを成分に持つ行列であり、ストライドと呼ばれる一定の間隔を空けながら入力の行列の上を上下にスライドする。また、畳み込み演算ではカーネルにより覆われた入力の部分行列とカーネルとの間の内積を求める計算が行われる。そして、入力と出力の大きさを変えないために入力の周りに適当な幅で値を埋めるのが一般的であり、この操作をパディングと呼ぶ。

ここで、畳み込み層の入出力となる行列のことを特微量マップと呼び、特微量マップの枚数を表現する際には特微量マップをチャンネルと呼ぶ。また、図3.3の畳み込み層は入力と出力として1チャンネルずつ持つが、入力と出力のいずれも複数チャンネルを取ることも可能である。そして、入力が複数チャンネルの場合はそれぞれの入力から求まる特微量マップを成分ごとに足しあわせた結果が出力となり、出力が複数チャンネルの場合は1チャンネルの入力に対してカーネルが出力の数だけ必要となる。

#### 3.3.3 CNNのプーリング層

プーリング層では、入力を局所領域に分解した後にそれぞれの領域への操作により情報量の圧縮を行う(図3.4)。また、この操作で最大値をとる場合をMax Pooling、平均値をとる場合をAverage Poolingと呼び、主にこの二つがプーリング層では用いられる。

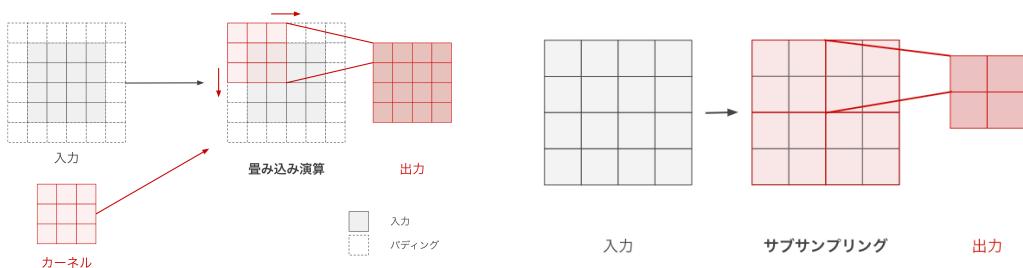


図3.3: 畠み込み層

図3.4: プーリング層

## 3.4 GAN

GAN (Generative Adversarial Networks) [17] はニューラルネットワークの応用例であり、学習データの特徴を持つ擬似的なデータを生成することを目指す手法である。また、この手法を用いた DCGAN [18] のように、実在しない人やホテルの画像などを生成することも可能である。

### 3.4.1 GAN の概要

GAN は Discriminator (識別モデル) と Generator (生成モデル) の二つのニューラルネットワークにより構成される(図 3.5)。まず、識別モデルはデータが Fake data (生成モデルの出力) と Real data (学習データ) のどちらであるかを識別することを目標に学習を進める。そして、生成モデルは識別モデルが学習データであると誤って識別するほど学習データに近いデータを出力することを目標に学習を進める。この二つのモデルが競合的に学習を進めてナッシュ均衡の状態を目指すことで、漸進的に生成モデルが学習データにより近いデータを生成できるようになると期待される。また、生成モデルの入力には適当な次元のベクトルをノイズとして与えるのが一般的であり、ノイズは生成モデルの出力の揺らぎを表現する潜在変数の役割を担う。

### 3.4.2 GAN の定式化

GAN では、生成モデルと識別モデルの目的関数をそれぞれ式 (3.6)、式 (3.7) として定式化する。また、学習データを  $\mathbf{x}$ 、生成モデルの出力を  $G(\mathbf{z}; \theta_G)$ 、ノイズを  $\mathbf{z}$ 、識別モデルの出力を  $D(\cdot; \theta_D)$ 、生成モデルと識別モデルのパラメータをそれぞれ  $\theta_G, \theta_D$ 、とする。

$$\arg \min_{\theta_G} \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}; \theta_G); \theta_D))] \quad (3.6)$$

$$\arg \max_{\theta_D} \mathbb{E}_{\mathbf{x}}[\log D(\mathbf{x}; \theta_D)] + \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}; \theta_G); \theta_D))] \quad (3.7)$$

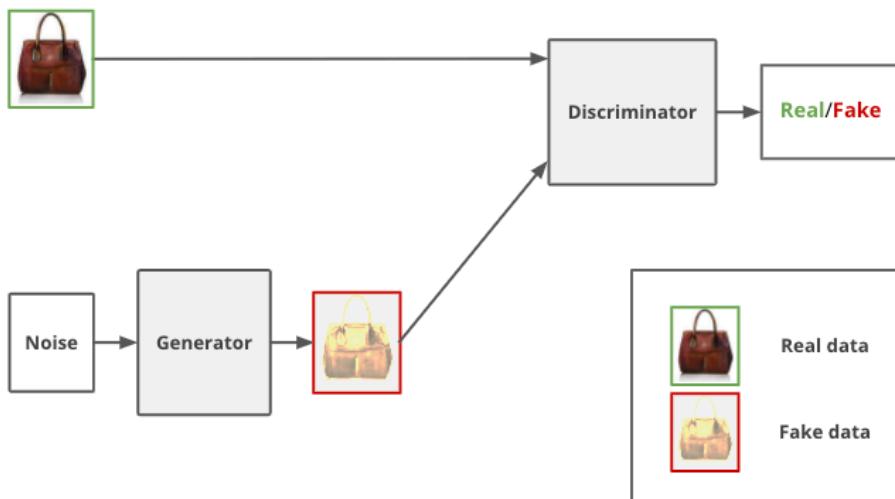


図 3.5: GAN の全体図

## 3.5 Pix2pix

Pix2pix [1] は生成モデルと識別モデルのいずれにも変換元の画像を条件として与えることで画像の変換を行う GAN である(図 3.7)。具体的には、線画から写真への変換(図 3.6a)や白黒画像からカラー画像への変換(図 3.6b)を行うことができる。

### 3.5.1 Pix2pix の定式化

Pix2pix では、生成モデルと識別モデルの目的関数をそれぞれ式 (3.8)、式 (3.9) として定式化する。また、目的関数は GAN とほとんど同じであるが、変換元の学習データ  $y$  を条件として与える点と生成モデルの目的関数に変換先の学習データと生成モデルの出力の差分の L1 ノルムを含める点が異なる。

$$\arg \min_{\theta_G} \mathbb{E}_{y,z} [\log(1 - D(y, G(y, z; \theta_G); \theta_D))] + \lambda \mathbb{E}_{x,y,z} [\|x - G(y, z; \theta_G)\|_1] \quad (3.8)$$

$$\arg \max_{\theta_D} \mathbb{E}_{x,y} [\log D(y, x; \theta_D)] + \mathbb{E}_{y,z} [\log(1 - D(y, G(y, z; \theta_G); \theta_D))] \quad (3.9)$$



図 3.6: Pix2pix のスタイル変換の例

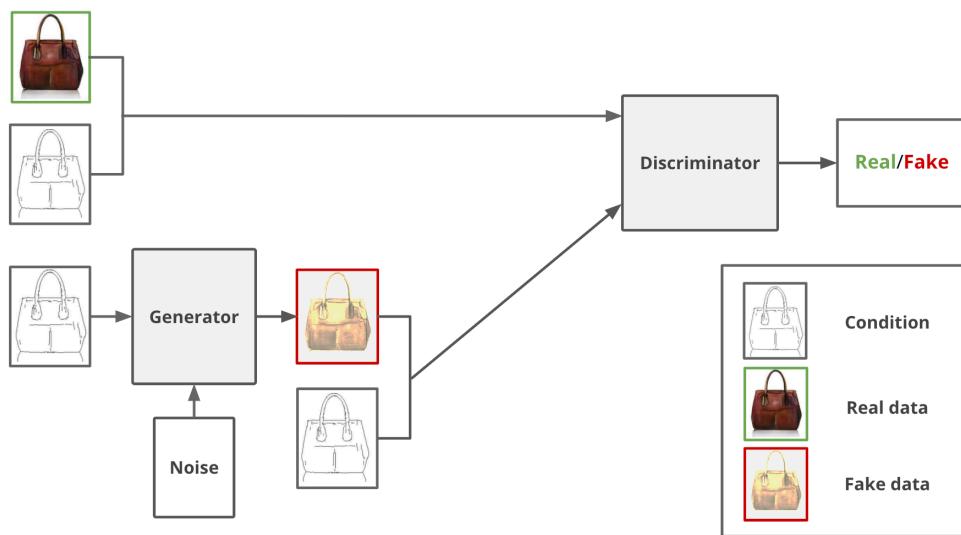


図 3.7: Pix2pix の全体図

### 3.5.2 Pix2pix の生成モデル

Pix2pix の生成モデルには、スキップコネクションを持った Encoder-Decoder を用いる（図 3.8a）。また、Dropout 層により GAN のノイズを表現する。ここで、Dropout [19] とは学習時にペルヌーイ分布に従ってランダムにある層の重みの一部を 0 として無視する正則化の手法である。

#### 3.5.2.1 Encoder-Decoder

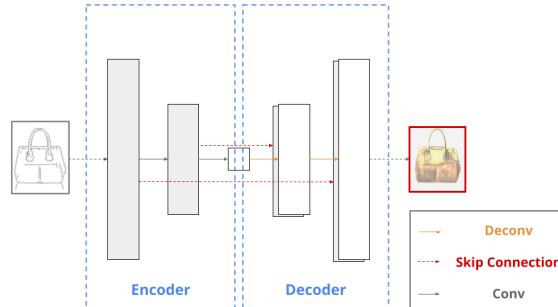
Encoder-Decoder は、画像から特微量を抽出する処理 (Encode) と特微量を元に画像を再構成する処理 (Decode) の二段階で構成されたモデルである。また、Pix2pix では、Encode の部分に畳み込み層を用い、Decode の部分に逆畳み込み層を用いる。逆畳み込み層は畳み込み層の逆演算に相当する演算を行う層であり、[20] の 3.2.2 節に詳しい。

#### 3.5.2.2 スキップコネクション

Pix2pix の生成モデルは図 3.8a で表現されるスキップコネクションを持つ。このスキップコネクションは U-net [21] 由来のものであり、変換前後の画像でのピクセルの対応関係を維持する役割を担う。

### 3.5.3 Pix2pix の識別モデル

Pix2pix の識別モデルでは、CNN を用いてパッチと呼ばれる小領域ごとに真偽を求め、その平均を出力とする（図 3.8b）。また、Pix2pix ではこの識別モデルを PatchGAN と名付けた。



(a) 生成モデルのネットワーク



(b) GAN と PatchGAN の識別モデルの比較

図 3.8

# 第 4 章 実験

本章では、提案手法と実験方法の説明を行った後、実験結果の考察を行う。

## 4.1 提案手法

本節では、提案手法の説明を行う。

### 4.1.1 提案手法の目標

提案手法の目標は、Pix2pix を参考に作成した提案モデルを用いて音の高さと大きさを維持したままギターの単音からハープの単音への音色の変換を行うことである。ここで、ある楽器を用いてある高さの一音を鳴らした時に出力される音として単音を定義する。また、音色の変換対象にギターとハープを選んだ理由は、弦楽器という共通点を持ちながらも音波の観察と音の聴き取りの評価により十分に異なると判定できると考えたからである。

### 4.1.2 提案モデルの構造

提案モデルの GAN には生成モデルと識別モデルのいずれにも変換元のギターの音を条件として与える(図 4.1)。基本構造は Pix2pix と変わらないが、決定論的に音を生成するためにノイズを表す Dropout 層を使用していない点に注意が必要である。

また、44100 Hz のサンプリング周波数でサンプリングした 1 秒の音響信号を音のデータとして使用する。ここで、量子化ビット数が 16 ビットでチャンネル数が 1 であるため、この音のデータは 44100 の長さを持つ 16 ビット整数の一次元配列である。

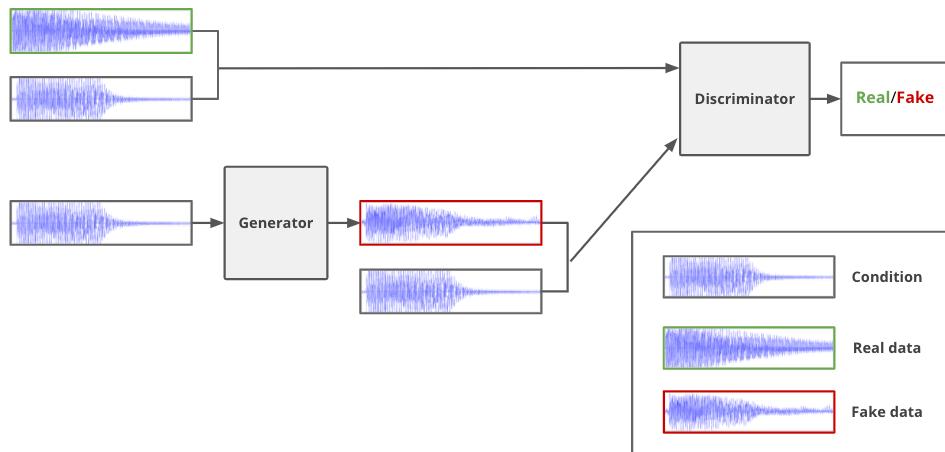
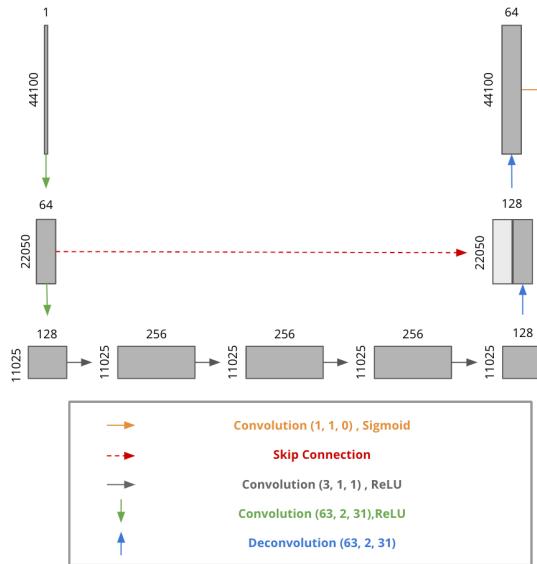


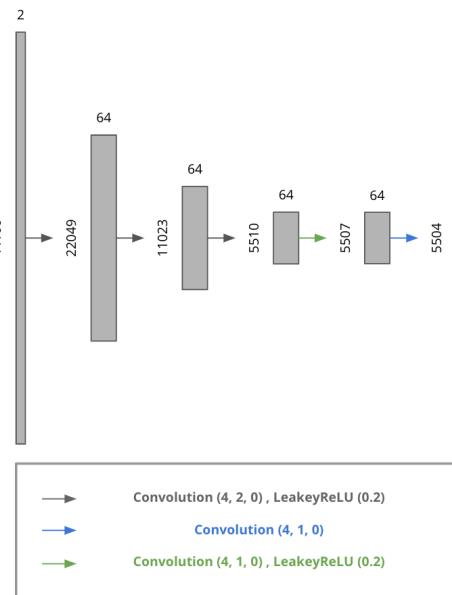
図 4.1: 提案モデルの全体図

### 4.1.3 生成モデルと識別モデル

提案モデルの生成モデルと識別モデルを図 4.2 に示す。まず、灰色の箱は特微量マップを表す。箱の上側と左側にそれぞれチャンネル数と特微量マップとなる一次元配列の長さを示す。そして、矢印はニューラルネットワークの層の操作を表す。畠み込み層と逆畠み込み層では (カーネルサイズ、パディング数、ストライド) としてそれぞれの値を示し、活性化関数の LeakyReLU では負の実数の定義域での一次関数の傾きの値を示す。また、スキップコネクションでは Encode 時の特微量マップを Decode 時にも利用することで実装している。



(a) 生成モデルのネットワーク



(b) 識別モデルのネットワーク

図 4.2: 本研究の生成モデルと識別モデル

## 4.2 実験方法

本節では実験方法の説明を行う。また、実験時のパラメータについては付録 B に示す。

### 4.2.1 データセット

データセットとして 1 秒のギターとハープの音を 88 組用いた。また、この 88 音は音の大きさが等しいものと仮定した。そして、音の高さとしては A0～C8 の 88 音の半音を選んだ。これらは一般的な 88 鍵のピアノのそれぞれの鍵盤の音に対応する（図 4.3）。

#### 4.2.1.1 音階表記

本実験では西洋音楽の 12 音階表記を音階表記に用いる。この表記では、 $C, C^\sharp, D, D^\sharp, E, F, F^\sharp, G, G^\sharp, A, A^\sharp, B$  の 12 段階の音の高さの集合をオクターブとして定める。そして、それぞれのオクターブに番号を振り、440 Hz の音を A4 と定めることで音の高さの絶対的な表記を可能にしている。

### 4.2.2 実験 1：提案モデルの表現力の評価実験

88 音を学習データと評価データのいずれでも使用し、同じ高さの音の間での変換をすることで、提案モデルの表現力を評価する実験を行った。

### 4.2.3 実験 2：提案モデルの汎化能力の評価実験

88 音のうち  $3/4$  を学習データ、 $1/4$  を評価データとする 4 分割交差検定をすることで、提案モデルの汎化能力を評価する実験を行った。また、この際のデータセットの分割方法を付録 C に示す。

### 4.2.4 データ拡張

88 音と小さいサイズのデータセットへの過学習を防ぐため、振幅方向でのデータ拡張を行った。具体的には、各エポックの任意の学習データの振幅に一様乱数に従って  $0.3 \sim 1.0$  の乱数をかけた。

### 4.2.5 評価方法

4.1.1 節の提案手法の目標を満たすかという観点から音波の観察と音の聴き取りにより評価を行った。また、データセットで組となる音の高さ（基音）は等しいが、基音より高い音（上音）の成分の組み合わせが異なるために音色の違いが生まれる。したがって、4.3 節では上音に注目した考察を中心に行う。

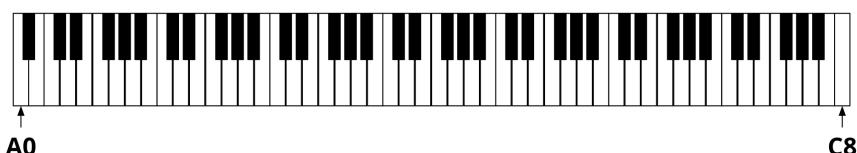


図 4.3: 88 鍵のピアノの鍵盤

## 4.3 実験結果

実験結果の考察を本節では行う。また、実験結果として記載する波形の図は三つの波形を上から並べている。これらは上から順に、変換元のギターの波形、生成モデルの出力の波形、変換先のハープの波形、である。そして、本節には一部の音波のみを記載し、付録 D に実験結果の音波を全て記載する。

### 4.3.1 実験 1：提案モデルの表現力の評価実験

提案モデルの表現力の評価実験を行ったところ、実験結果は二つに大別された。

#### 4.3.1.1 ハープの音を表現できた場合

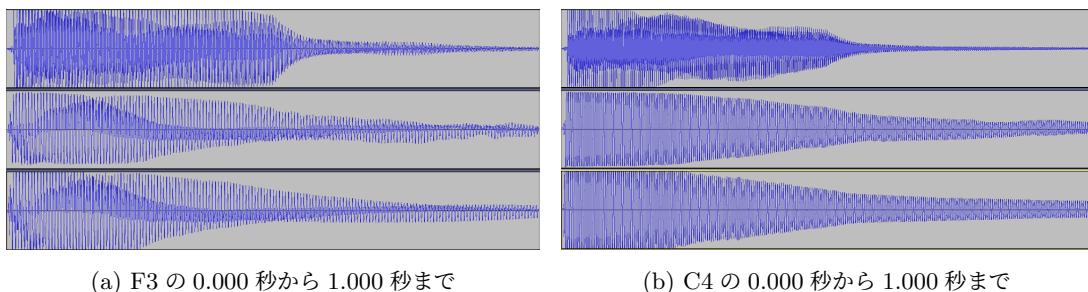
88 音のうち 86 音ではハープの音を表現できた（図 4.4a）。また、特に C4 から D5♯ の 16 音では目標のハープの音に極めて近い音色の音を生成できた（図 4.4b）。これらの音は上音が少ないため、他の音と比べて表現が容易であったと考えられる。

#### 4.3.1.2 ハープの音を表現できなかった場合

D7♯ と E7 の 2 音では音の高さを維持できたが、ハープの音を十分に表現できなかった（図 4.5）。いずれの音でも学習データである変換先のハープの音の振動が安定しておらず、この不安定さを学習できなかった。また、高音では不安定な振動の学習データが多く、高音での安定したデータセットの作成が難しいことがわかった。

### 4.3.2 実験 2：提案モデルの汎化能力の評価実験

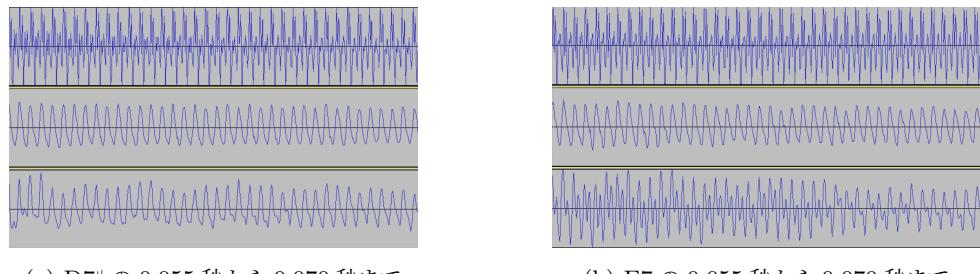
提案モデルの汎化能力の評価実験を行ったところ、実験結果は三つに大別された。



(a) F3 の 0.000 秒から 1.000 秒まで

(b) C4 の 0.000 秒から 1.000 秒まで

図 4.4



(a) D7♯ の 0.055 秒から 0.070 秒まで

(b) E7 の 0.055 秒から 0.070 秒まで

図 4.5

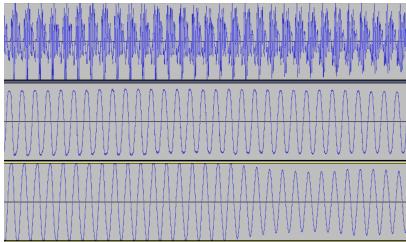


図 4.6: D4♯ の 0.100 秒から 0.200 秒まで

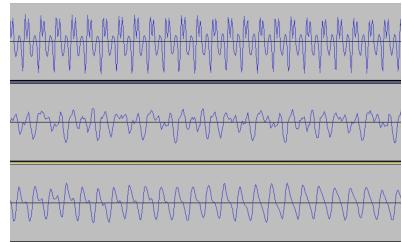


図 4.7: D7♯ の 0.1700 秒から 0.110 秒まで

#### 4.3.2.1 ハープの音を表現できた場合

D4,D4♯,G4,F5,F5♯ の 5 音ではハープの音を表現できた (図 4.6)。これらの高さの音では提案モデルの表現力の評価実験の際にも特にハープの音色に近い音を生成でき、上音が少ないほど表現が容易であるという推察が強められた。

#### 4.3.2.2 ハープの音を表現できず音の高さも維持できなかった場合

C7,D7♯,E7,F7♯,G7,G7♯,A7,B7,C8 の 9 音では音の高さも維持できず、騒音が生成された (図 4.7)。これらの高さの音では提案モデルの表現力の評価実験の際にもハープの音を表現できず、高音域における安定したデータセットの作成が難しいという推察が強められた。

#### 4.3.2.3 ハープの音を表現できず音の高さを維持できた場合

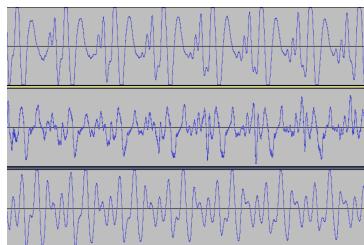
14 音を除く 74 音では音の高さは維持できたがハープの音を表現できなかった (図 4.8)。これらの高さの音では生成された音波が安定した振動をせず上音の成分がハープよりも多い音波が多く観測された。また、これらの高さの音では提案モデルの表現力の評価実験の際にはハープの音色の音波を表現できていたため、提案モデルの汎化能力の低さが明らかになった。

### 4.3.3 課題

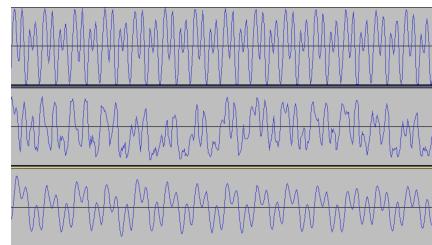
提案モデルの表現力の評価実験の実験結果からさらに四つの課題が明らかになった。

#### 4.3.3.1 音の減衰の表現

振動の減衰を表現できていない音がいくつかあった。これらの音は、E2 以下の低音域ではハープとは全く異なる波形で減衰し (図 4.9a)、A6 以上の高音域ではほとんど振動が見られなかった (図 4.9b)。また、提案モデルの表現力不足で微小な振動の表現が難しいことが原因として考えられる。

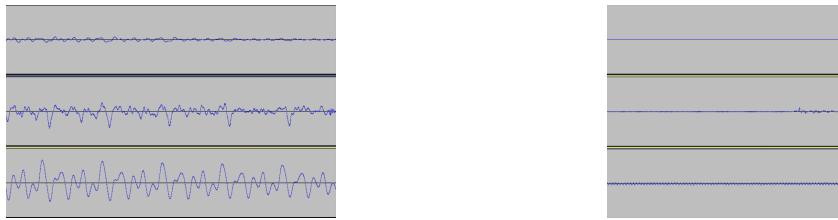


(a) D1 の 0.300 秒から 0.500 秒まで



(b) F6 の 0.070 秒から 0.080 秒まで

図 4.8



(a) A0 の 0.800 秒から 1.000 秒まで

(b) B7 の 0.980 秒から 1.000 秒まで

図 4.9

#### 4.3.3.2 音の大きさの維持

部分的に大きさを維持できていない音がいくつかあった。特に波形の前半で生成波形の振幅が小さくなる様子が多く見られた(図 4.10)。振幅の無作為化が原因であると考えられ、学習の初段階での振幅の固定や振幅の大きさの別のネットワークによる調整などの工夫が必要であると考えられる。

#### 4.3.3.3 音波の滑らかさの表現

ノイズまじりの音がいくつかあった。これらの音では音波の滑らかさを表現できていないことがわかった(図 4.11)。音波を滑らかにするような加工を生成後に加えるなどの工夫が必要であると考えられるが、音波の滑らかさを表現できないという課題は他の音の生成の研究でも見られる。

#### 4.3.3.4 データセットの不安定さ

ここまで三つは提案モデルの改善により解消されると考えられるが、問題のあるデータセットが一部に見られた。一つの問題点は高音において振動が不安定になることであり、4.3.1.2 節にて述べた。そして、音の鳴り出での振動が不安定であるという問題点も見られた。具体的には、ギターの音の鳴り出しの遅延の方がハープより大きい場合(図 4.12a)や周期的な音になるまでに遅延があるために不規則な振動となる場合(図 4.12b)などがあった。

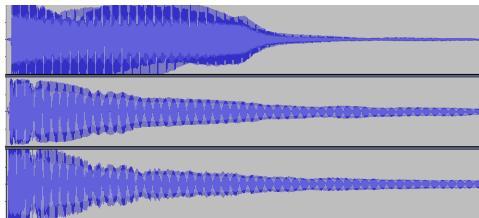


図 4.10: C5 の 0.000 秒から 1.000 秒まで

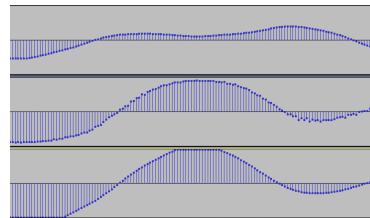
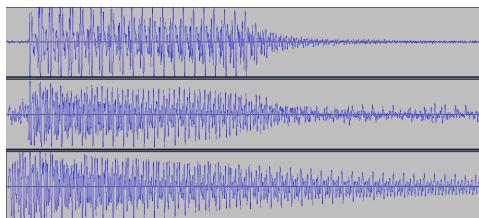
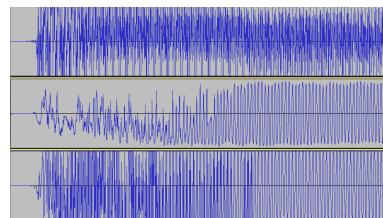


図 4.11: D2♯ の 0.100 秒から 0.103 秒まで



(a) F1♯ の 0.000 秒から 1.000 秒まで



(b) A7 の 0.000 秒から 0.030 秒まで

図 4.12

# 第5章 まとめ

本章では、実験結果の考察と展望についてまとめる。

## 5.1 実験結果のまとめ

提案手法の目標は音の高さと大きさを維持したままでギターの単音からハープの単音への音色の変換を行うことである。実験の結果、この目標に足る表現力を持つ提案モデルを作成できたが、作成した提案モデルの汎化能力は十分でないことも確認できた。

汎化能力の低い理由としては主に三点あると考えられる。一点目は、Pix2pix で用いられる Dropout 層を用いていない点である。Dropout 層は GAN のノイズを表すだけでなく、汎化能力を高める効果があると期待される。二点目は、データ拡張の工夫が振幅の無作為化のみである点であり、時間方向にずらすなどの工夫もすべきであった。三点目は、音の表現として音響信号を用いた点であり、時間-周波数表現を用いたモデルを作成して性能比較をする必要があったと考えられる。

さらに、本研究では波形の観察を中心とした考察を行ったが、本研究の考察を明確化するためにはより定量的な判定が必要であったと考えられる。具体的には、音の高さの維持、音の大きさの維持、音色の変換、という三点についての判定である。

## 5.2 展望

以下の三点を解決することで本研究の提案手法を音楽での音色の変換に適用できると考えられる。

### 5.2.1 楽器の重ね合わせ

音楽はそれぞれの楽器から出力される音の重ね合わせになっている。楽器ごとに音色は異なるため、音色の変換を行うには楽器ごとの音波に分解すること（音源分離）が必要である。また、楽曲の作成時に楽器ごとに分離したデータ（パラデータ）の公開が一般的になれば音源分離の必要はなくなる。

### 5.2.2 音の重ね合わせ

ある音が単音の重ね合わせである重音の場合は単音ごとに音色の変換を行う必要がある。この時、単音だけでなく重音もデータセットに加えることで音色の変換が可能であると考えられる。しかし、この際にデータセットが膨大な量になる可能性があり、追加するデータセットの工夫が必要となる。

### 5.2.3 音の繋ぎ方

上記の二つが可能である時、時間方向の音の繋ぎ方を工夫する必要がある。単位時間ごとに区切るのみで音色変換は可能であると考えていたが、音の連続的な音色の変化に対応するためには二次元の表現を用いるべきであると考えられる。

# 謝辞

本研究にてご指導を賜った指導教官の金子知適准教授に厚く感謝いたします。また、中屋敷太一さんをはじめとする金子研の構成員の方々には多くの助言をいただきました。ありがとうございました。

# 参考文献

- [1] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125–1134, 2017.
- [2] Keunwoo Choi, György Fazekas, Kyunghyun Cho, and Mark Sandler. A tutorial on deep learning for music information retrieval. *arXiv preprint arXiv:1709.04396*, 2017.
- [3] Sicong Huang, Qiyang Li, Cem Anil, Xuchan Bao, Sageev Oore, and Roger B Grosse. Timbretron: A wavenet (cyclegan (cqt (audio))) pipeline for musical timbre transfer. *arXiv preprint arXiv:1811.09620*, 2018.
- [4] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.
- [5] Stanley Smith Stevens, John Volkmann, and Edwin Broomell Newman. A scale for the measurement of the psychological magnitude pitch. *The journal of the acoustical society of america*, Vol. 8, No. 3, pp. 185–190, 1937.
- [6] Stanley S Stevens and John Volkmann. The relation of pitch to frequency: A revised scale. *The American Journal of Psychology*, Vol. 53, No. 3, pp. 329–353, 1940.
- [7] Judith C Brown. Calculation of a constant q spectral transform. *The Journal of the Acoustical Society of America*, Vol. 89, No. 1, pp. 425–434, 1991.
- [8] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan. Neural audio synthesis of musical notes with wavenet autoencoders. In *International Conference on Machine Learning*, pp. 1068–1077. PMLR, 2017.
- [9] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaaan Gulrajani, Chris Donahue, and Adam Roberts. Gansynth: Adversarial neural audio synthesis. *arXiv preprint arXiv:1902.08710*, 2019.
- [10] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.
- [11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [12] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.
- [13] James J Callahan. *Advanced calculus: a geometric view*. Springer Science & Business Media, 2010.
- [14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, Vol. 25, pp. 1097–1105, 2012.
- [15] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Liyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern Recognition*, Vol. 77, pp. 354–377, 2018.
- [16] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model

- for a mechanism of pattern recognition. pp. 193–202. Springer, 1980.
- [17] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.
  - [18] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
  - [19] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, Vol. 15, No. 1, pp. 1929–1958, 2014.
  - [20] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pp. 1520–1528, 2015.
  - [21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

# 付録 A Adam

Adam は勾配降下法のアルゴリズムの一つであり、図 A.1 に示す手順にしたがってパラメータの更新を行う。また、Adam については [11] に詳しい。

```
HyperParameter :  $\beta_1, \beta_2 \in (0, 1], \eta, \epsilon$ 
Objective Function :  $f$ 
Parameter :  $\theta$ 

Initialization:
     $t \leftarrow 0$                                 /* Initialize timestep */
     $\theta \leftarrow \theta_0$                          /* Initialize Parameter */
     $m_1 \leftarrow 0$                                 /* Initialize 1st moment */
     $m_2 \leftarrow 0$                                 /* Initialize 2nd moment */

end

Procedure:
while  $\theta$  not converged do
     $t \leftarrow t + 1$                                 /* Update timestep */
     $g \leftarrow \nabla_{\theta} f(\theta)$                   /* Compute gradient of  $f(\theta)$  */
     $m_1 \leftarrow \beta_1 \cdot m_1 + (1 - \beta_1) \cdot g$  /* Update biased 1st moment */
     $m_2 \leftarrow \beta_2 \cdot m_2 + (1 - \beta_2) \cdot (g \odot g)$  /* Update biased 2nd moment */
     $\hat{m}_1 \leftarrow m_1 / (1 - \beta_1^t)$            /* Compute bias-corrected 1st moment */
     $\hat{m}_2 \leftarrow m_2 / (1 - \beta_2^t)$            /* Compute bias-corrected 2nd moment */
     $\theta \leftarrow \theta - \eta \cdot \hat{m}_1 / (\sqrt{\hat{m}_2} + \epsilon)$  /* Update Parameter */

end
return  $\theta$ 
end
```

図 A.1: Adam の疑似コード

## 付録 B 学習時のパラメータ

表 B.1 に提案モデルの学習時のパラメータを示し、表 B.2 に Adam のハイパーパラメータを示す。

パラメータ	値
バッチサイズ	1
エポック数	1000

表 B.1

パラメータ	値
$\beta_1$	0.5
$\beta_2$	0.999
$\eta$	0.0002
$\epsilon$	$10^{-8}$

表 B.2

## 付録 C データセットの分割

22 音ずつの 4 つのサブセットにデータセットを分割した (図 C.1)。また、データセットの 4 分割は、88 音をシャッフルして配列に格納した後に 22 音ずつ順に選ぶことで行った。

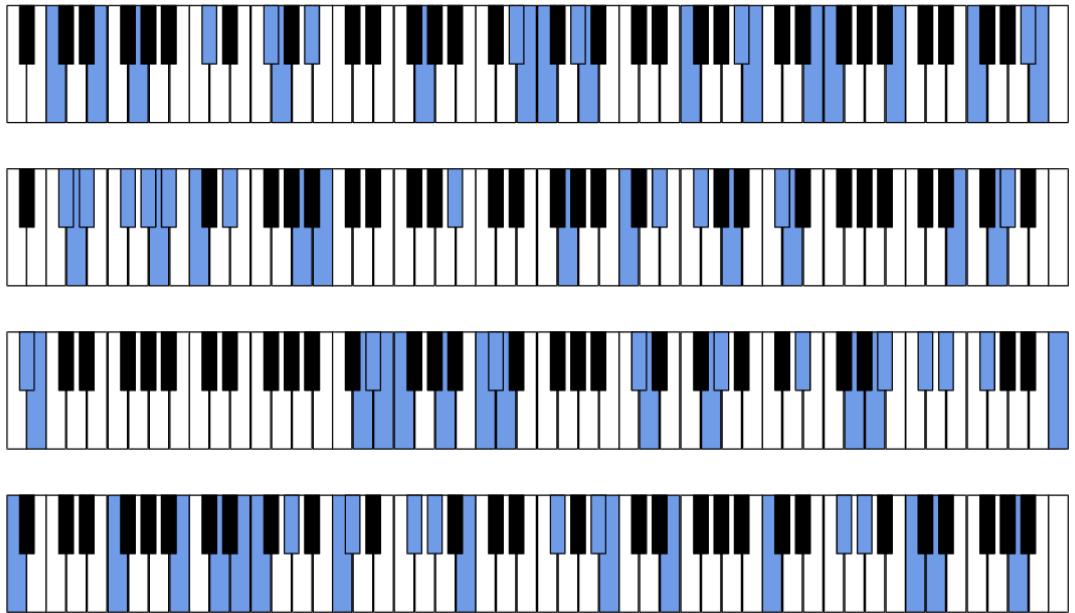


図 C.1: データセットのサブセット

# 付録 D 実験結果

4.3 節に載せることのできなかった波形の図を本章に載せる。また、波形の記載方法は 4.3 節と同様であるが、簡略化のため、キャプションには音階名のみ載せている。

## D.1 提案モデルの表現力の評価実験

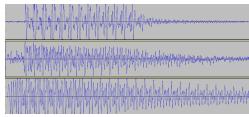


図 D.1: A0

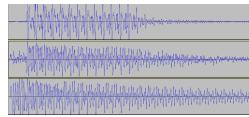


図 D.2: A0#

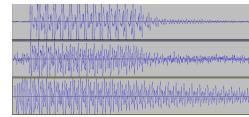


図 D.3: B0

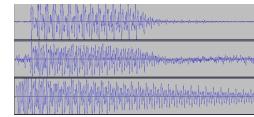


図 D.4: C1

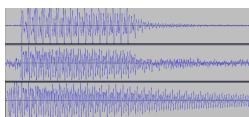


図 D.5: C1#

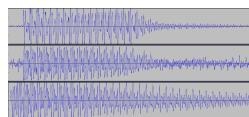


図 D.6: D1

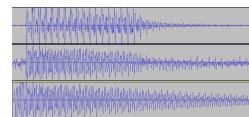


図 D.7: D1#

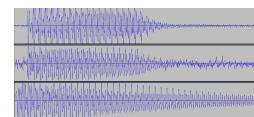


図 D.8: E1

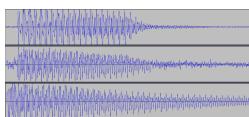


図 D.9: F1

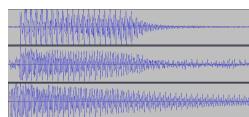


図 D.10: F1#

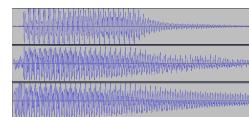


図 D.11: G1

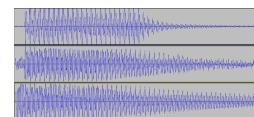


図 D.12: G1#

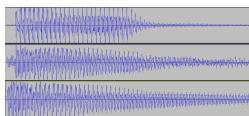


図 D.13: A1

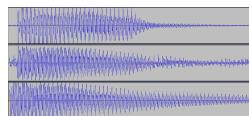


図 D.14: A1#

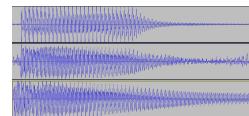


図 D.15: B1

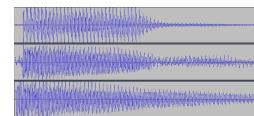


図 D.16: C2

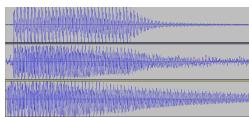


図 D.17: C2#

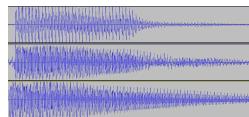


図 D.18: D2

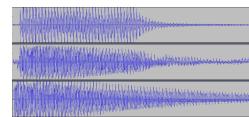


図 D.19: D2#

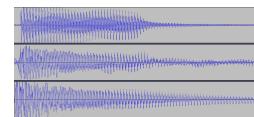


図 D.20: E2

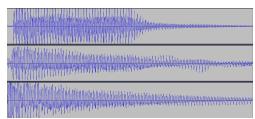


図 D.21: F2

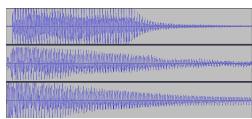


図 D.22: F2♯

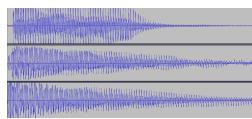


図 D.23: G2

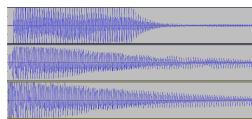


図 D.24: G2♯

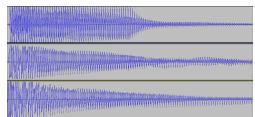


図 D.25: A2

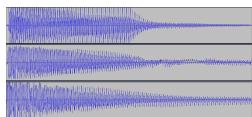


図 D.26: A2♯

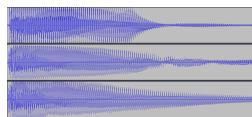


図 D.27: B2

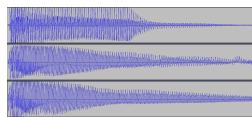


図 D.28: C3

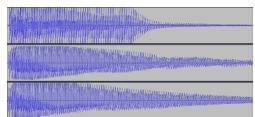


図 D.29: C3♯

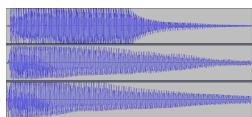


図 D.30: D3

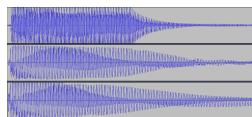


図 D.31: D3♯

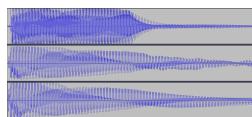


図 D.32: E3

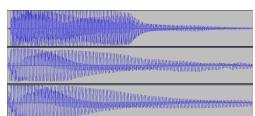


図 D.33: F3

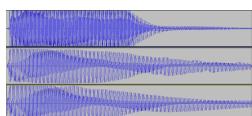


図 D.34: F3♯

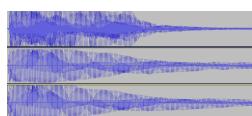


図 D.35: G3

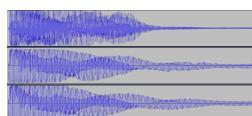


図 D.36: G3♯

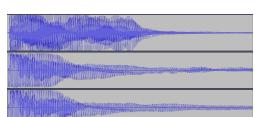


図 D.37: A3

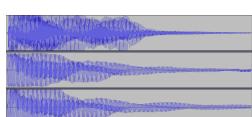


図 D.38: A3♯



図 D.39: B3

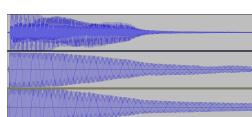


図 D.40: C4

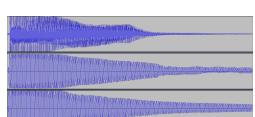


図 D.41: C4♯

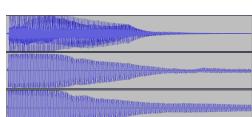


図 D.42: D4

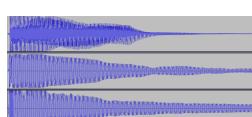


図 D.43: D4♯

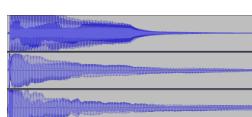


図 D.44: E4

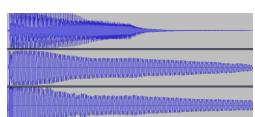


図 D.45: F4

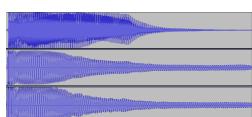


図 D.46: F4♯

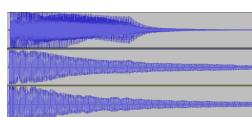


図 D.47: G4

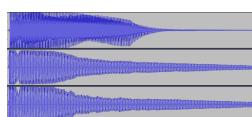


図 D.48: G4♯

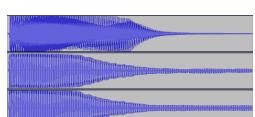


図 D.49: A4

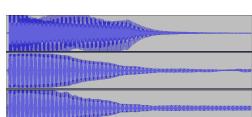


図 D.50: A4♯



図 D.51: B4

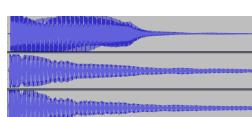


図 D.52: C5

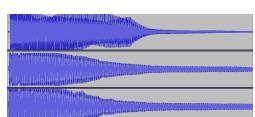


図 D.53: C5♯

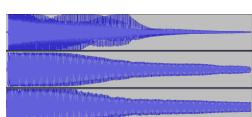


図 D.54: D5

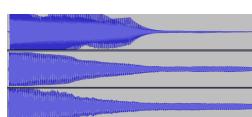


図 D.55: D5♯

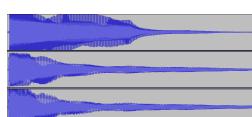


図 D.56: E5

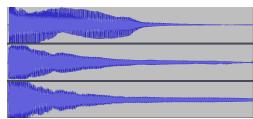


図 D.57: F5

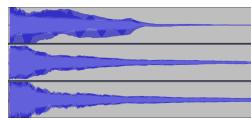


図 D.58: F5♯

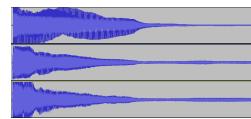


図 D.59: G5

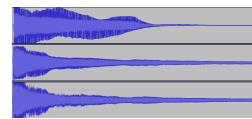


図 D.60: G5♯

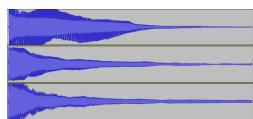


図 D.61: A5

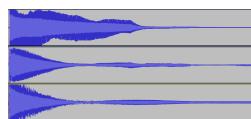


図 D.62: A5♯



図 D.63: B5

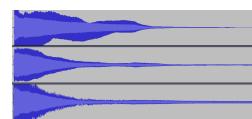


図 D.64: C6



図 D.65: C6♯



図 D.66: D6



図 D.67: D6♯

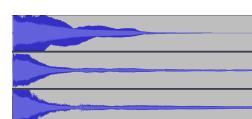


図 D.68: E6

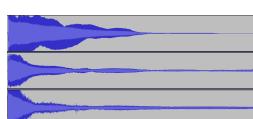


図 D.69: F6

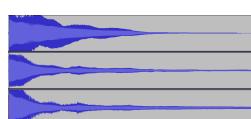


図 D.70: F6♯



図 D.71: G6

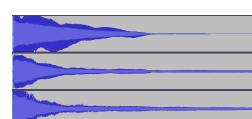


図 D.72: G6♯

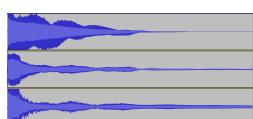


図 D.73: A6



図 D.74: A6♯

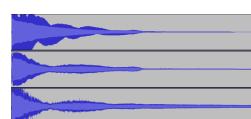


図 D.75: B6



図 D.76: C7

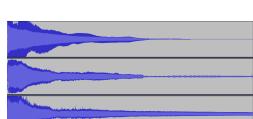


図 D.77: C7♯

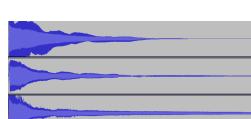


図 D.78: D7



図 D.79: D7♯

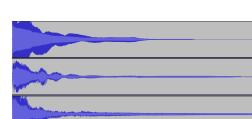


図 D.80: E7

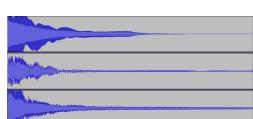


図 D.81: F7

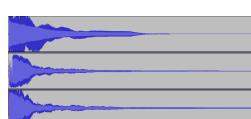


図 D.82: F7♯

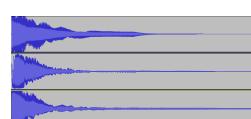


図 D.83: G7

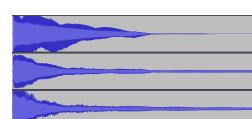


図 D.84: G7♯



図 D.85: A7



図 D.86: A7♯



図 D.87: B7

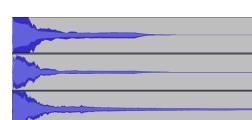


図 D.88: C8





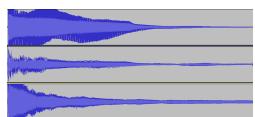


図 D.149: A5

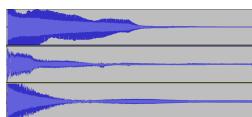


図 D.150: A5♯

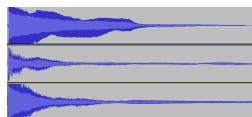


図 D.151: B5

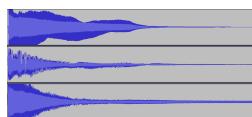


図 D.152: C6

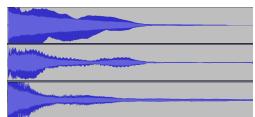


図 D.153: C6♯

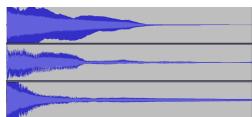


図 D.154: D6

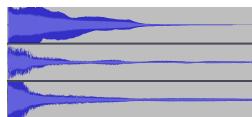


図 D.155: D6♯

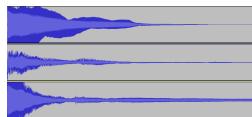


図 D.156: E6

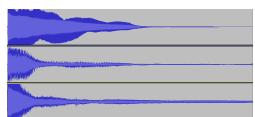


図 D.157: F6

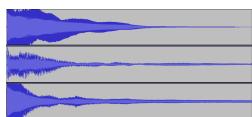


図 D.158: F6♯

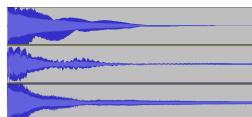


図 D.159: G6

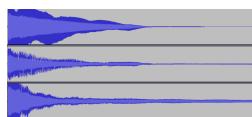


図 D.160: G6♯

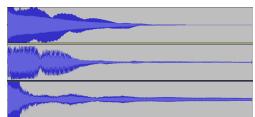


図 D.161: A6



図 D.162: A6♯

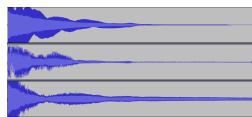


図 D.163: B6

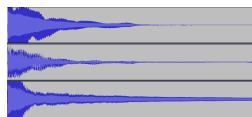


図 D.164: C7

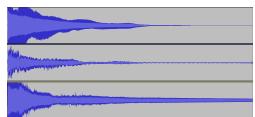


図 D.165: C7♯

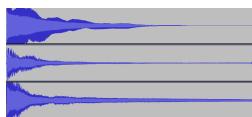


図 D.166: D7



図 D.167: D7♯



図 D.168: E7



図 D.169: F7



図 D.170: F7♯

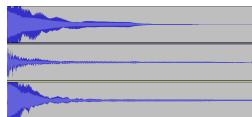


図 D.171: G7



図 D.172: G7♯



図 D.173: A7



図 D.174: A7♯



図 D.175: B7

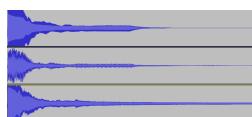


図 D.176: C8