公立はこだて未来大学 2017 年度 システム情報実習

グループ報告書

Future University Hakodate 2017 System Information Science Practice
Group Report

プロジェクト名

フラクタル×ジャズ

Project Name

Fractal × Jazz

プロジェクト番号/Project No.

4

プロジェクトリーダ/Project Leader

1015238　吉田周平　Shuhei Yodhida

グループ名

フラクタルグループ　　ジャズグループ

Group Name

Fracral Group　　JazzGroup

グループリーダ/Group Leader

〈フラクタルグループ〉

1015238　吉田周平　Shuhei Yoshida

〈ジャズグループ〉

1015037　竹内舜二　Shunji Takeuchi

グループメンバ/Group Member

〈フラクタルグループ〉

1015025　居附未紗　Misa Itsuki

1015265　近藤渚紗　Nagisa Kondo

〈ジャズグループ〉

1014078　中川哲哉　Tetsuya Nakagawa

1015049　北田明日香　Asuka Kitada

1015090　伊藤祐大　Yudai Ito

1015186　佐藤悠介　Yusuke Sato

1015191　高橋大介　Daisuke Takahashi

1015234　中島凱斗　Kaito Nakajima

担当教員

川越敏司　川口聡　斉藤朝輝

Advisor

Toshiji Kawagoe　Satoshi Kawaguchi　Asaki Saito

提出日

2018 年 1 月 19 日

Date of Submission

Janualy 19, 2018

概要

本プロジェクトは、「プロのジャズミュージシャンの演奏を模倣するアドリブ自動生成システムの作成」を通年の目標とする。コンピュータがプロの演奏を模倣し、かつメロディーは自動で生成するといった研究は少なく、音楽の自動生成という研究分野に貢献できると考える。目標を達成するため、様々なアプローチをおこなった。

まず音楽理論、とりわけジャズについての知識を学習した。このときジャズの、特にブルースにおいて使われる、ブルーノートスケールという伴奏に関わらずメロディーに使うことができるスケールに着目した。次に音楽の多くが従っていると言われている 1/f ゆらぎという性質に着目した。ガードナー [16] を用いて、フラクタルの性質をもつ比例雑音の白色雑音、褐色雑音、1/f ゆらぎについて、それぞれの特徴を有する乱数生成方法について学んだ。その後、メジャースケールとブルーノートスケール、比例雑音三種それぞれの特徴を有するアドリブの自動生成システムを合計 6 種類作成した。また、これらを用いて作成された曲を評価し、統計検定を用いて、各種スケールや雑音の優位性を調べた。このとき、どのスケール、どの雑音についても優位性は示されなかった。

そこで別のアプローチとして、ニューラルネットを用いた機械学習によるアドリブ生成システム、リカレントニューラルネットを用いた機械学習によるアドリブ生成システムの作成を目指した。そのためにまず、斎藤 [20] や、巣籠 [23] を用いて、ニューラルネット、リカレントニューラルネット、機械学習についての理解を深めた。その後、二種の機械学習を用いて、ある音の次に来る音の確率を出力する作曲システムを作成した。

模倣するプロのジャズミュージシャンはマイルス・デイヴィスとし、比較の対象はチャーリー・パーカーとした。システムによる学習をおこなうにあたり、両者のジャズ・アドリブの教師データとして、合計 110 曲を MIDI データ化した。

キーワード　音楽, ジャズ, 機械学習, ニューラルネットワーク

（※文責: 伊藤祐大）

Abstract

In this project, we decided that "the making of the system to generate ad-lib like a professional jazz musician" is our goal through this year. There are few studies that a computer is copied from a professional performance and the melody automatically generates and thinks that I can contribute to the research field called the automatic generation of the music. In the first semester, in particular, we aimed for imitating take1 of "Bags' Groove" by Miles Davis. We performed various approaches to achieve a goal.
At first, we learned music theory about the jazz among other things. We paid our attention to the blue note scale that we could use for a melody regardless of the accompaniment in jazz particularly the blues then. Next, we paid our attention to the property called 1/f nod that most music obeyed. We used Gardner[16] and learned about the white noise, brown noise, 1/f nod called the proportion noise that has a property of the fractal to learned a random number generation method to have each characteristic. After that, we made an ad-lib automatic generation system having a major scale and a blue note scale, each three kinds of proportion noises characteristic and six kinds in total. In addition, we evaluated these music and checked the superiority of various scales and noises using statistics official approval. The superiority was not shown about scale and noise then.
Therefore, we aimed at the making of the ad-lib generation system by the machine learning using the neural network, the recurrent neural network as different approach. We used Saito[20], Sugomori[23] and studied about the neural network, the recurrent neural network and the machine learning. After that, we made the composition system which output the probability of the forthcoming sound next to a certain sound using two kinds of machine learning.
The systems imitate Miles Davis who is professional jazz musician, and the subject of comparison is Charlie Parker. We converted 110 songs to MIDI data as teacher data of both jazz ad-libs for learning by the system.

Keyword Music, Jazz, Machine Learning, Neural Network

（※文責: 伊藤祐大）

目次

第 1 章　背景

この章では、まず音楽の自動生成の歴史や先行研究を紹介する。また、それらと比較して、本プロジェクトの目指すところを明らかにする。そのうえで課題となる取り組みについても説明する。

（※文責: 伊藤祐大）

1.1　音楽自動生成の歴史

そもそも音楽とは何だろうか。第六版広辞苑では「音による芸術」とある。西洋音楽では、一般にリズム、メロディー、ハーモニーという三要素をもつものとされる。この問いについては音楽ジャンルや文化などにより、長くに渡って様々な定義が存在し、音楽とは何かを一言で定義することは難しい。

また、人の音楽行為について、現代では一般的に作曲、演奏、鑑賞が基本とされる。作曲では作曲者が感情を音によって表現する。演奏では演奏者がそれを読みとったり原曲を変化させたりし、演奏で表現する。そして鑑賞ではそれを聴いて味わったり、価値を見極めたりする。このような音楽行為について、音楽をたしなむ人々の多くは鑑賞を主に行ってきたことから、作曲と演奏についてこれを自動化しようという試みが行われてきた。それが自動作曲、自動演奏である。

音楽の自動演奏について、身近でわかりやすい例ではオルゴールがある。簡単な構造のものでは、ぜんまいなどの動力がピンのついたシリンダーを回し、それが専用の楽器を弾いて決められた音を鳴らしていく仕組みになっている。蓄音機などの録音技術の発達以降は、インテリアや工芸製品としての価値に重きが置かれた。さらにコンピュータの発達以降は、MIDI などの楽譜やピアノロールをデータに置き換えたものからコンピュータが自動的に楽曲を演奏する方向に発達した。近年では、スタインウェイの「SPIRIO」(http://www.steinway.co.jp/spirio) のように、プロの演奏に限りなく近い演奏を再現させる自動演奏製品もある。

音楽の自動作曲について、古くは「さいころのミサ」と呼ばれる、ジョスカンデプレ (1450/55 - 1521) の代表曲「ミサ・ディ・ダディ」がある。この呼ばれ名は、楽譜のテノール部にサイコロの目が記されており、サイコロを振った出目によって曲を作ったと言われていることに由来している。モーツァルト (1756 - 1791) の「音楽のサイコロ遊び」と呼ばれるものもある。これは、あらかじめ作曲者が複数のフレーズを、それらが偶然どのように選ばれても違和感なく演奏できるように配慮して用意し、サイコロを振った出目によってフレーズを選び並べて演奏するといったものである。しかしこれらの例は、作曲者の音楽知識や作曲技術によるところが多く、音楽の自動生成とは言い難い。コンピュータの発達以降、様々な自動作曲ソフトが公開されており、DTM(Desk Top Music) のためのソフトに付属で自動作曲機能がついていたりもする。その中でも近年、深層学習を用いた自動作曲システムがみられる。例として、Ji-Sung Kim 氏の「Deep Jazz」(https://deepjazz.io/) がある。Jazz 曲の MIDI データを与えると、人工知能がその MIDI データを読み込み、それに似たジャズ曲を新しく自動で作曲するというものである。また、Ampermusic(https://www.ampermusic.com/) という自動作曲システムもある。これは、音楽知識のない人でも、音楽のジャンル、雰囲気、曲の長さや使う楽器などを入力することで、AI が自動で作曲を行うというものである。これらは音楽自動生成の先行研究の一例である。

（※文責: 伊藤祐大）

1.2 本プロジェクトの目標

そこで、本プロジェクトでは「プロのジャズミュージシャンの演奏を模倣するアドリブ自動生成システムの作成」を目標に設定した。この目標を設定するにあたって、いくつかのねらいがあった。まず、1.1 音楽自動生成の歴史にて述べたような先行研究において、プロの演奏を模倣するような、すなわち模倣の対象とする演奏家が実際に演奏したと思えるようなメロディーの自動生成システムは少ないことがある。したがって、この研究の成果は、自動作曲、自動演奏の研究に貢献できると考える。また、数ある音楽ジャンルの中でとりわけジャズのアドリブパートを題材としたのは、ジャズという音楽ジャンルの特徴に深く関係しており、これについては 2 章で詳しく述べる。

本プロジェクトでは、模倣するプロのジャズミュージシャンをマイルス・デイヴィス (1926 - 1991) とチャーリー・パーカー (1920 - 1955) の二人とした。マイルス・デイヴィスはモダン・ジャズの帝王と呼ばれ、ジャズ界を牽引した代表的なジャズ・トランペッターである。チャーリー・パーカーはモダン・ジャズの父と呼ばれ、モダン・ジャズの原型であるビバップスタイルの創成に携わった、こちらも代表的なジャズ・サクソフォニストである。マイルスの演奏のおおまかな特徴として、速いパッセージや激しい跳躍、ハイトーンなどのテクニックにあまり頼らず、またトランペットは中音域が美しいとして多用していたことなどがある。これらの特徴から、最初に模倣する目標として適していると考えた。チャーリー・パーカーの演奏はマイルスとは大きく異なり、ジャズにスリルをもたらした「デタラメ風音楽」の発明者などと言われるだけあって、高速指まわしや跳躍、アプローチ・ノートなどのテクニックを多用している。マイルスと真逆ともいえるチャーリーのこの特徴から、たとえば本プロジェクトが目標とするシステムによってマイルスを模倣し自動生成したマイルス風アドリブが作れたとして、それがマイルス本人のアドリブとどの程度似ているか、また全く違った特徴であるチャーリーのアドリブとどの程度似ていないか、などを調べるうえで適していると考えた。

今回模倣する課題曲として”Bags’ Groove” の Take1、”K.C.Blues” を選んだ。”Bags’ Groove” は、ジャズ・アドリブの入門とも言える F ブルースの曲であり、最初に模倣する目標として適していると考えた。”K.C.Blues” は、前述のマイルス、チャーリー両者がアドリブ演奏をした採譜データがあり、二人の演奏を比べるうえで適していると考えた。この F ブルースなどのジャズに関係する音楽理論については、2 章で詳しく説明する。また、マイルス・デイヴィス、チャーリーパーカー、”Bags’ Groove”、”K.C.Blues” については、6 章で詳しく説明する。

（※文責: 伊藤祐大）

1.3 課題の概要

本プロジェクトの通年の目標である「プロのジャズミュージシャンの演奏を模倣するアドリブ自動生成システムの作成」を達成するにあたって、多くの課題がある。これについて、本プロジェクトで行ってきたことに沿って説明する。まず、基礎的な音楽理論を学び、曲のつくりについて理解を深める。このとき、ジャズ特有の音楽理論、特に F ブルースにおける音楽理論について着目する。これらについては 2 章で述べる。次に、多くの音楽がしたがっているといわれる 1/f ゆらぎという性質について着目し、ガードナー [16] を参考にして、フラクタルの性質をもつ比例雑音である

白色雑音、褐色雑音、1/f ゆらぎの 3 種雑音について学ぶ。さらに、これらの雑音の特徴を用いた作曲システムを作成する。これについては 3 章で述べる。また、2 章で説明する F メジャースケール、F ブルーノートスケールに対し、それぞれに 3 種雑音の作曲システムを用いた計 6 種類のアドリブを生成し、これを比較実験する。このとき、比較実験について、実験方法や結果の検定方法などについても学ぶ。これらについては 3 章、4 章で述べる。

次に、比例雑音による作曲システムとは別のアプローチとして、1.1 音楽自動生成の歴史で述べたような先行研究が深層学習を用いて作られていることに着目し、その足掛かりとして、ニューラルネットを用いた機械学習による作曲システムの作成を目指す。そのために、斎藤 [20] を参考にして、ニューラルネットや機械学習についての基礎的な理解を深める。これについては、5 章で述べる。また、それらの知識を活かして、ニューラルネットを用いた機械学習による作曲システムを作成する。これについては、6 章で述べる。

このシステムの反省から、時系列データの扱いに長けたリカレントニューラルネットを用いた機械学習による作曲システムの作成を目指す。そのために、巣籠 [23] を参考にして、リカレントニューラルネットの基礎的な理解を深める。これについては 5 章で述べる。

それとは別に、ミュージシャンを学習するうえで採譜データが多く必要と考え、「Charlie Parker Omnibook: For C Instruments. (Treble Clef Version)」[Criterion Music Corp(1982)]、「Miles Davis Omnibook: For C Instruments」[Hal Leonard Corp; Spi 版 (2015)] を用いて、本プロジェクトが演奏の模倣を目指したチャーリー・パーカー、マイルス・デイヴィス両者合わせて 100 曲以上のジャズアドリブの採譜を MIDI データ化する。これについては 7 章で述べる。最後にリカレントニューラルネットを用いた機械学習による作曲システムを作成する。これについては 7 章で述べる。

次章以降では、具体的な活動内容について説明する。

（※文責: 伊藤祐大）

第 2 章　ジャズ理論

この章では、これから取り組むプロのジャズアドリブを模倣したフレーズの自動生成に先駆け、まずは音楽理論についての基礎知識とジャズの特有の音楽理論の特徴を述べていく。

（※文責: 佐藤悠介）

2.1　音楽理論の基礎

音楽の自動生成システムを作るにあたり、まずは現在使われている音階、つまり音の高さが定まるまでの歴史とそれらを用いた音楽理論について述べる。

現在私たちが耳にする音楽はド、ド#、レ、レ#、ミ、ファ#、ソ、ソ#、ラ、ラ#、シ、シ#という 12 種類の音階から任意の音を選び、並べて作られている。初めてこの 12 種類という音階に分けたのは、ピタゴラスが考案したピタゴラス音律であると言われている。また、ピタゴラス音律は世界最古の音階とも言われている。

東条ら [24] によると、「ピタゴラスは複数の長さの違う弦を同時に鳴らしてその調和の具合を考察した.」とある。その結果、ある弦の長さの半分の長さの弦を鳴らすと元の弦が出す音はとてもよく調和する音であることを発見した。これはある 2 つの音の周波数の比が 1 : 2 であり、これらの関係をオクターブ、または 2 倍音という。ピタゴラスはオクターブの関係の他に 1 : 3 の関係である 3 倍音もよく調和する音であることを発見し、これを用いてオクターブをいくつかに分割した。

例えば、ドの音を基準としてその 3 倍音は現在使われている音でいうとソの音にあたるが、このソの音はオクターブ外に存在してしまう。そこで 3 倍音である音の周波数に $\frac{1}{2}$ をかけて 1 オクターブ音階を下げることで基準であるドの音と同じオクターブ内に存在させる。つまり、ある音に $\frac{3}{2}$ をかけていき、基準となる音のオクターブ内に存在させ、それでもオクターブ外になってしまった場合はさらに $\frac{1}{2}$ をかけて 1 オクターブ音階を下げていく操作を続けていく。

この操作を行うことで「基準音 : 基準音のオクターブ上」の周波数比が 1 : 2 となる範囲をいくつかに分割することができる。ドを基準の音として先述した操作を 13 回行ったときの各音の名称と振動数比を求めると表 2.1 のようになる。

ここで 13 回目の操作で導き出される音に着目すると、基準音との周波数比がほぼ 1 : 1 となるのでピタゴラスはここで操作を中止してオクターブを 12 分割するにした。その後、新たに作られた音階はピタゴラス音律を改良していったものであり、現在使われている十二平均律と呼ばれる音階の元になっている。

加度 [17] によると、音楽理論の基礎としてスケールやコードというものがよく用いられる。また、音楽理論においてド、レ、ミ、ファ、ソ、ラ、シなどの音名は C、D、E、F、G、A、B のようにアルファベットの A から G を用いて表される。1 オクターブは C、C#、D、D#、E、F、F#、G、G#、A、A#、B の 12 種類の音から成り立っており、その 1 つ 1 つを半音と呼ぶ。また、C と D、A と B のようにそれぞれの音の間隔が半音 2 つ分であることを全音という。

また、ある 2 つの音の高さの関係を音程と呼ぶ。東条ら [24] によると、

> 半音 7 個の五度を完全五度（perfect fifth）, 半音 5 個の四度を完全四度（perfect fourth）と

表 2.1

音	振動数比
ド	1
ソ	$\frac{3}{2} \fallingdotseq 1.50$
レ	$\frac{3^2}{2^3} \fallingdotseq 1.13$
ラ	$\frac{3^3}{2^4} \fallingdotseq 1.69$
ミ	$\frac{3^6}{2^6} \fallingdotseq 1.27$
シ	$\frac{3^5}{2^7} \fallingdotseq 1.90$
ファ#	$\frac{3^6}{2^9} \fallingdotseq 1.42$
ド#	$\frac{3^7}{2^{11}} \fallingdotseq 1.07$
ソ#	$\frac{3^8}{2^{12}} \fallingdotseq 1.60$
レ#	$\frac{3^9}{2^{14}} \fallingdotseq 1.20$
ラ#	$\frac{3^{10}}{2^{15}} \fallingdotseq 1.35$
ファ	$\frac{3^{11}}{2^{17}} \fallingdotseq 1.35$
ド	$\frac{3^{12}}{2^{19}} \fallingdotseq 1.01$

呼ぶ. ある音階において五度が半音 8 個の場合を増五度 (augmented fifth), 半音 6 個の場合を減五度 (diminished fifth) と呼ぶ. 半音 4 個の三度を特に長三度 (major third), 半音 3 個の三度を短三度 (minor third) と呼ぶ. それ以外の二度, 六度, 七度は, 三度と同じ扱いである. つまり, 長二度と短二度はそれぞれ半音 2 個と半音 1 個に, 長六度と短六度は半音 9 個と半音 8 個に, 長七度と短七度は半音 11 個と半音 10 個に相当する.

とあり、音程にはいくつかの名称が存在する。また、半音 12 個に相当する音程を完全八度といい、音程の関係をまとめたものが表 2.2 になる。

表 2.2

2 音間の間隔（半音）	音程の名称
1	短二度
2	長二度
3	短三度
4	長三度
5	完全四度
6	減五度
7	完全五度
8	増五度, 短六度
9	長六度
10	短七度
11	長七度
12	完全八度

まず、加度 [17] によると、スケールとは 1 オクターブの中から選択された音の並びのことであ

る。例えば、ド、レ、ミ、ファ、ソ、ラ、シ、ド (C、D、E、F、G、A、B、C) という 7 種類の音で構成された並びはメジャースケールと呼ばれるスケールの一つであり、C、D、E、F、G、A、B は C メジャースケールという。（図 2.1）

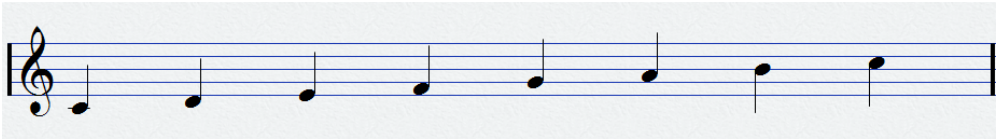


図 2.1　C メジャースケール

また、東条ら [24] によると、スケールの始まりとなる音を主音と呼び、主音の半音下の音を導音と呼ぶ。メジャースケールはある基準の音から長二度、長三度、完全四度、完全五度、長六度、長七度、完全八度となる音で構成されている。また、ラ、シ、ド、レ、ミ、ファ、ソ、(C、D、D#、F、G、G#、A#、C) という 7 種類の音で構成された音の並びはナチュラルマイナースケールというスケールの一つであり、この例だと C ナチュラルマイナースケールという。(図 2.2)

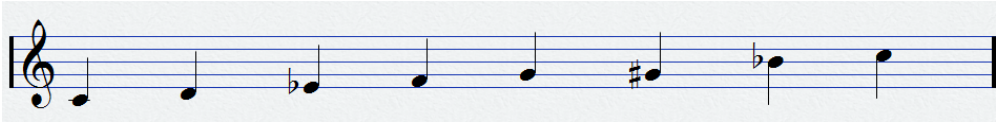


図 2.2　C ナチュラルマイナースケール

ナチュラルマイナースケールはある基準の音から長二度、短三度、完全四度、完全五度、短六度、短七度、完全八度となる音で構成されている。マイナースケールは他に 2 種類存在し、ナチュラルマイナースケールは導音が存在しないためにスケールを順に上昇させた音がやや硬く不自然な聴こえであるという点がある。これを解消するためにナチュラルマイナースケールの主音の短七度の音を半音上げて長七度にしたハーモニックマイナースケールというものがある。(図 2.3) しかし、ハーモニックマイナースケールは主音からそれぞれ短六度と長七度となる音の間隔が増二度となるため独特な響きを持つ。ハーモニックマイナースケールをさらに自然な響きにするために主音から短 6 度となる音も半音上げたスケールをメロディックマイナースケールという。(図 2.4)

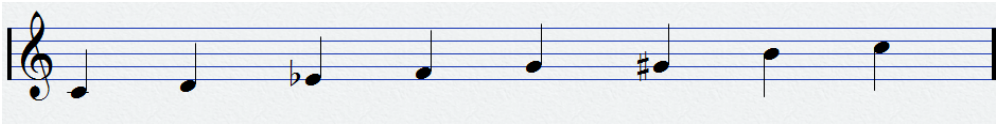


図 2.3　C ハーモニックマイナースケール

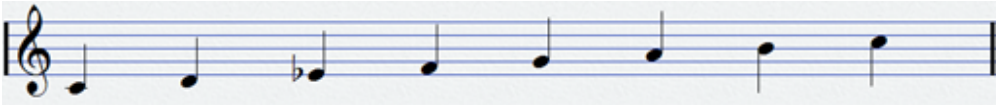


図 2.4　C メロディックマイナースケール

しかし、メロディックマイナースケールはメジャースケールと比較すると主音から短三度となるか長三度となるかの違いでしかないためメロディックマイナースケールの音を順に下降させていくとメジャースケールのように聴こえてしまう。そのため、下降させるときのみナチュラルマイナー

スケールを用いる。このようにスケールには様々な種類があるが、それらはまとめてダイアトニックスケールと呼ばれている。

作曲においてスケールと一緒にコードという考えを用いる。東条ら [24] によると、コードとは音の高さが異なる複数の音を同時に鳴らしたものであり、和音とも呼ばれる。3 つの音を同時に鳴らしたものは 3 和音と呼ばれ、4 つの音を同時に鳴らしたものは 4 和音と呼ばれる。コードの構成音の中でも特に 1 番下に積まれている音を根音、ルート音と呼ぶ。よく使われるコードとしてメジャーコードとマイナーコードというものがある。メジャーコードはルート音とその長 3 度上の音、そしてルート音とその完全 5 度の音の計 3 音からなるコードであり、マイナーコードはルート音とその短 3 度上の音、そしてルート音とその完全 5 度の音の計 3 音からなるコードである。例としてドをルート音とした場合のメジャーコードは CEG からなる C メジャー (単に C とも表記する)、同様にマイナーコードは CE ♭ G からなる C マイナー (Cm とも表記する) と呼ばれる。(図 2.5)

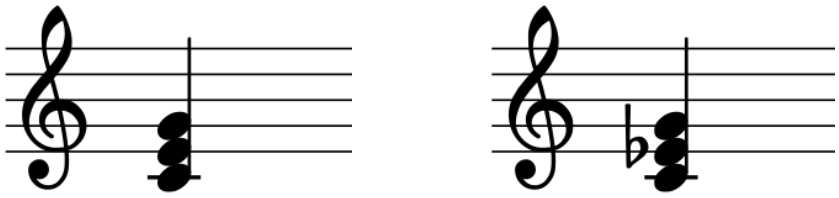


図 2.5　C メジャースケール（左）と C マイナースケール（右）

さらに先述した 3 和音のコードにメジャーコードのルート音から短 7 度にあたる音を付け加えたセブンスコード、メジャーコードのルート音から長 7 度の音を付け加えたメジャーセブンス、マイナーコードのルート音から短 7 度の音を付け加えたマイナーセブンス等の種類があり、ジャズにおいてよく用いられる。（図 2.6）

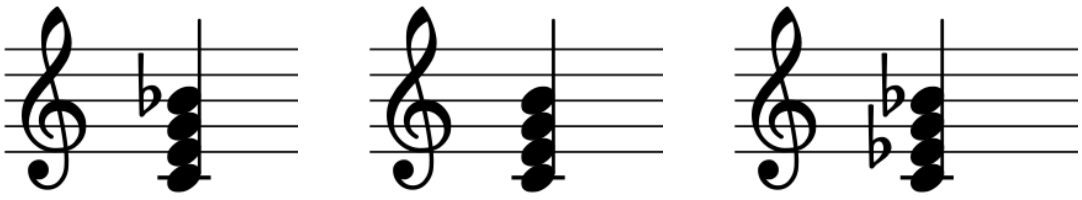


図 2.6　C セブンスコード（左）、C メジャーセブンスコード（中央）、C マイナーセブンスコード（右）

実際にコードを用いて作曲を行う場合には、様々な種類のコードを並べることで行われていく。このコードを並べたものをコード進行と呼び、コード進行を考える方法としてダイアトニックコードとカデンツという考え方がある。

東条ら [24] によると、ダイアトニックコードとはメジャースケールなどの 7 音で構成されたスケールの 7 音それぞれに割り当てられたコードのことである。それぞれのコードは全てそのスケール音の構成音のみからできている。例として C メジャースケールのダイアトニックコードはそれぞれ、C メジャー、D マイナー、E マイナー、F メジャー、G メジャー、A マイナー、B マイナー (♭ 5) というコードになり、これらのコードは C、D、E、F、G、A、B のみの音から構成されて

いる。最後の B マイナー (♭ 5) というコードはルート音とその短三度上の音、そしてルート音とその減五度の音の計 3 音からなるコードである。

ダイアトニックコードのコード一つ一つにはローマ数字が与えられており、I から VII の数字で表す。先ほどの C メジャースケールのダイアトニックコードを例とすると、表 2.3 のようになる。

表 2.3　C メジャースケールにおけるダイアトニックコード

I	II	III	IV	V	VI	VII
C	Dm	Em	F	G	Am	Bm

ダイアトニックコードにはそれぞれ機能があり、後述のカデンツという考えと深い関係がある。まず、I のコードが持つ機能をトニックと呼ぶ。トニックとはそのダイアトニックコードの中で安定感を持つものとされている。次に、V のコードが持つ機能をドミナントと呼ぶ。ドミナントはトニックの機能を持つコードへ進みやすいとされている。最後に IV のコードが持つ機能をサブドミナントと呼ぶ。サブドミナントはドミナントの機能を持つコードへ進みやすいとされている。残りの II、III、VI、VII のコードは代理コードと呼ばれており、それぞれサブドミナント、トニック、トニック、ドミナントの代わりとして使うことができる。

ダイアトニックコードを用いてコード進行を立てていくが、それにはカデンツという考えを用いることができる。東条ら [24] によると、

> 一般にカデンツとは, ドミナント機能を持つ和音によってもたらされた緊張をトニック機能を持つ和音に進行してリラックスさせ, 楽曲の終息感をもたらす和音の動きである. 与えられた調においては, V の和音 (ドミナント) での緊張は I の和音 (トニック) への進行によって「解決される」という.

とあり、カデンツは以下の 3 種類がある。

1. トニック - ドミナント - トニック
2. トニック - サブドミナント - ドミナント - トニック
3. トニック - サブドミナント - トニック

（※文責: 佐藤悠介）

2.2　ジャズにおける音楽理論

ジャズにおいてよく使われるスケールとコードにブルーノートスケールとチャーチモード (教会旋法)、セブンスコードがある。

東条ら [24] によると、ブルーノートスケールとはメジャースケールに用いられる音の第 3 音、第 5 音、第 7 音をそれぞれ半音下げたスケールのことを言う。例えば、C メジャースケールである C、D、E、F、G、A、B を C ブルーノートスケールにすると C、D、E ♭、F、G ♭、A、B ♭となる。(図 2.7) ブルーノートスケールの特徴として、メジャースケール、マイナースケール問わずに同じ調であればどこでも使うことができることや、スケール上の音はバラバラに使うのではなくスケールの音を順に降下、昇順させるなどの一定の方向性と連続性を持って使わないとブルーノートスケール特有の響きを得ることができないことなどがある。

ブルーノートスケールと同様にチャーチモード (教会旋法) というスケールもジャズにおいて用

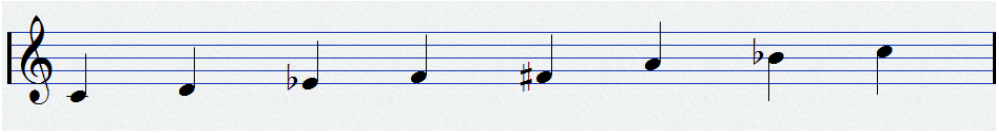


図 2.7　C ブルーノートスケール

いられる。チャーチモードは 7 種類あるが、それらはメジャースケールの開始位置が異なるだけなので簡単に作ることができる。C メジャースケールを例とした場合、C から始まる C、D、E、F、G、A、B を C イオニアンスケールと呼ぶ。(図 2.8) 同様に D から始まる D、E、F、G、A、B、C を D ドリアンスケール (図 2.9)、E から始まる E、F、G、A、B、C、D を E フリジアンスケール (図 2.10)、F から始まる F、G、A、B、C、D、E を F リディアンスケール (図 2.11)、G から始まる G、A、B、C、D、E、F を G ミクソリディアンスケール (図 2.12)、A から始まる A、B、C、D、E、F、G を A エオリアンスケール (図 2.13)、B から始まる B、C、D、E、F、G、A を B ロクソリディアンスケールと呼ぶ。(図 2.14)

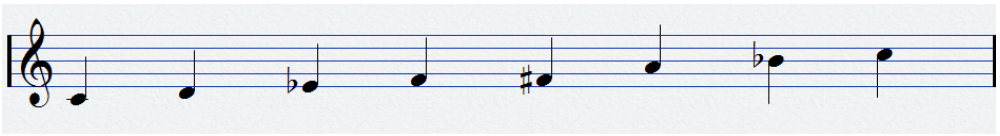


図 2.8　C イオニアンスケール

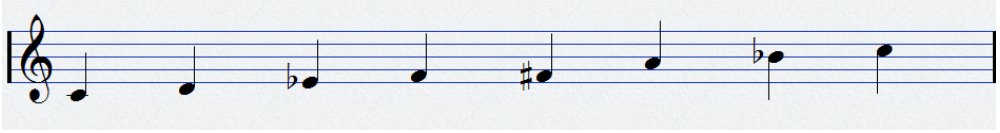


図 2.9　D ドリアンスケール

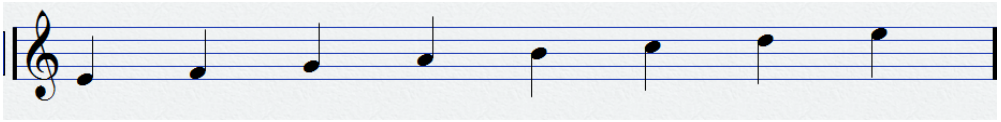


図 2.10　E フリジアンスケール

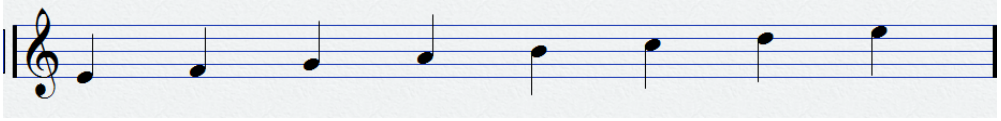


図 2.11　F リディアンスケール

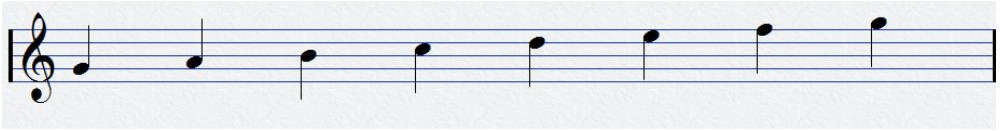


図 2.12　G ミクソリディアンスケール

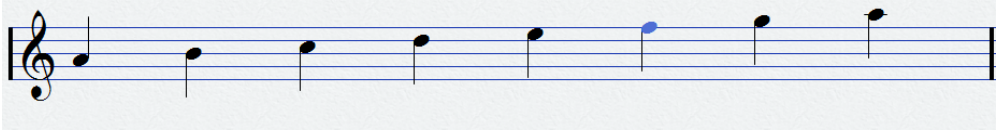


図 2.13　A エオリアンスケール

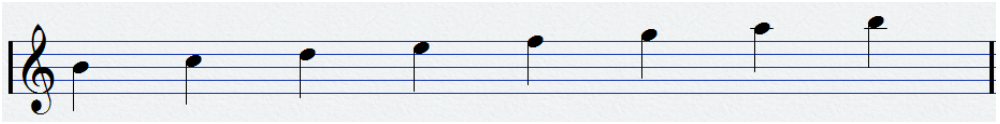


図 2.14　B ロクソディアンスケール

ジャズにおいてはメジャースケールのルート音から短七度の音を付け足したセブンスコードという４和音のコードがよく用いられている。ジャズでは曲中のほとんどのコードがセブンスコードであり、第４の音がその曲で用いられているスケールに存在しない音になっている場合がある。このコード進行に加えて実際の調に含まれない音を構成音に持つブルーノートスケールも合わせると、これらの音は不協和という、聴いていて違和感を覚えるような響きを持つが、私たちの耳ではこのコード進行とスケールが織りなす曲に対して違和感を覚えることはないのである。Levine[3] によると、この現象は西洋音楽の理論では説明することができないと述べている。

ピアノ演奏の際には左手はコードを意識した伴奏、右手はメロディを担当するのだが、セブンスコードを弾く場合は左手のみで 4 つの鍵盤を抑えるのは困難であるため、コードの第 3 音と第 7 音の鍵盤のみを抑える。これにより 2 つの鍵盤を押す際の指の開き方を維持すれば様々なセブンスコードを演奏中でも容易に抑えることができる。

前期の活動では F ブルースと呼ばれる音楽を作成したが、ここでは今回 F ブルースを選んだ理由を述べる。F ブルースとはスケールが F のブルースと呼ばれる音楽のことで、ジャズ、ブルースにおいてよく使われる。ブルースの特徴としてフレーズを演奏する際に曲のコード進行に合わせてスケールを変える必要がないので演奏がしやすいという利点がある。F スケールがブルースにおいてよく使われる理由としてピアノやトランペットにおいて弾きやすいスケールだからである。例えばピアノで F スケールは一音を除いて白鍵のみで演奏することができるため演奏がしやすく、トランペットなどの楽器にも同様のことがいえる。(図 2.15)

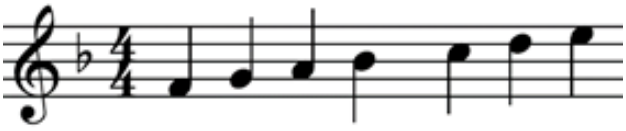


図 2.15　F メジャースケール

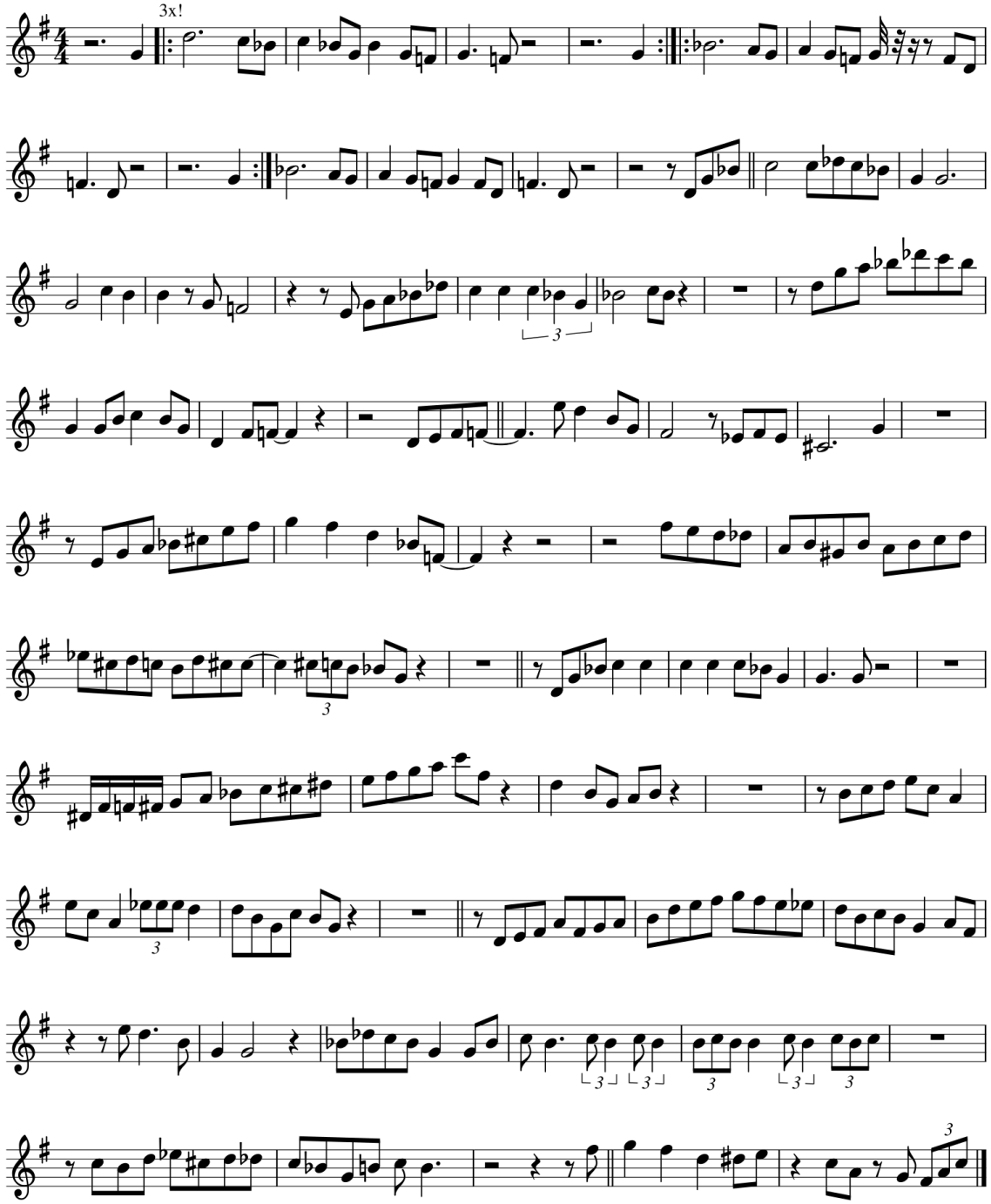
つまり、F ブルースは伴奏のコードを気にせずに単一のスケールのみでフレーズを演奏することができ、かつ演奏がしやすいスケールという特徴を持った曲であるといえる。

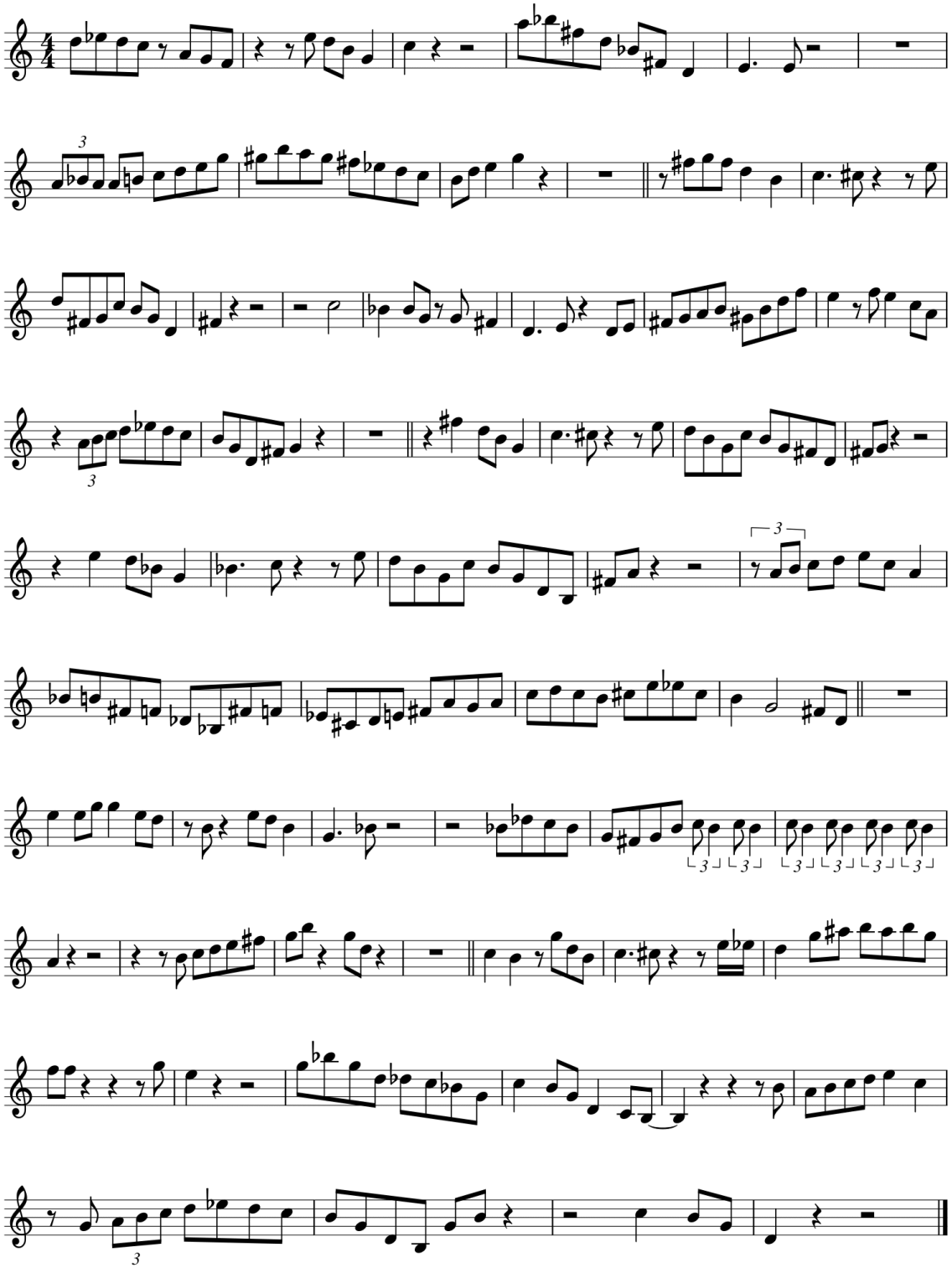
次ページからの楽譜は、F ブルースの曲である”Bags’ Groove take1” の楽譜である。

楽譜の作成には音楽ソフト「Cubase8.0」、「MuseScore2」、「Finale」を用いた。

（※文責: 佐藤悠介）

Bags' GROOVE





2.3　音楽的なアドリブの作成方法

ジャズのメロディの特徴の一つとしてアドリブというものがある。アドリブとは、奏者の即興演奏のことで、基本的にはあらかじめ決められたコード進行と曲のムードに沿って演奏される、1940年代初頭のビバップ期以降の、いわゆるモダン・ジャズにおいて外せない要素の一つである。このプロジェクトの目標であるアドリブの自動生成に向け、アドリブの作成方法について加度 [17] が説明している一例を紹介する。

まず、ジャズ・アドリブのフレーズは「スケール」、「アルペジオ」、「半音階」などの音の役割によって分類される。スケールとは、曲のスケールに沿って順々に音を鳴らしていく奏法であり、用いることで曲の調を明確にできる。アルペジオとは、和音の構成音をばらして一音ずつ発音する奏法であり、用いることでコードを明確にできる。例として C メジャーコードのアルペジオは C、E、G の音のみを用いて鳴らすフレーズとなる。(図 2.16)

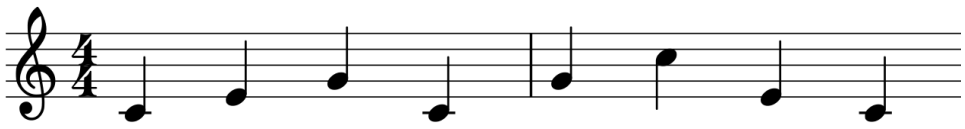


図 2.16　C メジャーコードのアルペジオ

半音階とは、曲のスケールに関わらず半音ずつ順々に音を鳴らしていく奏法であり、ジャズにおける特徴的な響きに欠かせないものである。しかし、スケールを外れる音が多く出てくるため、使いすぎると違う調の演奏に聴こえてしまうことがある。以上のような役割を持つ音を効果的に並べることで、ジャズ・アドリブのフレーズを作ることができる。

また、フレーズを作るうえでのジャズっぽさを表現するテクニックがいくつか紹介されている。まずは休符の入れ方である。音楽には強い性質を持つ拍と弱い性質を持つ拍とが交互に訪れる特徴があり、4 拍子は順番に強拍、弱拍、強拍、弱拍と分けられる。(図 2.17)

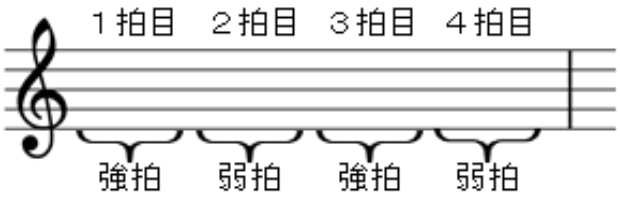


図 2.17　強拍と弱拍

ジャズやポップスなどアメリカ発祥のいわゆる新しい音楽には弱拍を補うという性質があり、ジャズを演奏する際には強拍に休符を作り、弱拍からフレーズを始めることが多い。またこの性質にしたがって、弱拍の音にアクセントをつけることでもジャズ・アドリブのフレーズ感を引きだすことができる。

次にテンション・ノートとアボイド・ノートである。ジャズでは、テンション・ノートというコードの構成音から外れた音がよく用いられる。テンション・ノートはコードの 9th、11th、13th のことであり、用いるコードの三和音に対して半音高い音になっているもの以外を使用する。三和

音に対して半音高い音になっているものは和音の機能を壊してしまうため、アボイド・ノートと呼ばれ、使用は避けられる。(表 2.4)

表 2.4　アボイド・ノート一覧（C メジャースケール）

ダイアトニックコード	アボイド・ノート
I（C）	完全四度
II（Dm）	長六度
III（Em）	短二度、短六度
IV（F）	なし
V（G）	完全四度
VI（Am）	短六度
VII（Bm（♭５））	短二度

最後にアプローチ・ノートである。主音の半音下の音のことを導音といい、この音が主音を導くという性質を応用したものがアプローチ・ノートと呼ばれる。導きたい音をターゲット・ノートと呼び、その半音下の音をターゲット・ノートの直前に使用し、ターゲット・ノートを導くものがアプローチ・ノートである。以上のようなテクニックを用いることで、よりジャズらしいフレーズを作ることができる。

アドリブの作成方法について、前期のプロジェクトでは音の役割によって音を選ぶことはしてこなかったため、今後の課題になると考える。

（※文責: 伊藤祐大）

第 3 章　雑音について

1/f ゆらぎの性質を持つ音楽は、心地良いとされている。この事は、ジャズでも等しく言えるのだろうか。それを調べるために、白色雑音、褐色雑音を用いて比較実験を行った。この章では、白色雑音と褐色雑音、1/f ゆらぎについてガードナー [16] を参考に記述する。

（※文責: 吉田周平）

3.1　白色雑音、褐色雑音、1/f ゆらぎについて

まず、「比例雑音（scaling noise)」について説明する。比例雑音とは、元の速度とは異なる速さで再生した際に音の大きさを調整することで、元の音と同じように聞こえる性質をもつ音である。

「白色雑音 (white noise)」は、ある時点の音と次の音との相関が全くない音で、自己相関係数は 0 に限りなく近くなる。

「褐色雑音 (Brouwnian noise)」は、ある時点の音と次の音との相関が強く、自己相関関数は 1 に限りなく近くなる。また、名前はブラウン運動からきており、白色雑音に合わせて褐色雑音と言う。

1/f ゆらぎは、その白色雑音と褐色雑音の性質を折衷したものである。そこで、折衷したことによって相関が強すぎず弱すぎずであることが、心地よさの理由の有力な説の一つとなっている。

（※文責: 吉田周平）

3.2　各雑音の特徴を有した作曲システム

白色雑音と褐色雑音、1/f ゆらぎのそれぞれの特徴を有する楽曲を乱数による作成システムを、ガードナー [16] に記述されている方法を用いて作成した。また、全てのシステムで休符の出現頻度を 30〜31% とした。

最初に、白色雑音の特徴を有した作曲システムについて説明する。白色雑音の特徴を有する楽曲は、サイコロやルーレットのような乱数発生器で生成できる。今回使用した方法は、特定の数の音の中で、真ん中の音が最も出やすくなるように作成した。例えば 7 音なら、ド：レ：ミ：ファ：ソ：ラ：シ = 1 : 2 : 3 : 4 : 3 : 2 : 1 となるようにした。視覚的にルーレットで表すと図 3.1 のようになる。

また、このような確率で作成した音の波形は図 3.2 のようになる。

図 3.2 は、横の軸が時間、縦の軸が音の高さとなっている。波形からは、前の音と次の音との相関性を認められない。また、これは 14 音で行った場合の波形だが、音の数を増やすことによって振れ幅が大きなることがわかった。

次に、褐色雑音の特徴を有した作曲システムについて説明する。褐色雑音の特徴を有する楽曲は、特定の音から次の音への移動距離を乱数算出して、それを繰り返すことで生成した。移動距離については、$-2 : -1 : 0 : +1 : +2 = 1 : 1 : 1 : 1 : 1$ という割合の確率になるようにした。視覚的にルーレットで表すと図 3.3 のようになる。

また、このような確率で作成した音の波形は図 3.4 のようになる。

図 3.1　白色雑音のルーレット

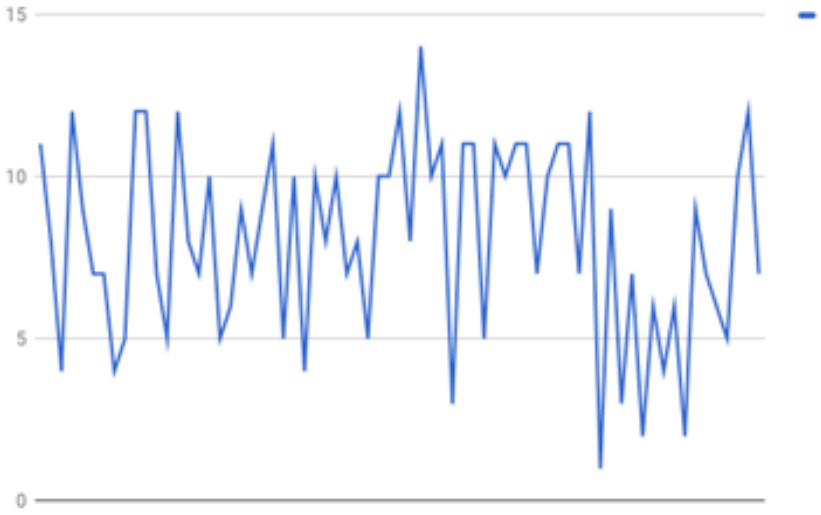


図 3.2　白色雑音の波形

図 3.3　褐色雑音のルーレット

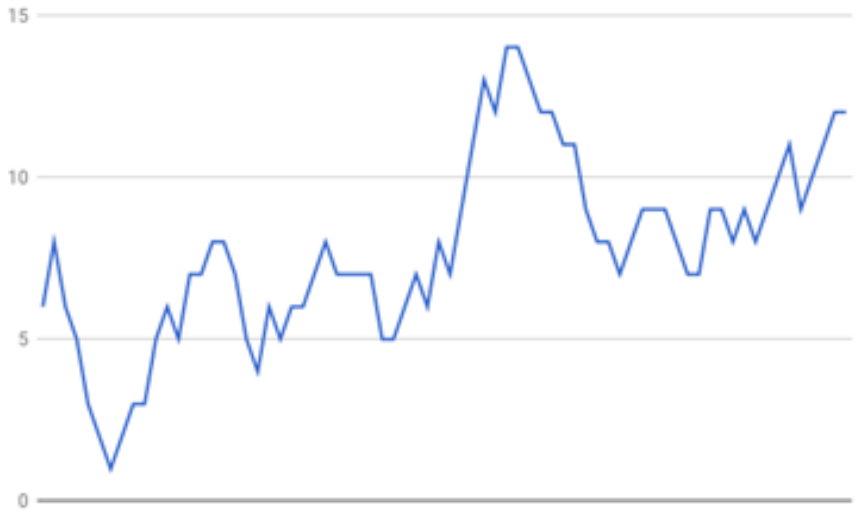


図 3.4　褐色雑音の波形

図 3.4 は、横の軸が時間、縦の軸が音の高さとなっている。波形からは、特定の音と次の音を比較すると比較的変化が小さいことがわかる。また、使用する音の数に合わせて音の移動距離を変えることで、汎用性を増すことが出来た。

さらに、褐色雑音に関しては「反射壁」と「吸収壁」と言う考え方を使用した。まず、褐色雑音のシステムは、制限を付けなければどこまでも高い音や低い音になってしまう。そこで、壁という考え方をする。

反射壁は、特定の音の範囲の中で一番高い音や低い音を超えそうになると、超えそうになった分だけ音を戻すという考え方だ。対して、吸収壁は、特定の音の範囲の中で一番高い音や低い音を超えそうになると、一番高い音または一番低い音で止まるという考え方である。

そして、1/f ゆらぎの特徴を有した作曲システムについて説明する。このシステムは、ガードナー [16] を参考にして、制作した。まず、複数個のサイコロがあると想定する。今回は 3 個とする。それぞれのサイコロをサイコロ A,B,C とする。そのサイコロを転がして、その目の総和に対応する音を出力する。その後 1 回目の操作をする。1 回目の操作はサイコロ C のみを振りなおす。そして、総和を計算しなおして対応する音を出力。2 回目の操作はサイコロを B のみを振りなおして、3 回目はサイコロ B,C の二つを振りなおす。

n 回目に振りなおすサイコロは、表 3.1 で「1」となっている場合になっている。

これは、n 回目の操作について、n を 2 進数にしたときの特定の桁にサイコロを対応させている。この表の場合は、サイコロ C を一桁目にサイコロ B を二桁目に対応させ、サイコロ A を三桁目に対応させている。そして、対応した桁の数字が「1」になったら、そのサイコロを振りなおして総和を変える。

上記の方法を使ったシステムによって作った際の波形は図 3.5 のようになった。

音の変化量が大きい時と小さい時があるが、それは振りなおすサイコロの数に依存するためである。変化量の最小値・最大値も変化するため、褐色雑音ほど規則的ではない。また、少なからず前の音に依存しているため、白色雑音ほど不規則的ではないと言える。故に、この方法で規則的すぎず不規則的すぎない音が出力できるのである。

また、システムの正当性を確かめるために、以上の方法にてサンプルデータを 500 個用意した。そして、そのデータを散布図とヒストグラムを用いて比較・検証する。以下にその図を示す。ただ

表 3.1

	サイコロ A	サイコロ B	サイコロ C
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

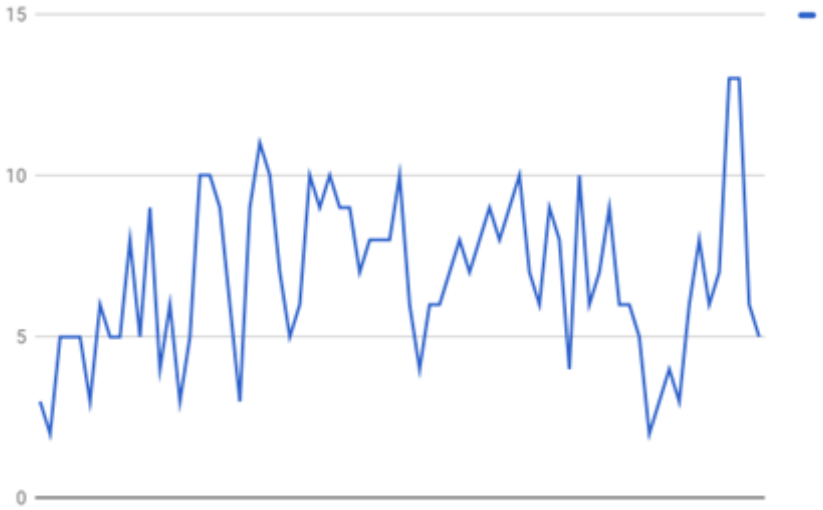


図 3.5　1/f ゆらぎの波形

し、散布図については横軸を $x(n)$ として縦軸を $x(n+1)$ とした。

次に、ヒストグラムを図 3.7 に示す。縦軸は音の出現回数、縦軸は音の高さを表す。

以上のグラフより、それぞれのシステムについて、各特徴を十分に有していると言える。よって、このシステムを使用した実験を行った。その実験の方法と結果を次章に記す。

（※文責: 吉田周平）

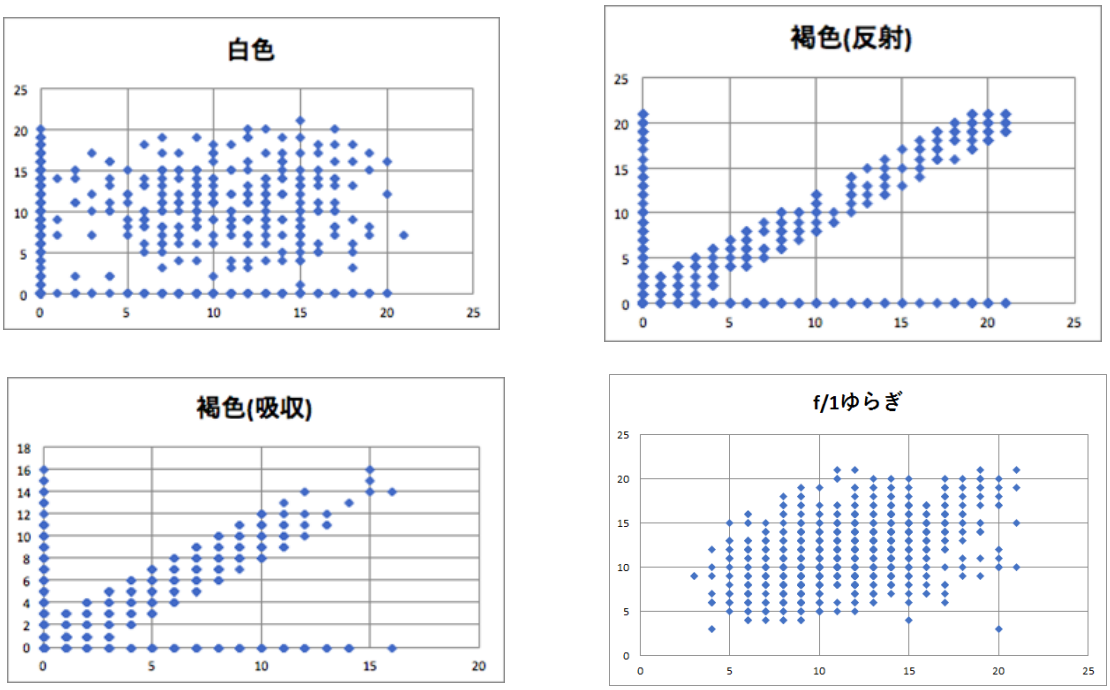


図 3.6　散布図

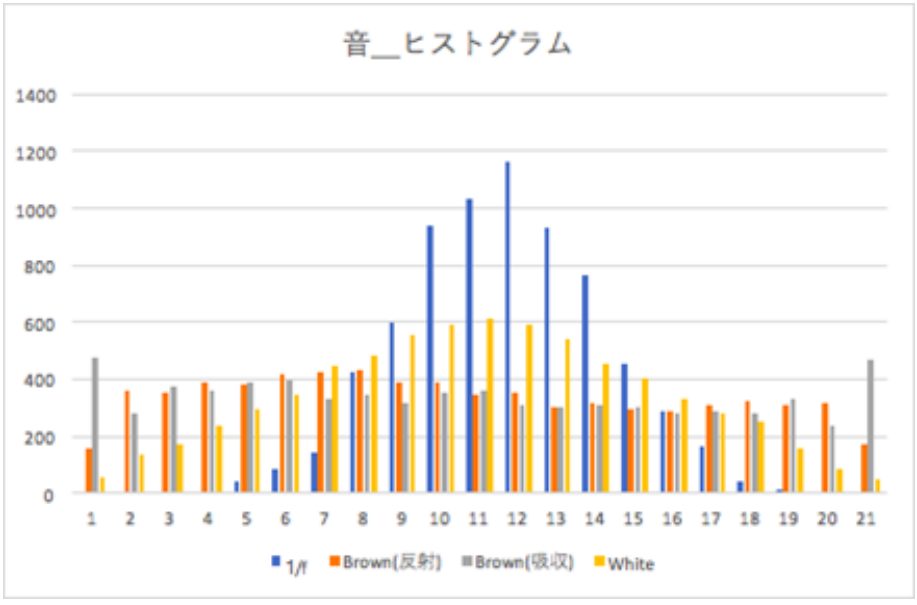


図 3.7

第 4 章　実験

3 章で記述した「白色雑音、褐色雑音、1/f ゆらぎの特徴を有した作曲システム」を使用して、比較実験を行った。この章では、実験の「方法と手順」をおよび「結果と考察」を述べる。

（※文責: 吉田周平）

4.1　方法

実験方法として、二重盲検法による評価方法を用いた。二重盲検法とは、評価実験における正当性を与えるための方法である。今回の実験では、出力データから曲を作成した人と曲の再生順を入れ替えた人が評価しないことで、二重盲検法を用いた評価方法であると言える。

評価は A,B,C の三段階で評価した。それぞれ「A：音楽になっている」「B：聴くことができる」「C：音楽になっていない」という基準を設定した。

作曲システムは「白色雑音」「褐色雑音」「1/f ゆらぎ」の特徴を有したものを使用した。また、褐色雑音に関しては反射壁と吸収壁のそれぞれのシステムを用意した。つまり、乱数の生成方法は 4 種類である。

スケールはＦメジャースケールとＦブルーノートスケールの 2 種類を使用した。

乱数の生成方法が 4 種類とスケールが 2 種類で、計 8 パターンの方法による作曲を行った。各方法につき 5 曲用意した。

さらに、以下の条件も付け加える。

- 評価人数は、学生８名であった。
- 作曲はアドリブ部分の 12 小節 48 音（全て四分音符または四分休符）とした。
- 実験する曲は、十分に聴こえる大きさで流した。
- 作曲にはピアノの音を用いた。

以上の条件の中で実験を行った。

次に、実験を行った際の手順について記述する。

1. スピーカーの音量を合わせて、お手本のアドリブを流す。
2. 曲の順番を入れ替えた人が、曲を順番に流す。
3. 評価者は評価基準に沿って評価する。

以上の手順で評価実験を行った。

評価の結果を「結果と考察」にて記述する。

（※文責: 吉田周平）

4.2 結果

実験による評価結果を A を 3 点、B を 2 点、C を 1 点として集計する。一つの作曲方法につき 5 曲用いたので、その合計点と平均点を表 4.1、4.2 に示す。

表 4.1 評価点

	白色	褐色（反射）	褐色（吸収）	1/f ゆらぎ	合計
F メジャー	56	61	67	67	251
F ブルーノート	53	71	66	63	253
合計	109	132	133	130	504

表 4.2 平均点

	白色	褐色（反射）	褐色（吸収）	1/f ゆらぎ	周辺平均値
F メジャー	11.2	12.2	13.4	13.4	12.55
F ブルーノート	10.6	14.2	13.2	12.6	12.65
合計	10.9	13.2	13.3	13.0	

そして、今回の実験の分析方法として「二元配置分散分析」を用いた。二元配置分散分析は、1/f のブルーノートスケールの曲が優位であるかを統計的に検討するために使用した。その計算結果を表 4.3 に示す。

表 4.3 二元配置分散分析

変動要因	変動	自由度	分散	観測された分散比	p-値	F-境界値	
スケール	0.1	1	0.1	0.02	0.88842365	4.149097744	帰無仮説は棄却されない
乱数	39	3	13	2.6	0.6921029	2.90111958	帰無仮説は棄却されない
交互作用	12.5	3	4.1666667	0.833333	0.48548861	2.90111958	帰無仮説は棄却されない
誤差	160	32	5				
全体	211.6	39					

結果として、p-値が F-境界値の範囲内に収まったあめに帰無仮説は棄却されず、ブルーノートスケールの 1/f が優位性を示すことは出来なかった。

（※文責: 中川哲哉）

4.3 考察

実験が失敗した理由としては、音符の長さが四分音符しかなかったことや、被験者がジャズを聴き慣れていないこと等があげられる。それらは、後期への課題とされる。

今回の実験では、1/f ゆらぎの特徴を有した作曲システムに優位性は見られなかった。そこで、新たな作曲システムの手法として、ニューラルネットワークを用いた。次章では、ニューラルネッ

Fractal × Jazz

トワークについて説明していく。

（※文責: 中川哲哉）

第 5 章　ニューラルネットワークについて

3 章では、ガードナー [16] を参考に、白色雑音、褐色雑音、1/f ゆらぎの特徴を有する音楽の生成を、乱数によって行うシステムを制作した。また、心地のよい音楽は 1/f ゆらぎの特徴を有しているのかを検証するため、作成した 3 つのシステムを用いて、F メジャースケールと F ブルーノートスケールの楽曲を生成し、評価実験を行った。結果、3 種の雑音および 2 種のスケールの楽曲の評価に優位差は見られなかった。

実験によって、1/f ゆらぎを有する楽曲が必ずしも心地のよい音楽とはいえないことが分かり、我々は次のアプローチについて思案した。そして、ジャズの自動作曲システムである Ji-Sung 氏の「Deep Jazz」が深層学習を用いていることなどから、ニューラルネットワークを用いた楽曲システムの作成を行うことにした。

本章では、まず、ニューラルネットワークの起源となったパーセプトロンというアルゴリズムについて説明する。その後、ニューラルネットワークの概要とその学習について解説する。また、学習した時系列データの扱いに特化したモデルである、「リカレントニューラルネットワーク」についても記載する。

最後に、本章は斉藤 [20] および、巣籠 [23] を参考に記述した。

（※文責: 居附未紗）

5.1　パーセプトロン

ここでは、パーセプトロンについての説明を記述する。以下の説明は斎藤 [20] の p21 から p37 までを参考に作成した。パーセプトロンとは、1957 年にローゼンブラットによって考案されたアルゴリズムである。複数の信号を入力として受け取り、一つの信号を出力する。パーセプトロンの信号は流す (1)/流さない (0) の 2 値の値であり、先へと伝達していく。ニューラルネットワークやニューラルネットワーク（ディープラーニング）の起源とされるアルゴリズムのため、ディープラーニングの考え方を学ぶためには、パーセプトロンを学ぶべきとされる。以下にパーセプトロンの図の例 (図 5.1) を表す。

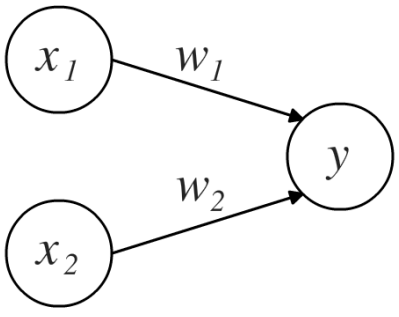


図 5.1　パーセプトロンの例

図 5.1 にある○はニューロン、またはノードと呼ばれる。ここでは受信した信号の総和が計算さ

れ、ある限界値を超えた場合にのみ信号を送る。このことをニューロンが発火すると表現し、ある限界値のことを閾値と呼ぶ。以下にパーセプトロンの式を表す。x_1、x_2 を入力信号、w_1、w_2 を重み、θ を閾値、y を出力信号として表している。ここで言う重みは信号の流れやすさをコントロールする値としての意味を持つ。まずニューロンに入力信号が送られ、重みが乗算される。その先のニューロンで、入力信号と重みが乗算された固有の重み（x_1、w_1、x_2、w_2）の総和が出力信号 y として出力される。この出力信号 y が閾値以下であった場合に 0 を、閾値を超えていた場合に 1 を出力する。入力信号それぞれの固有の重みは各信号をコントロールする要素として働き、信号の重要性は重みの大きさに比例して高くなる。

$$y = \begin{cases} 0 & (w_1x_1 + w_2x_2 \leq \theta) \\ 1 & (w_1x_1 + w_2x_2 > \theta) \end{cases} \tag{5.1}$$

例として、パーセプトロンを用いて AND,NAND,OR ゲートの論理回路の真理値表を満たすような w_1、w_2、θ の値を考える。まず、AND ゲートについて考えていく。AND ゲートは 2 つの入力、つまり x_1、x_2 が 1 のときに y は 1 を出力し、それ以外のパターンだと 0 を出力する。表 5.1 はそれを表した真理値表である。

表 5.1　AND ゲートの真理値表

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

これをパーセプトロンで表現する場合、表 5.1 の真理値表の条件をみたすような w_1、w_2、θ の値を設定する。まず、(5.1) 式にそれぞれのパターンの x_1、x_2 の値を代入し、閾値 θ の範囲を求める。例えば $x_1 = 0$、$x_2 = 0$ の場合、$w_1x_1 + w_2x_2 = w_1 \times 0 + w_2 \times 0 = 0$ よって $y = 0$ より $0 \leq \theta$ が範囲となる。これを他のパターンの x_1、x_2 の値で求めてみると、表 5.1 における θ の範囲はそれぞれ表 5.2 のようになる。

表 5.2　AND ゲートの閾値の条件を加えた真理値表

x_1	x_2	θの範囲	y
0	0	$0 \leq \theta$	0
0	1	$w_2 \leq \theta$	0
1	0	$w_1 \leq \theta$	0
1	1	$w_1 + w_2 > \theta$	1

表 5.2 より、出力結果が $y = 1$ となるためには、$0 \leq \theta$、$w_1 \leq \theta$、$w_2 \leq \theta$、$, w_1 + w_2 > \theta$ の条件を満たす w_1、w_2、θ の値を求めれば良い。これを満たすような値はグラフで表すと図 5.2 のようになる。x 軸を x_1、y 軸を x_2 として $(w_1, w_2, \theta) = (0.5, 0.5, 0.7)$ の場合の 2 次元グラフであり、出力が 1 のときの座標を○、0 のときの座標を●で表している。これを $0.5x_1 + 0.5x_2 \leq 0.7$ の直線で 2 分すると○と●の座標を 2 分することができる。$0.5x_1 + 0.5x_2 \leq 0.7$ の範囲内にある座標では出力が 0 であり、範囲外の座標では出力が 1 となる。

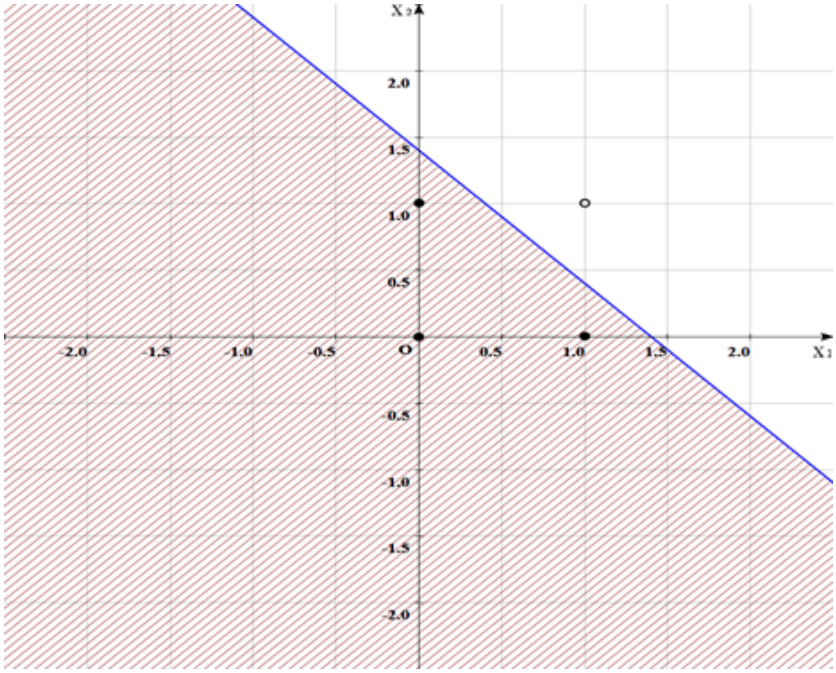


図 5.2　AND ゲートが出力 0 となる範囲

次に NAND ゲートについて考えてみる。NAND ゲートは AND ゲートとは逆で、x_1、x_2 が 1 のときに y は 0 を出力し、それ以外のパターンだと 1 を出力する。

表 5.3　NAND ゲートの真理値表

x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	0

今回も AND ゲートのときと同じように閾値 θ の範囲を求めるが、$x_1 = 0$、$x_2 = 0$ のとき $w_1x_1 + w_2x_2 = 0 \leq \theta$ となるので、θ の値が正の場合 $y = 1$ と出力したいのに $y \neq 1$ となっていまい条件が成立しなくなる。そこで、新たに $-1 \leq \theta \leq 0$、$-1 \leq w_1 \leq 0$、$-1 \leq w_2 \leq 0$、$w_1 + w_2 = -1$ を条件として考えてみると、$x_1w_1 + w_2x2 = 0 > \theta$ となり $y = 1$ と出力できるので条件が成立する。新たに追加した条件を元にそれぞれのパターンの x_1、x_2 の値を代入していくと表 5.4 のような範囲になる。

表 5.4　NAND ゲートの閾値の条件を加えた真理値表

x_1	x_2	θの範囲	y
0	0	$0 > \theta$	1
0	1	$w_2 > \theta$	1
1	0	$w_1 > \theta$	1
1	1	$w_1 + w_2 \leq \theta$	0

表 5.4 より、出力結果が $y = 1$ となるためには $0 > \theta$、$w_2 > \theta$、$w_1 > \theta$、$w_1 + w_2 \leq \theta$ の条件を満たす w_1、w_2、θ の値を求めれば良い。今回では $(w_1, w_2, \theta) = (-0.5, -0.5, -0.7)$ の場合のラフを表している。AND ゲートの時と同じように、出力が 1 のときの座標を○、0 のときの座標を●で表している。これを $-0.5x_1 - 0.5x_2 \leq -0.7$ の直線で 2 分すると、○と●の座標を 2 分することができる。

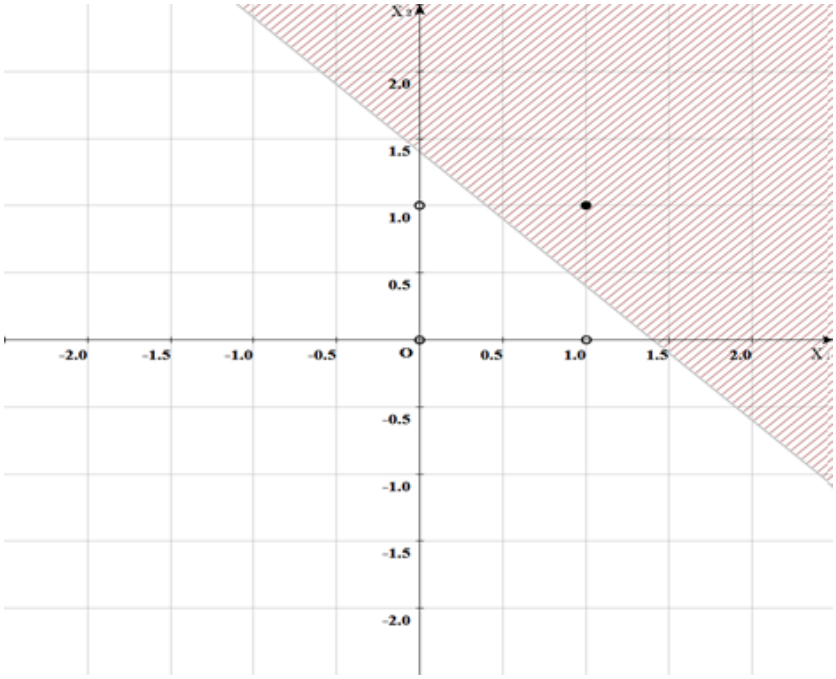


図 5.3　NAND ゲートの出力が 0 となる範囲

最後に、OR ゲートについて考える。OR ゲートは入力信号 x_1、x_2 の値のいずれかが 1 であれば出力が 1 になる。

表 5.5　OR ゲートの真理値表

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

これも同じように θ の範囲を求めていく。$x_1 = 1$、$x_2 = 0$ の場合、$w_1x_1 + w_2x_2 = w_1 > \theta$ となれば 1 が出力される。$x_1 = 0$、$x_2 = 1$ の場合でも $w_2 > \theta$ となれば 1 が出力され、$x_1 = 1$、$x_2 = 1$ では $w_1 + w_2 > \theta$ となり、θ の範囲は表 5.6 のようになる。

表 5.6 より、出力結果が $y = 1$ となるためには $0 \leq \theta$、$w_2 > \theta$、$w_1 > \theta$、$w_1 + w_2 > \theta$ の条件を満たす w_1、w_2、θ の値であれば良い。今回では $(w_1, w_2, \theta) = (0.5, 0.5, 0.4)$ の場合のグラフを表している。座標○、●を $0.5x_1 + 0.5x2 \leq 0.4$ の直線で 2 分したグラフが図 5.4 になる。$0.5x_1 + 0.5x_2 \leq 0.4$ の範囲にはいっている座標●が出力 0 を表す。グラフの作成には「GRAPES 7.35」[友田勝久] を用いた。

以上のようにパーセプトロンを用いれば論理回路の表現を行うことが可能であることがわかっ

表 5.6　OR ゲートの閾値の条件を加えた真理値表

x_1	x_2	θの範囲	y
0	0	$0 \leq \theta$	0
0	1	$w_2 > \theta$	1
1	0	$w_1 > \theta$	1
1	1	$w_1 + w_2 > \theta$	1

図 5.4　OR ゲートの出力が 0 となる範囲

た。一例を通してみるとパーセプトロンの構造は AND、NAND、OR ゲートの全てで同じであり、異なるのは重みと閾値のパラメータのみである。構造が同じであるため、値を変えるだけで AND、NAND、OR ゲートに対応することができる。

ここで出力結果を更にコントロールするためにバイアス b の導入を行う。閾値 θ を $-b$ とし、以下の式に表す。重み（w_1, w_2）は入力信号の重要度をコントロールし、バイアス（b）は $y = 1$ になるように発火のしやすさを調整するパラメータとして働く。このバイアスが負の方向に向かえば向かうほど発火がしやすくなり、出力 1 が出やすくなる。

$$y = \begin{cases} 0 & (b + w_1 x_1 + w_2 x_2 \leq 0) \\ 1 & (b + w_1 x_1 + w_2 x_2 > 0) \end{cases} \tag{5.2}$$

パーセプトロンを用いれば AND、NAND、OR ゲートの倫理回路を表すことが出来たが、単層のパーセプトロンには限界が存在する。ここで限界の一つの例としてバイアスを加えた式を用いて XOR ゲートについて考える。XOR ゲートは排他的論理和と呼ばれる論理回路であり、$x_、x_2$ の値のいずれかが 1 のときのみ、出力が 1 となる回路である。

ここで、x_1 を x 軸、x_2 を y 軸の 2 次元グラフにして考えてみる。$(b, w_1, w_2) = (-0.2, 1, 1)$ のとすると、パーセプトロンは $-0.2 + x_1 + x_2 = 0$ の直線で分断される領域を作る。一方は 0 を、もう一方は 1 を出力する。これを $-0.2 + x_1 + x_2 = 0$ の直線で●と○を 2 分するように分断され

表 5.7　XOR ゲートの真理値表

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

表 5.8　XOR ゲートのバイアスの条件を加えた真理値表

x_1	x_2	θの範囲	y
0	0	$b \leq 0$	0
0	1	$b + w_2 > 0$	1
1	0	$b + w_1 > 0$	1
1	1	$b + w_1 + w_2 \leq 0$	0

ていればパーセプトロンで表現することが出来たといえる。しかし、図 5.5 で示しているとおり分断する領域となる直線を引いても、●と○を 2 分するようにくくることが出来ない。曲線で 2 分する場合だと、●と○を 2 分するようにくくることができるが、単層のパーセプトロンは曲線を描くことが出来ないため XOR ゲートを表現することができないのである。

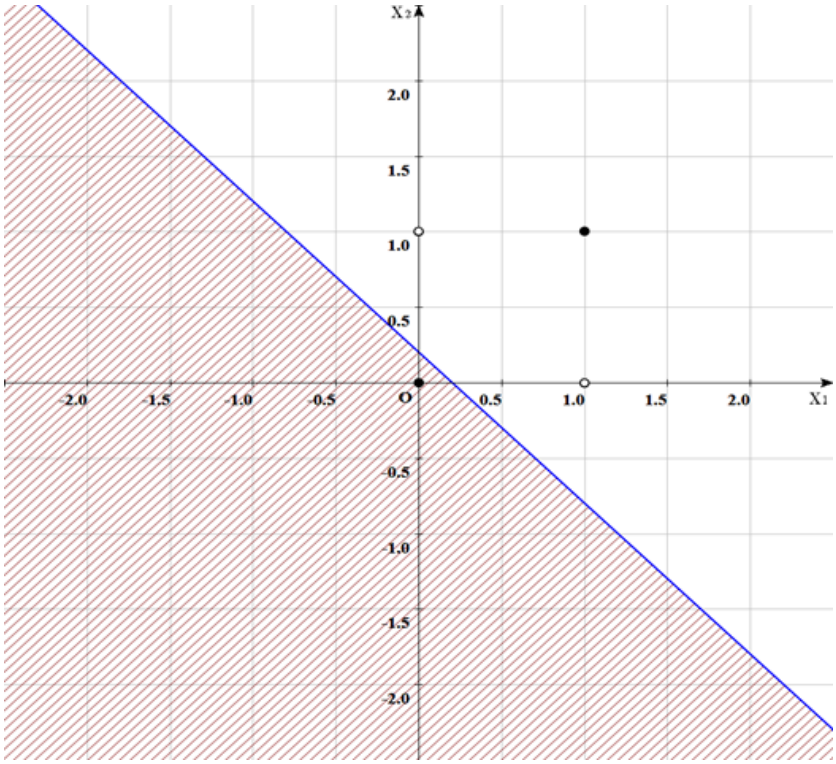


図 5.5　XOR ゲートが出力 0 となる範囲

単層のパーセプトロンでは XOR ゲートを表現することはできなかった。しかし、パーセプトロンには層を重ねることで単一では表せないものでも表現できる利点がある。このいくつも層を重ねるパーセプトロンを多重パーセプトロンと呼ぶ。この多重パーセプトロンを用いてもう一度 XOR

ゲートについて考えていく。今回ではこれまで作った既存のゲートを組み合わせることで、XOR ゲートを表現する方法を用いる。図 5.6 が AND,NAND,OR ゲートの回路記号である。

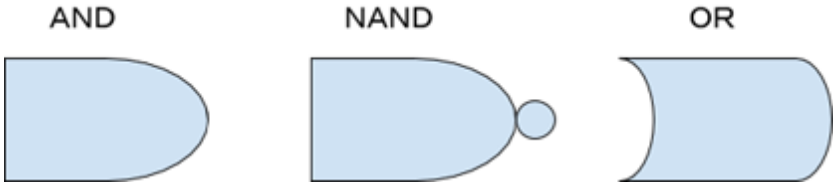


図 5.6　論理回路記号

これらの倫理回路記号を組み合わせて出来た XOR ゲートの回路図が図 5.7 である。x_1、x_2 が NAND、OR ゲートの入力信号、s_1 が NAND ゲート、s_2 が OR ゲート、y が AND ゲートの出力信号である。

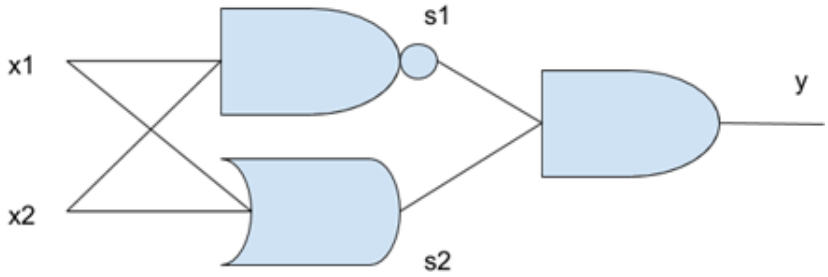


図 5.7　論理回路記号

この回路図で XOR ゲートが表現されているか確かめるため、図 5.7 の回路図の真理値表を作成する。真理値表は表 5.9 のようになる。前述したとおり s_1 は NAND ゲートの出力結果、s_2 は OR ゲートの出力結果を表しているため、これら 2 つを AND ゲートへ入力すれば最終的な出力結果が出る。AND ゲートは 2 つの入力が 1 のときのみに 1 を出力するので、$s_1 = 1$、$s_2 = 1$ のとき、$y = 1$ となる。ここで、x_1、x_2、y の値を着目しながら表 5.9 を見ていくと、入出力が XOR ゲートの真理値表と同じなので、多層パーセプトロンを用いて表現できていることが分かる。

表 5.9　複数の論理ゲートを用いた XOR ゲートの真理値表

x_1	x_2	s_1	s_2	y
0	0	1	0	0
0	1	1	1	1
1	0	1	1	1
1	1	0	0	0

上記より、倫理回路記号を複数個組み合わせることで XOR を表現することに成功した。パーセプトロンは層を重ねることで、より柔軟な表現が可能になったと言ってよい。次の節では、このパーセプトロンを更に発展させた形式を説明する。

（※文責: 高橋大介）

5.2 ニューラルネットワーク

5.1 パーセプトロンでは、パーセプトロンについて説明した。パーセプトロンは、複雑な関数であってもそれを表現できる可能性を秘めている。例えば、コンピュータが行うような複雑な処理も、パーセプトロンで表現することができる。しかし、期待する入力と出力を満たす適切な重みを決めるのは、人の手で行わなければならない。

ニューラルネットワークは、適切な重みパラメータをデータから自動で学習できるという性質を持つため、重みを機械的に決めることができる。本節では、斎藤 [] を参考にニューラルネットワークの概要について述べる。また、データから重みパラメータを学習する方法については 5.3 で述べる。

（※文責: 居附未紗）

5.2.1 ニューラルネットワークとは

ニューラルネットワークを図で表すと、図 5.8 のようになる。同じ縦の列の丸（ニューロン）をまとめて「層」という。ここで、一番左の層を「入力層」一番右の層を「出力層」中間の層を「中間層」と呼ぶ。また、中間層は「隠れ層」とも呼ばれ、入力層や出力層とは異なり、人の目に見えない事を表している。

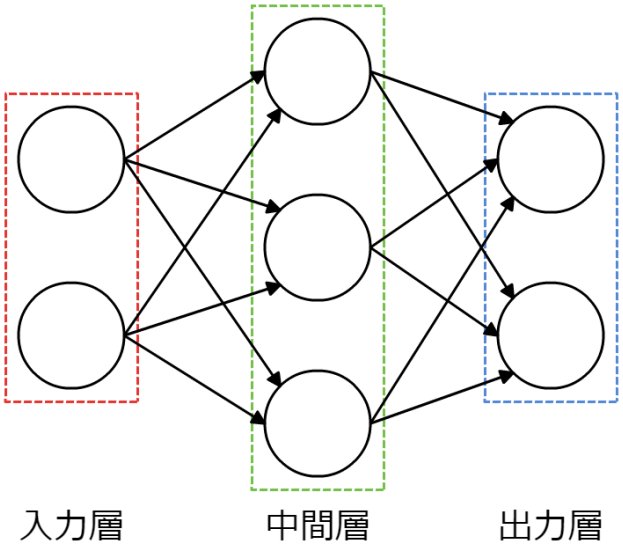


図 5.8

尚、本報告書では、入力層から出力層に向かって順に、「第 0 層」「第 1 層」「第 2 層」と呼ぶこととする。図 5.8 では、第 0 層は入力層、第 1 層は中間層、第 2 層は出力層に対応している。

また、図 5.8 を見ると、パーセプトロンと同じような形をしていることがわかる。ニューラルネットワークとパーセプトロンのニューロンの繋がり方は同じである。

（※文責: 居附未紗）

5.2.2 信号の伝達

ニューラルネットワークにおける信号の伝達方法の前に、パーセプトロンの伝達方法について復習する。まずは、図 5.9 のようなパーセプトロンについて考える。

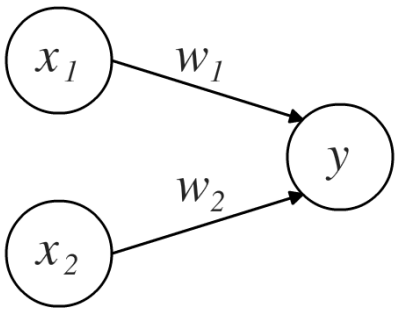


図 5.9　パーセプトロン

このパーセプトロンは x_1 と x_2 の 2 つの信号を受け取り、y を出力する。これを数式で表すと式 (5.3) で表される。

$$y = \begin{cases} 0 & (b + w_1 x_1 + w_2 + x_2 \leq 0) \\ 1 & (b + w_1 x_1 + w_2 x_2 > 0) \end{cases} \tag{5.3}$$

ここで、b はバイアス、w_1、w_2 は重みである。このパーセプトロンの動作は、x_1 と x_2 の 2 つの信号がニューロンの入力となり、それら 2 つの信号にそれぞれ重みが乗算され、次のニューロンに送信される。次のニューロンでは、それら重み付けされた信号とバイアスの和が計算され、その和が 0 を超えたら 1 を出力し、そうでなければ 0 を出力する。

では、式 (5.3) をよりシンプルな形に書き換える。式 (5.3) を簡略化するためには、0 を超えたら 1 を出力し、そうでなければ 0 を出力するという動作を、1 つの関数で表す。ここでは $h(x)$ という新しい関数を導入し、式 (5.3) を次の式 (5.4)、(5.5) のように書き換える。

$$y = h(b + w_1 x_1 + w_2 x_2) \tag{5.4}$$

$$h(x) = \begin{cases} 0 & (x \leq 0) \\ 1 & (x > 0) \end{cases} \tag{5.5}$$

式 (5.4) は入力信号の総和が $h(x)$ という関数によって変換され、その変換された値が出力 y になるということを表している。そして、式 (5.5) で表される $h(x)$ 関数は、入力が 0 を超えたら 1 を返し、そうでなければ 0 を返す。よって、式 (5.3) と式 (5.4)、(5.5) は同じことを行っている。

式 (5.5) に登場した $h(x)$ のような入力信号の総和を出力に変換する関数を、一般に「活性化関数（activation function）」と呼ぶ。名前が意味するように、活性化関数は入力信号の総和がどのように活性化（発火）するかということを決定する役割がある。

では、さらに式 (5.4) を書き換える。式 (5.4) では、重み付きの入力信号の総和を計算し、そして、その和が活性化関数によって変換されるという 2 段階の処理を行っている。そのため、式 (5.4) を丁寧に書くとすれば、次の 2 つの式に分けることができる。

$$a = b + w_1 x_1 + w_2 x_2 \tag{5.6}$$

$$y = h(a) \tag{5.7}$$

式 (5.6) では、重み付き入力信号とバイアスの総和を計算し、それを a とする。そして、式 (5.7) において、a が $h()$ で変換され y が出力され、という流れになる。

さて、ここまでニューロンは 1 つの○で図示してきたが、式 (5.6)、(5.7) を明示的に表すとすれば、図 5.10 のように表すことができる。

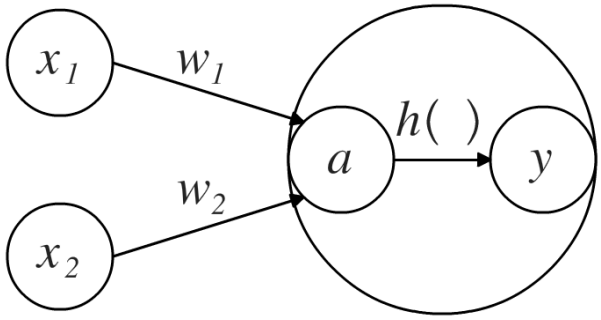


図 5.10

図 5.10 で表されるように、これまでのニューロンの○の中に、活性化関数によるプロセスを明示的に図示している。つまり、重み付き信号の和の結果が a というノードになり、そして、活性化関数 $h()$ によって y というノードに変換される、ということがはっきり示されている。

それでは続いて、活性化関数について詳しく見ていくことにする。この活性化関数が、パーセプトロンからニューラルネットへ進むための架け橋となる。

（※文責: 居附未紗）

5.2.3 活性化関数

式 (5.5) で表される活性化関数は、閾値を境にして出力が切り替わる関数で、「ステップ関数」や「階段関数」と呼ばれる。そのため、パーセプトロンでは活性化関数にステップ関数を利用していると言うことができる。つまり、活性化関数の候補としてたくさんある関数の中で、パーセプトロンは「ステップ関数」を採用しているということである。では、活性化関数にステップ関数以外を用いるとどうなるか。実は、活性化関数をステップ関数から別の関数に変更することで、ニューラルネットワークの世界へ進むことができる。では早速、ニューラルネットで利用される活性化関数を紹介する。

ニューラルネットでよく用いられる活性化関数の 1 つは、式 (5.8) で表される「シグモイド関数 (sigmoid function)」である。

$$h(x) = \frac{1}{1 + exp(-x)} \tag{5.8}$$

式 (5.8) の $exp(-x)$ は e^{-x} を意味し、e はネイピア数の 2.7182…の実数を表す。

ステップ関数とシグモイド関数にはいくつかの共通点があるが、そのなかでも重要な共通点は、両者ともに「非線形関数」であることである。「線形関数」とは、出力が入力の定数倍になるような関数のことを言い、そのため、線形関数はまっすぐな 1 本の直線となる。一方、非線形関数は、線形関数のように単純な 1 本の直線ではない関数を指す。シグモイド関数は曲線、ステップ関数は階段のような折れ曲がった曲線で表されるため、ともに非線形な関数である。

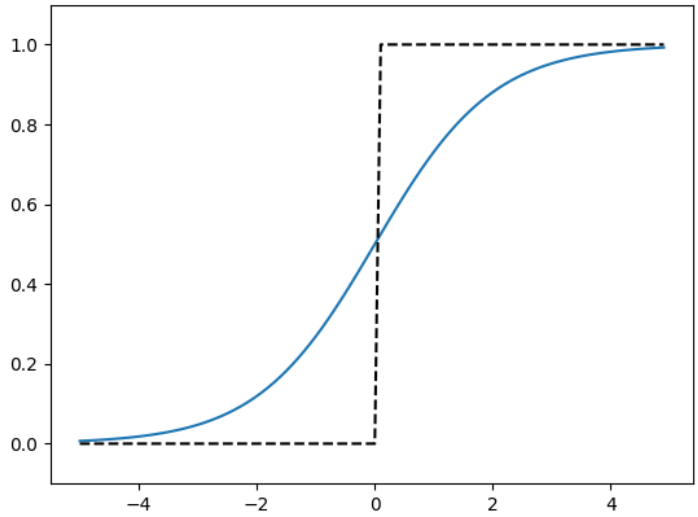


図 5.11　シグモイド関数（実線）とステップ関数（点線）

ニューラルネットワークでは、活性化関数に非線形関数を用いる必要がある。これは言い換えると、活性化関数には線形関数を用いてならないということである。

その理由は、活性化関数に線形関数を用いると、ニューラルネットワークの層を深くすることの意味がなくなるためである。

線形関数の問題点は、どんなに層を深くしても、それと同じことを行う隠れ層のないネットワークが必ず存在する、という事実に起因する。たとえば、線形関数である $h = cx$（c は定数）を活性化関数として、$y(x) = h(h(h(x)))$ を行う計算を 3 層のネットワークに対応させて考える。この計算は $y = c \times c \times c \times x$ の掛け算を行う。しかし、同じことは $y(x) = ax$（ただし $a = c^3$）の 1 回の掛け算、つまり、隠れ層のないネットワークで表現できる。この例のように、線形関数を用いた場合、多層にすることの利点を生かすことができない。そのため、層を重ねることの恩恵を得るためには、活性化関数に非線形関数を使う必要がある。

（※文責: 居附未紗）

5.2.4　ニューラルネットワークの出力層の設計

機械学習の問題は、「分類問題」と「回帰問題」に大別できる。分類問題とは、データがどのクラスに属するかという問題である。たとえば、人の写った画像から、その人が男性か女性かを分類するような問題が分類問題に相当する。一方、回帰問題は、ある入力データから、（連続的な）数値の予測を行う問題である。たとえば、人の写った画像から、その人の体重を予測するような問題が回帰問題である。

ニューラルネットワークは、分類問題と回帰問題の両方に用いることができる。しかし、分類問題と回帰問題のどちらに用いるかで、出力層の活性化関数を変更する必要がある。一般的に、回帰問題では恒等関数を、分類問題ではソフトマックス関数を使う。

恒等関数は、入力をそのまま出力する。そのため、出力層で恒等関数を用いるときは、入力信号をそのまま出力するだけになる。

一方、分類問題で使われるソフトマックス関数は、次の式で表される。

$$y_k = \frac{exp(a_k)}{\sum_{i=1}^{n} exp(a_i)} \tag{5.9}$$

ここでは、出力層が全部で n 個あるとして、k 番目の出力 y_k を求める計算式を表している。式に示すように、ソフトマックス関数の分子は入力信号の指数関数、分母はすべての入力信号の指数関数の和から構成されている。

なお、ソフトマックス関数を図で表すと図 5.12 のようになる。式 (5.9) の分母と図 5.12 からわかるように、ソフトマックスでは出力の各ニューロンが、すべての入力信号から影響を受ける。

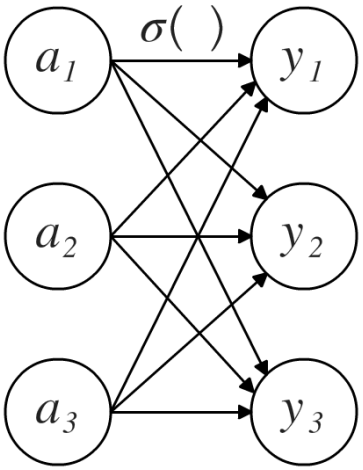


図 5.12

ソフトマックス関数の出力は 0 から 1.0 の間の実数になる。また、出力の総和は $y_1 + y_2 + \cdots + y_n = \frac{\sum_{i=1}^{n} exp(a_i)}{\sum_{i=1}^{n} exp(a_i)}$ より、1 になる。この性質は、ソフトマックス関数の重要な性質であり、この性質からソフトマックス関数の出力を「確率」として解釈でき、問題に対して確率的（統計的）な対応ができるようになる。

最後に、出力層のニューロンの数は、解くべき問題に応じて、適宜決める必要がある。クラス分類を求める問題であれば、出力層の数は分類したいクラスの数に設定するのが一般的である。

（※文責: 居附未紗）

5.3　ニューラルネットワークの学習

本節では、ニューラルネットワークの学習、特に勾配法を用いた最適な重みパラメータの取得方法についての説明を行う。

はじめに、ニューラルネットワークの学習の流れは、損失関数の値を減らすために各重みの勾配を求め、重みパラメータを勾配方向に微小量だけ更新することを繰り返す。その後、訓練モデルで使用していないデータである、テストデータを使用して訓練したモデルの実力、すなわち汎化能力を評価する。これがニューラルネットワークの学習の流れである。この節では最適なパラメータを探索するために、ニューラルネットワークの性能の悪さを指す指標である損失関数を用いて探索する。主な損失関数としてここでは 2 乗和誤差と交差エントロピー誤差の 2 つを説明する。

二乗和誤差は

$$E = \frac{1}{2}\sum_{k=0}^{k}(y_k - t_k)^2 \tag{5.10}$$

と表されるものである。式 (5.10) の y_k はニューラルネットワークの出力、t_k は訓練データ、k はデータの次元数を表している。二乗和誤差の最大値と最小値を求めるために、式 (5.10) をグラフで表すと下記の 4 通りのグラフになる（ただし、簡単化のため $k = 1$、t_k を正解グラフとする）。

1. $t_0 = t_1 = 0$ のとき、
   $E = \frac{1}{2}{y_1}^2 + \frac{1}{2}{y_0}^2$
   となり、図 5.13 ようなグラフになる。

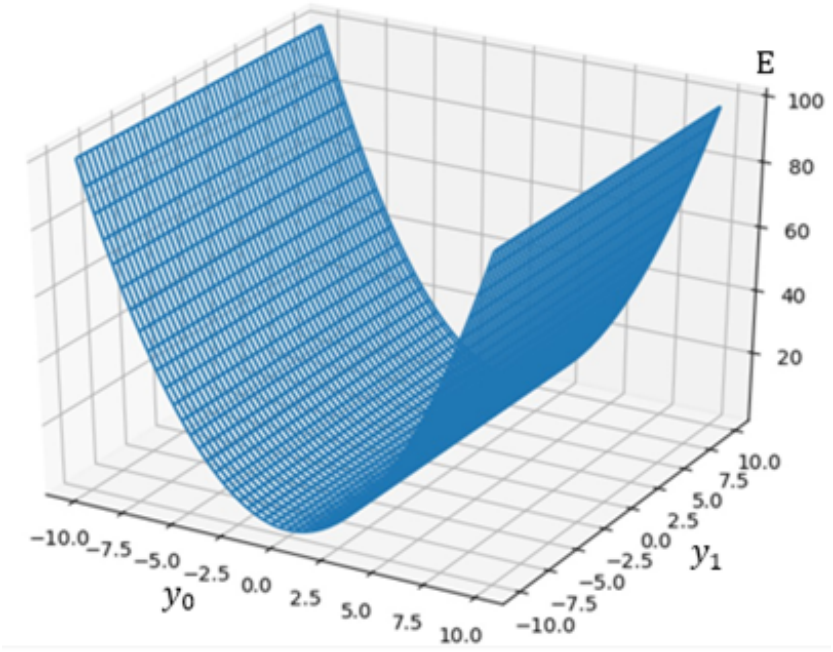


図 5.13

2. $t_0 = 1$、$t_1 = 0$ のとき、
   $E = \frac{1}{2}{y_1}^2 - y_1 + \frac{1}{2}{y_0}^2 + \frac{1}{2}$
   となり、図 5.14 のグラフのようになる。
3. $t_0 = 0$、$t_1 = 1$ のとき、
   $E = \frac{1}{2}{y_1}^2 + \frac{1}{2}{y_0}^2 - y_0 + \frac{1}{2}$
   となり、図 5.15 のグラフのようになる。
4. $t_0 = t_1 = 1$ のとき、
   $E = \frac{1}{2}{y_1}^2 - y_1 + \frac{1}{2}{y_0}^2 - y_0 + 1$
   となり、図 5.16 のようなグラフになる。

以上、4 通りのグラフから二乗和誤差の最大値は $+\infty$ に発散し、最小値は $0(y_k = t_k)$ であることが分かる。

交差エントロピー誤差は、

$$E = -\sum_{k=0}^{k} t_k \log y_k \tag{5.11}$$

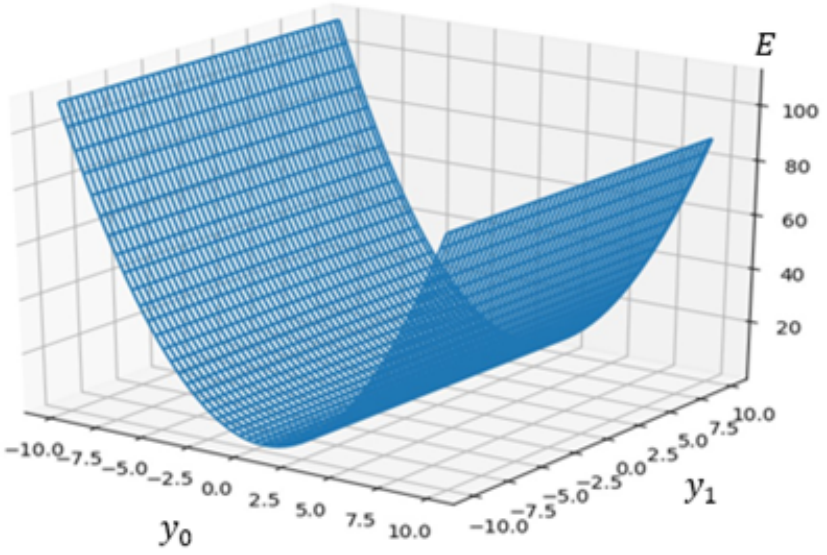


図 5.14

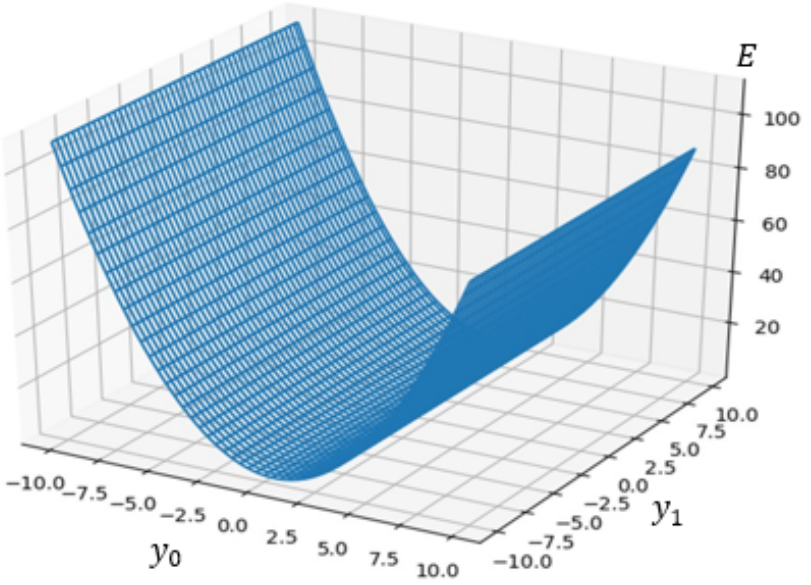


図 5.15

で表されるものである。式 (5.11) の y_k はニューラルネットワークの出力、t_k は正解ラベル、k はデータの次元数を表している。交差エントロピー誤差の最大値と最小値を求めるために、式 (5.11) をグラフで表すと下記のとおりのグラフになる（ただし、$k = 1$ とする）。

1. $t_0 = t_1 = 0$ のとき、
   $$E = 0$$
   となり、図 5.17 のようなグラフになる。
2. $t_0 = 1$、$t_1 = 0$ のとき、
   $$E = -\log y_0$$
   となり、図 5.18 のグラフのようになる。
3. $t_0 = 0$、$t_1 = 1$ のとき、

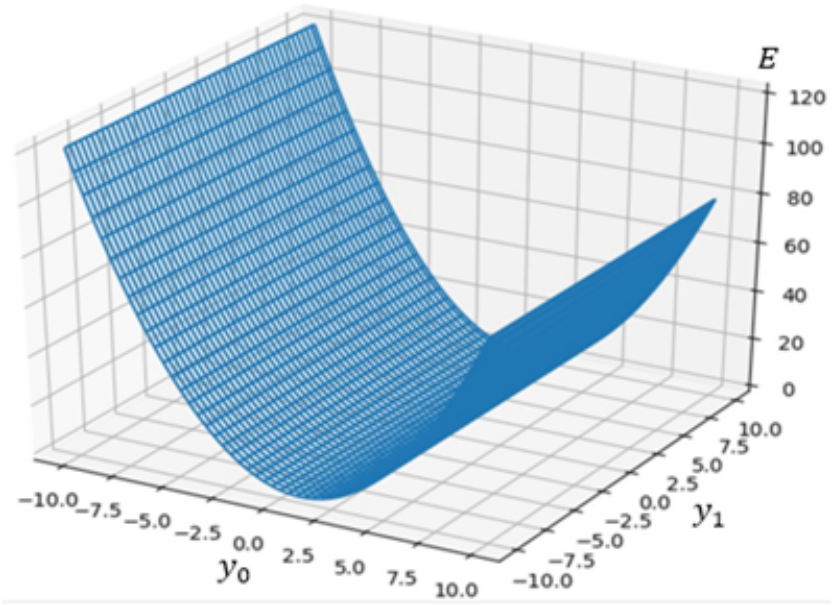


図 5.16

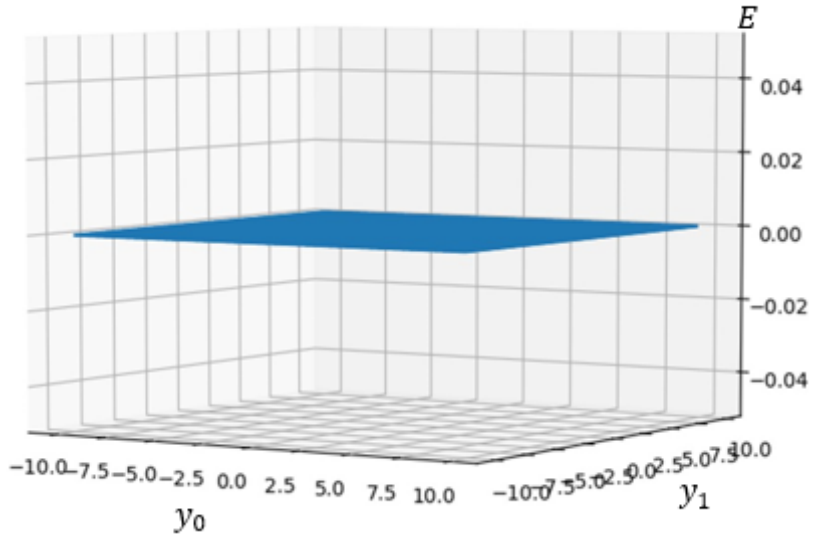


図 5.17

$$E = -\log y_1$$

となり、図 5.19 のグラフのようになる。

4. $t_0 = t_1 = 1$ のとき、

$$E = -(\log y_0 + \log y_1)$$

となり、図 5.20 のグラフのようになる。

これら 4 通りのグラフから、交差エントロピー誤差の最大値は $+\infty$ に発散し、最小値は $-\infty$ に発散することが分かる。ただし、ニューラルネットワークの出力である y_k にはソフトマックス関数などを用いるので、交差エントロピー誤差の最小値は $0(y_k = t_k, 0 \leq y_k \leq 1)$ となる。

ニューラルネットワークの学習において損失関数の値を最小に近づけることによって最適な重みパラメータを獲得する。

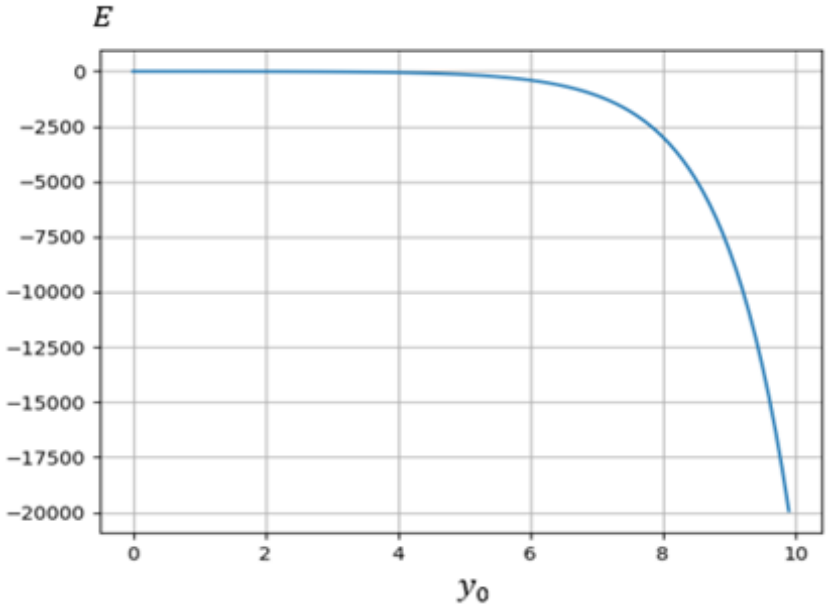


図 5.18

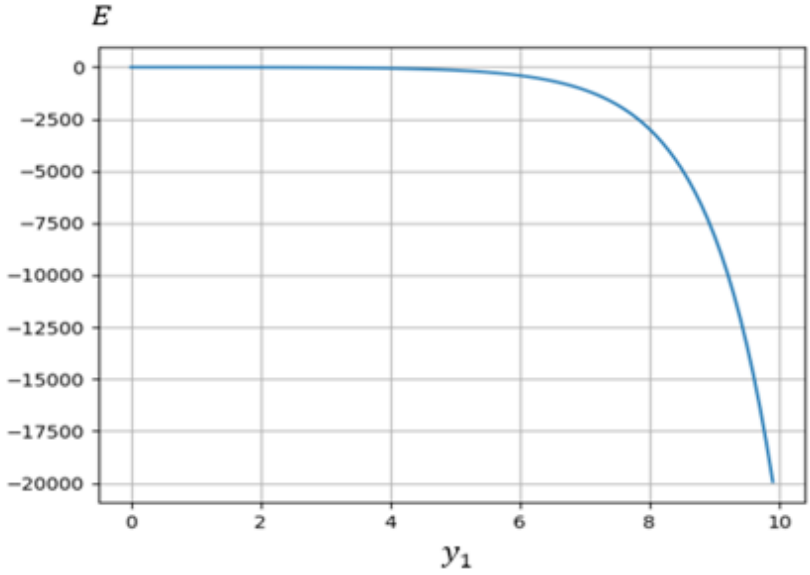


図 5.19

ここで、最適な重みパラメータを獲得するにあたって、この章では勾配法を用いて損失関数の値を最小に近づけることを考える。まず、ニューラルネットワークの学習における勾配について説明する。ニューラルネットワーク学習における勾配とは、重みパラメータに関する損失関数のことである。また、勾配とはすべての変数に対する偏微分をベクトルとしてまとめたのである。数式で表すと、

$$\frac{\partial E}{\partial w_{ik}} \tag{5.12}$$

である。これは重み w がどれだけ変化すると、損失関数 E がどれだけ変化するのかを表した式である。また、i は何番目の入力なのか、k は i 番目の入力に関する、何個目の重みなのかを表している。

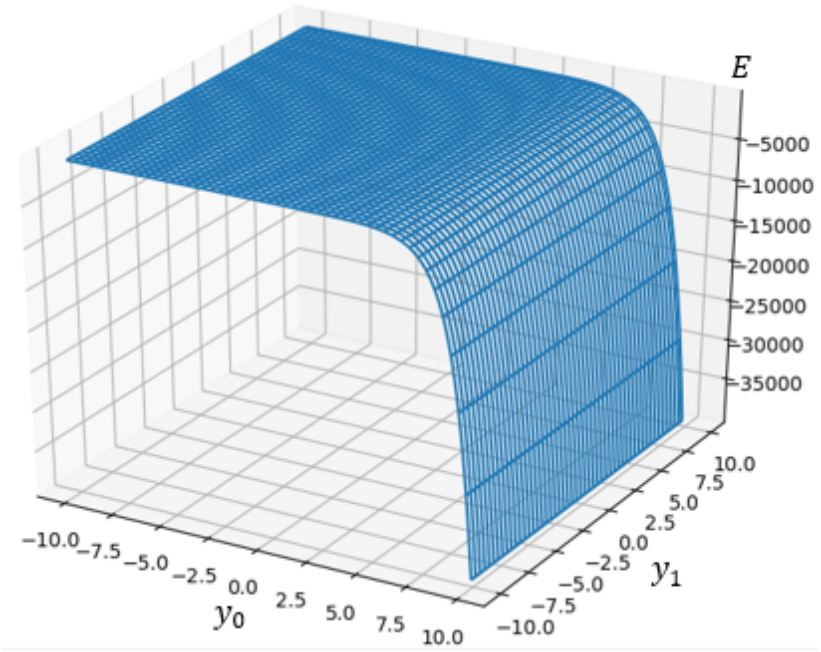


図 5.20

この式 (5.12) を用いて、重みを更新する。重みの更新式は、

$$w_{ik}^{n+1} = w_{ik}^{n} - \delta \frac{\partial E}{\partial w_{ik}} \tag{5.13}$$

で表される。n はある時点での更新回数を表し、$(n+1)$ はその次の時点での更新回数を表している。δ は学習率を表している。学習率とは、1 回の学習でどれだけ学習すべきか、どれだけパラメータを更新するかを決める値のである。学習率は人間側が主導で決める必要があり、その値は 1 より小さな値を設定するのが普通である。また、学習率は小さすぎる値であると計算回数が増えてしまい、大きすぎると、適切な重みパラメータにならないので、いろいろな値を試しながら設定する必要がある（6 章で私たちのプロジェクトで実際に用いた数式と計算過程を載せているので、そちらを参照)。

この章では損失関数の値を減らすために各重みの勾配を求め、重みパラメータを勾配方向に微小量だけ更新することを繰り返す。その後、訓練モデルで使用していないデータである、テストデータを使用して訓練したモデルの実力、すなわち汎化能力を評価する、というニューラルネットワーク学習の流れと、それに伴って使用する式についての説明をした。次章では、本章で学んだ勾配法によるニューラルネットワークの学習を用いて作成したシステムの概要について説明をする。

（※文責: 竹内俊二）

5.4　ディープニューラルネットワークの問題点と解決策

本節では、ニューラルネットワークの発展形のディープニューラルネットワークについてとその実装における問題点と解決策を記述する。今回、本プロジェクトでは反映することが出来なかったが、システムの改善案として記述しておくことにする。

入力層と出力層のみのニューラルネットワークではとても簡単なパターンの学習しか出来な

かった。そこで、より複雑な学習を可能にするためのモデルを考えたい。そのモデルの実装のためには、

- 隠れ層内のニューロン数を増やす。
- 隠れ層の数を増やす。

というアプローチを用いることが適切だと考えられる。

後者のアプローチは、層の数を増やしている。つまり、層を深くすると言える。このような深い層をもつネットワークを「ディープニューラルネットワーク」と呼ぶ。また、ディープニューラルネットワークを学習する方法のことを総じて「ディープラーニング」または、「深層学習」と呼ぶ。では、実際に上の 2 つのアプローチを用いると、予測精度はどうなるでしょうか。本プロジェクトでは実験していませんが、次のことがわかっています。

隠れ層内のニューロン数を増やす場合は、一定のニューロン数まで増やすと予測精度が増加しなくなる。そこで、計算実行時間などを考慮して最適なニューロン数を設定することが必要となってくる。隠れ層の数を増やす場合は、層が増えるごとに予測精度が減少してしまう。層が多くなれば複雑なパターンを表現することが出来るはずだが、実装してみると、学習が出来ていないことが分かる。

何故このようなのことが起きるのか、また、その解決策についてを述べる。

5.4.1 勾配消失問題

単純に層を増やすだけでは学習が上手くいかない原因として、「勾配消失問題」が挙げられます。では、何故勾配が消失してしまうのでしょうか。

隠れ層が二層のニューラルネットワークについて巣籠 [23] に習って考えてみる。

入力層を x、隠れ層を $h(1)$,$h(2)$、出力層を y として、かつ、それぞれのニューロン数を I, J, K, L とする。また、重み行列を W, V, U、バイアスベクトルを $\boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}$ とする。活性化関数はシグモイド関数 $\sigma(\cdot)$ とする。

それぞれの出力を考えると、

$$h_{(1)} = \sigma(Wx + \boldsymbol{b}) \tag{5.14}$$
$$h_{(2)} = \sigma(Vh_{(1)} + \boldsymbol{c}) \tag{5.15}$$
$$y = softmax(Uh_{(2)} + \boldsymbol{d}) \tag{5.16}$$

と表すことが出来る。さらに、以下のように定義する。

$$p = Wx + \boldsymbol{b} \tag{5.17}$$
$$q = Vh_{(1)} + \boldsymbol{c} \tag{5.18}$$
$$r = Uh_{(2)} + \boldsymbol{d} \tag{5.19}$$

重み $W = \begin{pmatrix} w_1 & w_2 & \cdots & w_J \end{pmatrix}^T$ に対する勾配を考えると、誤差関数 E_n を w_j で微分したら良いので、

$$\frac{\partial E_n}{\partial w_j} = \frac{\partial E_n}{\partial p_j}\frac{\partial p_j}{\partial w_j} = \frac{\partial E_n}{\partial p_j}x$$

と表すことが出来る。なので、$\frac{\partial E_n}{\partial p_j}$ についてかんがえることにする。

偏微分の連鎖律を使うと、

$$\frac{\partial E_n}{\partial p_j} = \sum_{k=1}^{K} \frac{\partial E_n}{\partial q_k} \frac{\partial q_k}{\partial p_j} \tag{5.20}$$

$$= \sum_{k=1}^{K} \frac{\partial E_n}{\partial q_k} (\sigma'(p_j) v_{kj}) \tag{5.21}$$

となる。これで、各勾配を求めることが出来るが、以上の式は 3 層のネットワークの場合であるため、4 層のネットワークを考える必要がある。そのためには、再度偏微分の連鎖律を使用したら良い。

$$\frac{\partial E_n}{\partial q_k} = \sum_{l=1}^{L} \frac{\partial E_n}{\partial r_l} \frac{\partial r_l}{\partial q_k} \tag{5.22}$$

$$= \sum_{l=1}^{L} \frac{\partial E_n}{\partial r_l} (\sigma'(q_k) u_{lk}) \tag{5.23}$$

と表せられるが、$\frac{\partial E_n}{\partial r_l}$ は出力部分なので、

$$\frac{\partial E_n}{\partial r_l} = -(t_l - y_l)$$

となる。つまり、ネットワークの誤差にあたるので、

$$\delta_j := \frac{\partial E_n}{\partial p_j} \tag{5.24}$$

$$\delta_k := \frac{\partial E_n}{\partial q_k} \tag{5.25}$$

$$\delta_l := \frac{\partial E_n}{\partial r_l} \tag{5.26}$$

とおくと、

$$\delta_j = \sum_{k=1}^{K} \sigma'(p_j) v_{kj} \delta_k \tag{5.27}$$

$$= \sum_{l=1}^{L} \sum_{k=1}^{K} (\sigma'(q_k) \sigma'(p_j)) u_{lk} v_{kj} \delta_l \tag{5.28}$$

という、4 層のネットワークにおける誤差逆伝播法の式が得られる。隠れ層が増えるごとにこれを繰り返すことになることは明白である。

ここで、$\sigma'(\cdot)$ というシグモイド関数の微分が出てくるが、これについて考えると $\sigma(x) = \frac{1}{1+\mathrm{e}^{-1}}$ であることから、

$$\sigma'(x) = \frac{\mathrm{e}^{-1}}{(1+\mathrm{e}^{-1})^2} \tag{5.29}$$

$$= \frac{1}{1+\mathrm{e}^{-1}} \frac{\mathrm{e}^{-1}}{1+\mathrm{e}^{-1}} \tag{5.30}$$

$$= \frac{1}{1+\mathrm{e}^{-1}} (1 - \frac{1}{1+\mathrm{e}^{-1}}) \tag{5.31}$$

$$= \sigma(x)(1-\sigma(x)) \tag{5.32}$$

さらに、これの右辺を平方完成すると、

$$\sigma'(x) = \sigma(x)(1-\sigma(x)) \tag{5.33}$$

$$= -(\sigma(x) - \frac{1}{2})^2 + \frac{1}{4} \tag{5.34}$$

よって、$\sigma'(x)$ は $\sigma(x) = \frac{1}{2}$ のとき、つまり $x = 0$ のとき最大値 $\frac{1}{2}$ をとることがわかる。

このことより、誤差逆伝播法の式には最大値 $\frac{1}{4}$ の $\sigma'(x)$ をかけられていることが分かる。さらに、隠れ層の数だけ式の中に $\sigma'(x)$ が出現するので、隠れ層の数が増えれば増える程に誤差逆伝播法の式全体の最大値が急激に減少してしまう。つまり、誤差の値が急激に 0 に近付いてしまうことがわかる。これが勾配消失問題の原因だと考えられる。

よって、この問題を回避するためには「微分しても値が極端に小さくならない活性化関数」を用いることが適切であると考えられる。以下より、その条件を満たすために提唱された活性化関数の例をだしていく。

隠れ層における活性化関数は「受けとる値が小さければ小さい値を返し、値が大きければ大きな値を返す」という条件を満たしていれば良い。では、その条件を満たす損失関数として、どのようなものが用いられているのかの例を以下に箇条書きする。

1. 双曲線正接関数 (tanh)
2. ReLU
3. Leaky ReLU

まず、双曲線正接関数はハイパボリックタンジェントとも呼ばれ、

$$\tanh(x) = \frac{\mathrm{e}^{x} - \mathrm{e}^{-x}}{\mathrm{e}^{x} + \mathrm{e}^{-x}}$$

で定義されている。これを微分した場合は

$$\tanh'(x) = \frac{4}{(\mathrm{e}^{x} + \mathrm{e}^{-x})^{2}}$$

であり、最大値は 1 であるため、シグモイド関数に比べて勾配が消失しにくいことが分かる。

次に、ReLU は、シグモイド関数や双曲線正接関数に起こりうる問題を改善したものと言える。

$$f(x) = \max(0, x)$$

で表され、入力が 0 より大きいか否かで値を変える。また、これを微分すると、

$$f'(x) = \begin{cases} 1 & (x > 0) \\ 0 & (x \leq 0) \end{cases}$$

となる。最大値 1 であり、計算も簡単なため良さそうに見える。しかし、一度不活性になるとその後もずっと不活性になってしまうといったことが起こってしまう。

最後に、Leaky ReLU は LReLU とも呼ばれ、ReLU を改良したものである。

$$f(x) = \max(\alpha x, x)$$

で表され、微分しても、

$$f'(x) = \begin{cases} 1 & (x > 0) \\ \alpha & (x \leq 0) \end{cases}$$

と、$f'(x)$ が 0 になってしまうことがないため、安定した学習が可能になると考えられる。しかしながら、実装してみると効果が出るかどうかはわからず、どうしてそうなるのかも分かっていない。

ここでは、3 つのアプローチについて述べたが、Parametric ReLU(PLeRU) や Randomized ReLU(RReLU) など、ReLU をベースにした損失関数が提案されている。その学習にあった活性化関数を選択することが大切である。

（※文責: 吉田周平）

5.5　リカレントニューラルネットワーク

時系列データを予測する、すなわち時間の概念をニューラルネットワークに取り入れるには、過去の状態をモデル内で保持しておかなければならない。現在に対する過去からの（目に見えない）影響を把握しておかなければならないので、これを「過去の」隠れ層として定義する必要がある。この考え方を最も単純に表したグラフィカルモデルが図 5.21 になる。層自体は「入力層 - 隠れ層 - 出力層」と一般的なニューラルネットワークと変わらないが、時刻 t における入力 $\boldsymbol{x}(t)$ に加え、時刻 $(t-1)$ における隠れ層の値 $h(t-1)$ を保持しておき、それも時刻 t における隠れ層に伝える点がこれまでと大きく異なる。時刻 t の状態を $t-1$ の状態として保持しフィードバックさせるため、過去の隠れ層の値 $h(t-1)$ には再帰的に過去の状態がすべて反映されていることになる。これがリカレント（＝再帰型）ニューラルネットワークと呼ばれる理由である。

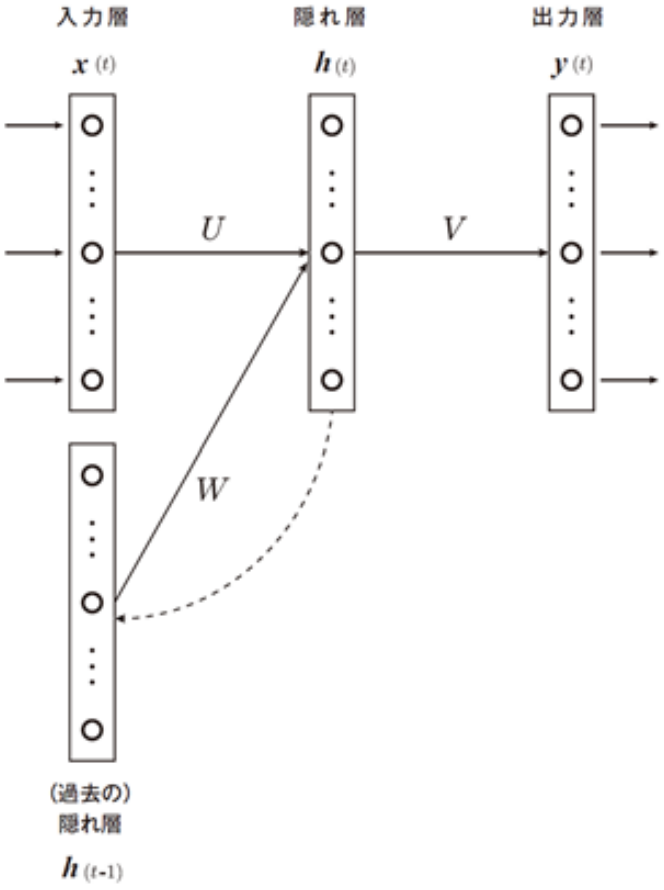


図 5.21

過去の隠れ層が加わったものの、モデルの出力を表す式は特別難しくなるわけではない。素直に式を書いていくと、まず隠れ層は

$$\boldsymbol{h}(x) = f(U\boldsymbol{x}(t) + W\boldsymbol{h}(t-1) + \boldsymbol{b}$$

と表せ、また出力層は

$$\boldsymbol{y}(t) = g(V\boldsymbol{h}(t) + \boldsymbol{c})$$

となる。ここで、$f(\cdot)$、$g(\cdot)$ は活性化関数、$\boldsymbol{b}$、$\boldsymbol{c}$ はバイアスベクトルである。隠れ層の式に過去から順伝播してくる項 $W\boldsymbol{h}(t-1)$ が付くが、それ以外は一般的なニューラルネットワークと違いはない。そのため、各モデルのパラメータも誤差逆伝播法により最適化できるはずである。誤差関数を $E := E(U, V, W, \boldsymbol{b}, \boldsymbol{c})$ として、それぞれのパラメータに対する勾配を考えてみる。隠れ層、出力層の活性化前の値をそれぞれ下記の $\boldsymbol{p}(t)$、$\boldsymbol{q}(t)$ で定義しておく。

$$\begin{aligned}\boldsymbol{p}(t) &:= U\boldsymbol{x} + W\boldsymbol{h}(t-1) + \boldsymbol{b} \\ \boldsymbol{q}(t) &:= V\boldsymbol{h}(t) + \boldsymbol{c}\end{aligned}$$

すると、隠れ層、出力層における誤差項

$$\begin{aligned}\boldsymbol{e}_h(t) &:= \frac{\partial E}{\partial \boldsymbol{p}(t)} \\ \boldsymbol{e}_o(t) &:= \frac{\partial E}{\partial \boldsymbol{q}(t)}\end{aligned}$$

に対して、

$$\begin{aligned}\frac{\partial E}{\partial U} &= \frac{\partial E}{\partial \boldsymbol{p}(t)}\left(\frac{\partial \boldsymbol{p}(t)}{\partial U}\right)^T = \boldsymbol{e}_h(t)\boldsymbol{x}(t)^T \\ \frac{\partial E}{\partial V} &= \frac{\partial E}{\partial \boldsymbol{q}(t)}\left(\frac{\partial \boldsymbol{q}(t)}{\partial V}\right)^T = \boldsymbol{e}_o(t)\boldsymbol{h}(t)^T \\ \frac{\partial E}{\partial W} &= \frac{\partial E}{\partial \boldsymbol{p}(t)}\left(\frac{\partial \boldsymbol{p}(t)}{\partial W}\right)^T = \boldsymbol{e}_h(t)\boldsymbol{x}(t-1)^T \\ \frac{\partial E}{\partial \boldsymbol{b}} &= \frac{\partial E}{\partial \boldsymbol{p}(t)} \bigodot \frac{\partial \boldsymbol{p}(t)}{\partial \boldsymbol{b}} = \boldsymbol{e}_h(t) \\ \frac{\partial E}{\partial \boldsymbol{c}} &= \frac{\partial E}{\partial \boldsymbol{q}(t)} \bigodot \frac{\partial \boldsymbol{q}(t)}{\partial \boldsymbol{c}} = \boldsymbol{e}_o(t)\end{aligned}$$

が求められ、誤差項のみを考えればよいことが分かる。過去の隠れ層という概念が追加されても、モデルの最適化を考える上でのアプローチは変わらない。

ただし、誤差関数 E については、sin 波の予測などを考える場合に少し注意が必要である。これまで誤差関数として用いてきた交差エントロピー誤差関数は、出力層における活性化関数 $g(\cdot)$ がソフトマックス関数（あるいはシグモイド関数）において求められる形だったが、sin 波の予測では、出力が確率ではなくそのままの値を用いる必要があるため、$g(x) = x$、出力の式は

$$\boldsymbol{y}(t) = V\boldsymbol{h}(t) + \boldsymbol{c}$$

という線形活性の式になる。このときの誤差関数について考えなければならない。とは言うものの、これは難しく考える必要はなく、誤差関数は最小化すべき「モデルの予測値 $y(t)$ と正解の値 $t(t)$ との誤差」を表す関数だという前提を踏まえると、例えば

$$E := \frac{1}{2}\sum_{t=1}^{T} \left\|\boldsymbol{y}(t) - \boldsymbol{t}(t)\right\|^2$$

という二乗誤差関数（squared error function）を与えればよいことになる。

リカレントニューラルネットワークの誤差を求める際、気を付けなければならない点が 1 つある。例えば誤差関数を二乗誤差関数として考える場合、誤差 $\boldsymbol{e}_h(t)$、$\boldsymbol{e}_o(t)$ は、

$$\begin{aligned}\boldsymbol{e}_h(t) &= f'(\boldsymbol{p}(t)) \bigodot V^T\boldsymbol{e}_o(t) \\ \boldsymbol{e}_o(t) &= g'(\boldsymbol{q}(t)) \bigodot (\boldsymbol{y}(t) - \boldsymbol{t}(t))\end{aligned}$$

で与えられることになる。これらの式自体は正しいが、リカレントニューラルネットワークでは、ネットワークの順伝播で時刻 $t-1$ における隠れ層の出力 $h(t-1)$ を考えたため、逆伝播の際も $t-1$ における誤差を考える必要がある。

リカレントニューラルネットワークのモデルを時間軸で展開することを考える。例えば下の図は時刻 $(t-2)$ の入力 $x(t-2)$ まで展開したものになるが、誤差 $\boldsymbol{e}_h(t)$ は $\boldsymbol{e}_h(t-1)$ に逆伝播し、さらに $\boldsymbol{e}_h(t-1)$ は $\boldsymbol{e}_h(t-2)$ に逆伝播する。このように、順伝播の際は $h(t)$ が $h(t-1)$ の再帰関係式で表されたのと同様、逆伝播の際は $\boldsymbol{e}_h(t-1)$ を $\boldsymbol{e}_h(t)$ の式で表す必要がある。このとき、誤差は時間をさかのぼって逆伝播していることになるため、これを Backpropagation Through Time と呼び、BPTT と略記する。

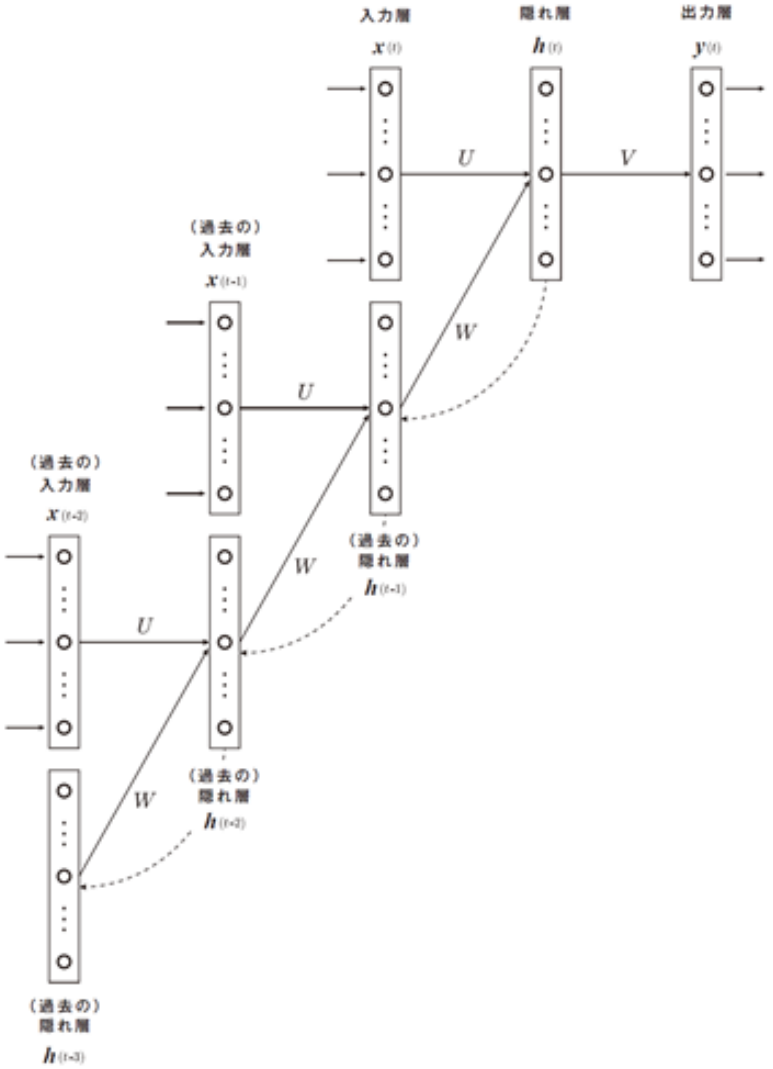


図 5.22

BPTT という手法の名前は付いていますが、考えるべきは $\boldsymbol{e}_h(t-1)$ を $\boldsymbol{e}_h(t)$ の式で表すことである。$(t-1)$ における誤差は、

$$\boldsymbol{e}_h(t-1) = \frac{\partial E}{\partial \boldsymbol{p}(t-1)}$$

であるので、再帰関係式を求めると、

$$\begin{aligned}\boldsymbol{e}_h(t-1) &= \frac{\partial E}{\partial \boldsymbol{p}(t)} \bigodot \frac{\partial \boldsymbol{p}(t)}{\partial \boldsymbol{p}(t-1)} \\ &= \boldsymbol{e}_h(t) \bigodot \left(\frac{\partial \boldsymbol{p}(t)}{\partial \boldsymbol{h}(t-1)} \frac{\partial \boldsymbol{h}(t-1)}{\partial \boldsymbol{p}(t-1)} \right) \\ &= \boldsymbol{e}(t) \bigodot (Wf'(\boldsymbol{p}(t-1)))\end{aligned}$$

が得られる。よって、再帰的に $\boldsymbol{e}_h(t-z-1)$ と $\boldsymbol{e}_h(t-z)$ は

$$\boldsymbol{e}_h(t-z-1) = \boldsymbol{e}_h(t-z) \bigodot (Wf'(\boldsymbol{p}(t-z-1)))$$

の関係で表すことができる。これによりすべての勾配が計算できることが分かり、各パラメータの更新式は、

$$\begin{aligned}U(t+1) &= U(t) - \eta \sum_{z=0}^{\tau} \boldsymbol{e}_h(t-z)\boldsymbol{x}(t-z)^T \\ V(t+1) &= V(t)^{\eta} \boldsymbol{e}_o(t)\boldsymbol{h}(t)^T \\ W(t+1) &= W(t) - \eta \sum_{z=0}^{\tau} \boldsymbol{e}_h(t-z)\boldsymbol{h}(t-z-1)^T \\ \boldsymbol{b}(t-1) &= \boldsymbol{b}(t) - \eta \sum_{z=0}^{\tau} \boldsymbol{e}_h(t-z) \\ \boldsymbol{c}(t+1) &= \boldsymbol{c}(t) - \eta \boldsymbol{e}_o(t)\end{aligned}$$

となる。この τ がどれくらいの過去までさかのぼって時間依存性を見るかを表すパラメータであるため、理想的には $\tau \to +\infty$ とすべきである。しかし、現実には勾配が消失（あるいは爆発）してしまうことを防ぐため、せいぜい $\tau = 10$〜100 くらいに設定するのが一般的である。

（※文責: 近藤渚紗）

第 6 章　機械学習を用いた作曲システム

6.1　教師データの編曲

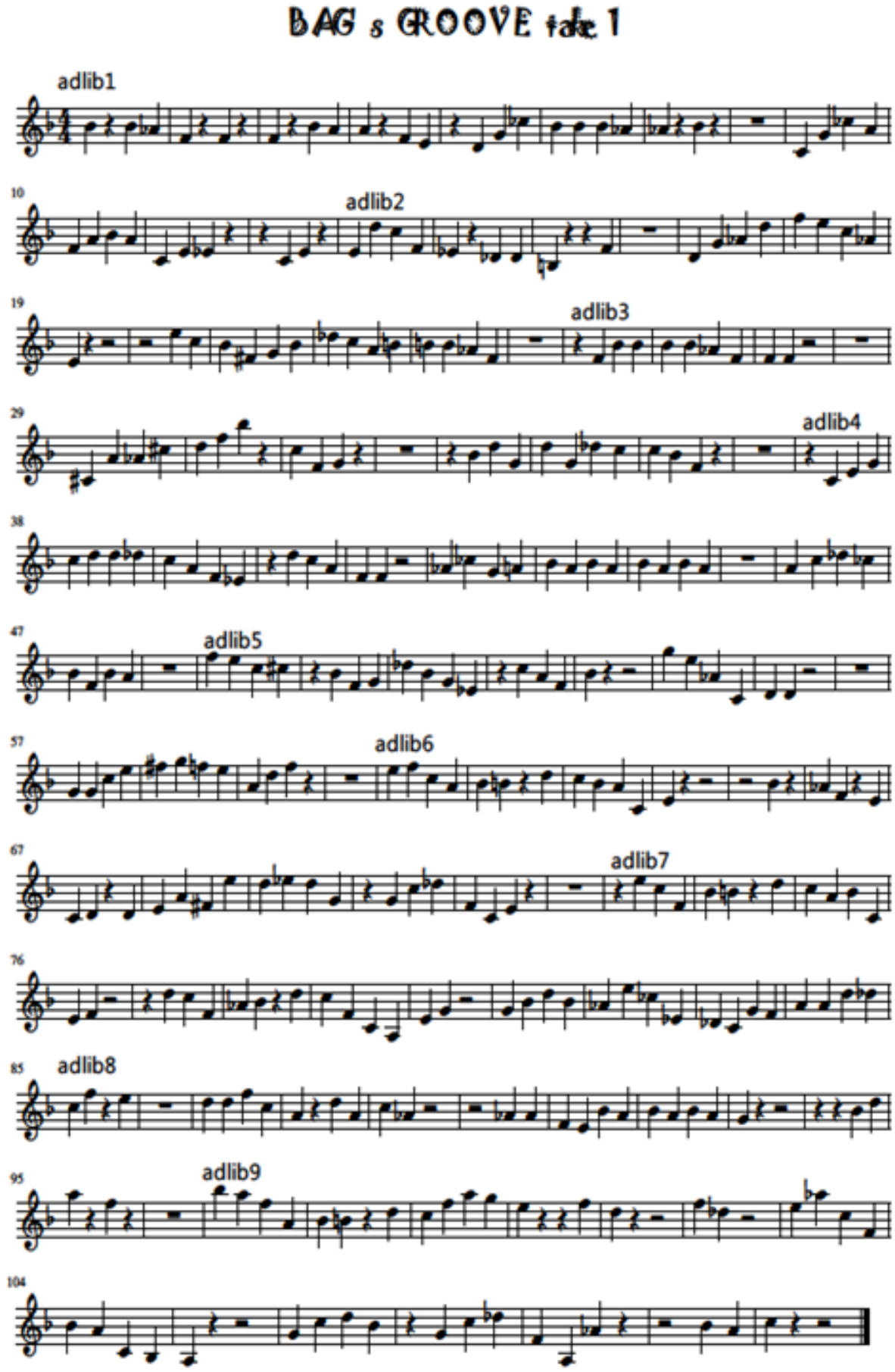
作曲システムを作成するにあたって、課題曲の編曲という作業がある。そこで、課題曲について述べておきたいと思う。

課題曲は”Bags’ Groove” を使用した。”Bags’ Groove” はビブラフォン奏者のミルト・ジャクソンが作曲した F ブルースの曲である。「Bags’」はジャクソンの目の下の隈を表している。私たちはマイルス・デイヴィスがアドリブを演奏したテイク 1 とテイク 2 の楽譜を入手し、課題曲とした。

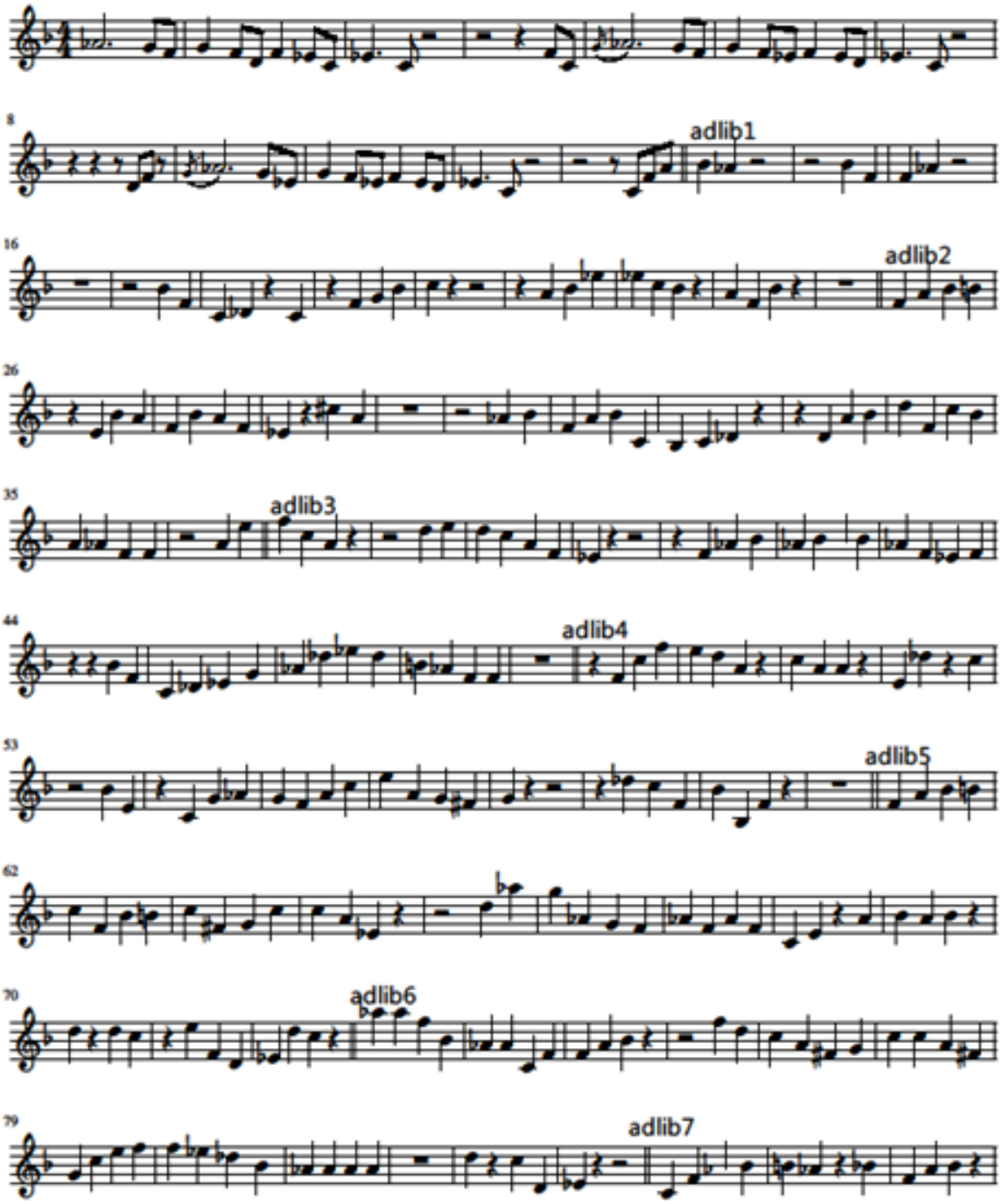
マイルス・デイヴィスは 1926 年に生まれたアメリカのジャズトランペット奏者である。ファンや評論家からは「ジャズの帝王」、「モダン・ジャズの帝王」と呼ばれる。2015 年にはマイルスがトランペットを手放し、創作活動を休止した空白の 5 年間を題材とした「MILES AHEAD マイルス・デイヴィス 空白の 5 年間」という映画がアメリカで公開された。

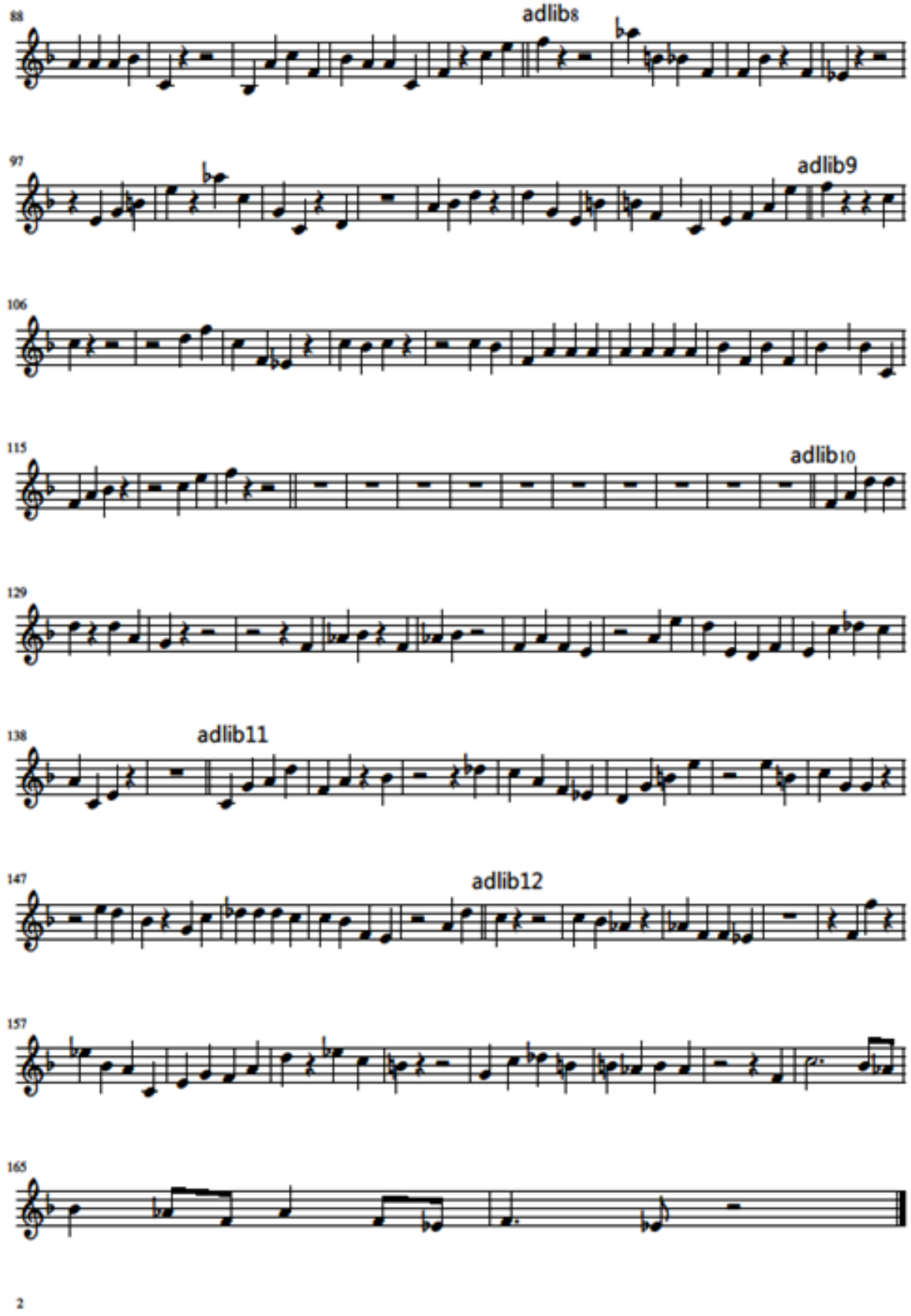
”Bags’ Groove” のテイク 1 のアドリブは 108 小節あり、12 小節を一区切りとするブルースの特徴から、9 つのアドリブを得た。アドリブは様々な音価、リズムで構成されている。それらの音符をすべて四分音符と四部休符に編曲した。

編曲した結果は以下の通りである。


BAG s GROOVE take 1
adlib1
adlib2
adlib3
adlib4
adlib5
adlib6
adlib7
adlib8
adlib9

BAG's GROOVE take2




88
adlib8
97
adlib9
106
115
adlib10
129
138
adlib11
147
adlib12
157
165
2

以上の楽譜を教師データ用に、編曲した楽譜の音符に図 6.1 のように対応する番号を付けた。

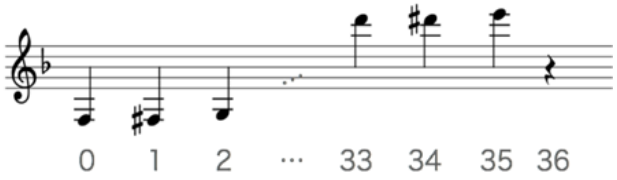


図 6.1

例として、音が第 1 間の F（ファ）であった場合、この音符の番号は 12 である。この番号を利用した。そして、今の音が $(x-1)$ であるとして、次の音 (x) にはどの音がくるのかを対応表を作成して確率を求めた。

（※文責: 北田明日香）

6.2 機械学習を用いたシステムの概要

5 章のようにして機械学習を用いた作曲システムを作成した。

1. 使用するオクターブ分をすべての音に対しての最適な重みを学習する。
2. 初期音を与えて、その音に対応した重みを使用して次の音を出力する。
3. 2 を繰り返して、必要な音の数を得た時にプログラムを終了する。

使用したプログラム言語は Python3 系である。図 6.2 にフローチャートを示す。

今回は、音を図 6.1 のようにして数値化した。休符は 36 とした。この数値を a に入れる。この a を基にして入力信号入力信号 x_{ai} を生成する。

$$x_{ai} = \begin{array}{r} \\ 0 \\ 1 \\ \vdots \\ 35 \\ 36 \end{array} \begin{bmatrix} & & & & \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

また、学習の最初は $w_{a,k,i}$ を初期化する必要がある。

$$\sum_i w_{a,k,i} = 1.0$$

上記の式を条件とした。

次に、重みと入力の積の和を y_k とし、次のニューロンに入れる。今回のニューラルネットワークの活性化関数として式 (6.1) で表されるソフトマックス関数に y_k を入れ、活性化させる。S_k は a の音の次に k の音が出てくる確率の値である。

$$S_k = Softmax(y_k) = \frac{e^{y_k}}{\sum_i e^{y_i}} \tag{6.1}$$

また、ニューラルネットワークの性能の悪さを表す損失関数には交差エントロピー誤差を用いた。交差エントロピー誤差 E は式 (6.2) で表され、この E の値を小さくしていくことを学習の目

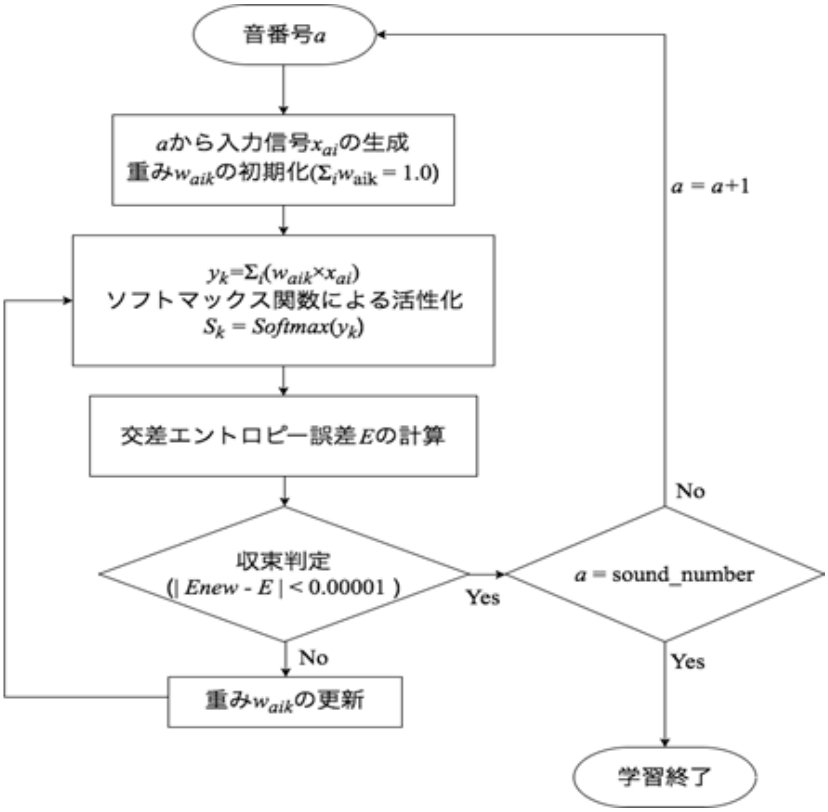


図 6.2　フローチャート

標とする。読み込んだ音の次に出る音を集計、それぞれを確率にしたものを教師データ t_k とする。

$$E = -\sum_k t_k \log S_k \tag{6.2}$$

重みをチューニングすることによって E の値を小さくしていく。重みの更新は勾配法を用いて行い、更新式は式 (5.5) によって表される。η は学習率とし、今回は $\eta = 0.01$ とした。また、学習回数は上限を 10,000 回とし、1 回前の損失関数の値の絶対値の差が 0.0001 以下ならば学習をやめることとする。(収束判定条件)

$$\begin{aligned} w_{a,k,i}^{new} &= w_{a,k,i} - \eta \frac{\partial E}{\partial w_{k,i}} \\ &= w_{a,k,i} - \eta \left\{ \frac{t_k \sum_{i \neq k} e^{y_i - y_k}}{\left(1 + \sum_{i \neq k} e^{y_i - y_k}\right)^2} \right\} x_i \end{aligned} \tag{6.3}$$

これらのプロセスをすべての音に対して行い、そうして求めた最適な各重みを使って音を出力させる。初期音は編曲した際に出てくる最初の音とし、その初期音の S_k の確率を用いて次の音を決定する。今回は 12 小節ですべて 4 分音符としたので 48 音得た時、プログラムを終了することにした。

（※文責: 近藤渚紗）

第 7 章　リカレントニューラルネットワークを用いたシステム

本プロジェクトでは、6 章で説明したアドリブ生成システムのほかに、リカレントニューラルネットワーク（以下、RNN）を用いたシステムの作成も行った。本章では、その RNN を用いたアドリブ生成システムについて述べる。

（※文責: 居附未紗）

7.1　教師データ

7.1.1　教師データに用いる曲の決定

教師データの楽曲は、「Miles Davis Omnibook: For C Instruments」[Hal Leonard Corp;Spi 版 (2015)] の 50 曲及び、「Charlie Parker Omnibook: For C Instruments. (Treble ClefVersion)」[Criterion Music Corp(1982)] の 60 曲の計 2 冊 110 曲の楽譜を MIDI ファイル化したものを用いた。次に、MIDI 化した楽曲名を表 7.1、7.2 に示す。

表 7.1 「Miles Davis Omnibook: For C Instruments」の収録曲

Airegin	All Blues
All of You	Au Privave
Bags' Groove	Bille's Bounce(Bill's Bounce)
Blue Haze	Budo
But Not for Me	Bye Bye Blackbird
Diane	Dig
Doxy	E.S.P.
Footprints	Four
Freddie Freeloader	A Gal in Calico
Green Haze	I Waited for You
I'll Remember April	If I Were a Bell
It Could Happen to You	It's Only a Paper Moon
Jeru	K.C. Blues
Love Me or Leave Me	Miles Ahead
Miles (Milestones)	My Funny Valentine
Oleo	On Green Dolphin Street
The Serpent's Tooth	Seven Steps to Heaven
Sippin' at Bells	So What
Solar	Some Day My Prince Will Come
Stablemates	Stella by Starlight
Stuff	Summertime
The Surrey with the Fringe on Top	The Theme
Trane's Blues	Tune Up
Walkin'	Well You Needn't(It's Over Now)
Woodyn' You	Yesterdays

表 7.2 「Charlie Parker Omnibook: For C Instruments. (Treble Clef Version)」の収録曲

AH-LEU-CHA(AH LEV CHA)	ANOTHER HAIRDO
ANTHROPOLOGY	AU PRIVAVE(No. 1)
AU PRIVAVE(No. 2)	BACK HOME BLUES
BALLADE	BARBADOS
BILLIE'S BOUNCE(BILL'S BOUNCE)	THE BIRD
BIRD GETS THE WORM	BLOOMDIDO
BLUE BIRD	BLUES(FAST)
BLUES FOR ALICE	BUZZY
CARD BOARD	CELERITY
CHASING THE BIRD	CHERYL
CHICHI	CONFIRMATION
CONSTELLATION	COSMIC RAYS
DEWEY SQUARE	DIVERSE
DONNA LEE	K. C. BLUES
KIM(No. 1)	KIM(No. 2)
KLAUN STANCE	KO KO
LAIRD BAIRD	LEAP FROG
MARMADUKE	MERRY-GO-ROUND
MOHAWK(No. 1)	MOHAWK(No. 2)
MOOSE THE MOOCHE	MY LITTLE SUEDE SHOES
NOW'S THE TIME(No. 1)	NOW'S THE TIME(No. 2)
ORNITHOLOGY	AN OSCAR FOR TREADWELL
PARKER'S MOOD	PASSPORT
PERHAPS	RED CROSS
RELAXING WITH LEE	SCRAPPLE FROM THE APPLE
SEGMENT	SHAWNUFF
SHE ROTE(No. 1)	SHE ROTE(No. 2)
SI SI	THRIVING FROM A RIFF
VISA	WARMING UP A RIFF
YARDBIRD SUTTE	STEEPLECHASE

表 7.1、7.2 中のすべての曲を幾つかの分類基準により分類し、システムに学習させる教師データの候補を選定した。はじめに、ジャンルがブルースであり、かつキーが F メジャーである楽曲をシステムに学習させる教師データの候補として選定した。選定した楽曲は、「Miles Davis Omnibook: For C Instruments」の”Sippin' at Bell”、「Charlie Parker Omnibook: For C Instruments. (Treble Clef Version)」の”AU PRIVAVE (No. 1)”、”BARBADOS”、 ”BILLIE'S BOUNCE(BILL'S BOUNCE)”、”BLUES FOR ALICE”、”NOW'S THE TIME (No. 1)”、”NOW'S THE TIME (No. 2)”、”SI SI” の計 8 曲である。次に、コード進行が単純であり「Miles Davis Omnibook: For C Instruments」及び、「Charlie Parker Omnibook: For C Instruments.(Treble Clef Version)」の双方に収録されている楽曲である”K.C. Blues” をシステムに学習させる教師データの候補として選定した。

上記の計 10 曲の教師データの候補について、2 章で述べた音楽理論に基づき、上昇スケール、下降スケール、上昇アルペジオ、下降アルペジオ、上昇コード外アルペジオ、下降コード外アルペジオ、上昇半音階、下降半音階、リズム、ゼクエンツの 10 種で構造の分析を行った。

本プロジェクトでは、コード外アルペジオを曲のスケールに使われている音かつ、連続する 3 音がそのキーのダイアトニックコードのいずれかに当てはまるものと定義した。上記の条件を満たしているかつその三音が連続して音程が上昇しているものを上昇コード外アルペジオ、下降しているものを下降コード外アルペジオと定義した。また、その曲のスケールで使われる音が 3 音連続で順に使われているかつ連続で上昇しているフレーズを上昇スケール、下降しているフレーズを下降スケールと定義した。また、加度 [17] より、一曲中に同じフレーズが繰り返し使用されている箇所をゼクエンツと定義し、2, 3 音の同じ短いフレーズが連続で使用されている箇所をリズムと定義した。楽譜を分析していく中でゼクエンツとリズムの出現回数が他の音型よりも少なかったため、ゼクエンツとリズムを同じ一つの項目として集計した。

その結果、”K.C. Blues” についてマイルス・デイヴィスとチャーリー・パーカーの 2 人奏者の間で構造に違いが出たため、システムに学習させる教師データを”K.C. Blues” に決定した。

ちなみに、”K.C. Blues” は今回模倣の対象にも選んでいるチャーリー・パーカーが作曲した C ブルースの曲である。「K.C.」はアメリカ合衆国ミズーリ州・カンザス州にまたがるカンザスシティという都市のことで、チャーリーは少年期にこの街で 3 年ほど暮らしている。

（※文責: 中島凱斗）

7.1.2 教師データの検定

システムの教師データとして決定したマイルス・デイヴィスとチャーリー・パーカーの K.C. Blues について、コルモゴロフ・スミルノフ検定を使用して違いがあるのか検定を行った。コルモゴロフ・スミルノフ検定とは、対応のない 2 標本 n_1、n_2 の母代表値に差があるかどうかを判定するときに使用する検定である。

コルモゴロフ・スミルノフ検定では検定統計量 T を求める。

$$T = 4(D_{max})^2 \times \frac{n_1 n_2}{n_1 + n_2} \tag{7.1}$$

ここで、式 (7.1) の (D_{max}) は $|d_i|$ のうちの最大値であり、$|d_i| = |P_{i1} - P_{i2}|$ である。また、P_{i1} は 1 群の i 階級の累積度数、P_{i2} は 2 群の i 階級の累積度数を表す。そして、T は $n_1, n_2 > 40$ のとき、自由度 $f = 2$ の X^2 分布に従う。

K.C Blues の検定を以下のような帰無仮説、対立仮説を設定して行った。

帰無仮説　H_0:マイルス・デイヴィス版 K.C. Blues とチャーリー・パーカー版 K.C Blues に差はない

対立仮説　H_1:マイルス・デイヴィス版 K.C. Blues とチャーリー・パーカー版 K.C Blues に差がある

以下の表 7.3 から $T \fallingdotseq 8.84$ であることが分かる。$\alpha = 0.05$ で自由度 $f = 2$ のときの棄却限界値は 5.99 である。$T > 5.99$ となるため、帰無仮説を棄却する。したがって、マイルス・デイヴィス版 K.C. Blues とチャーリー・パーカー版 K.C. Blues に差があると言える。

表 7.3:

	観察度数 (出現回数)		累積度数		累積相対度数		
階級 (音の種類)	第 1 群 (マイルス版) 出現回数	第 2 群 (チャーリー版) 出現回数	第 1 群	第 2 群	第 1 群	第 2 群	差
1	0	0	0	0	0.000	0.000	0.000
2	0	0	0	0	0.000	0.000	0.000
3	0	0	0	0	0.000	0.000	0.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
51	0	0	0	0	0.000	0.000	0.000
52	2	1	2	1	0.010	0.004	0.006
53	2	2	4	3	0.021	0.012	0.008
54	0	0	4	3	0.021	0.012	0.008
55	6	6	10	9	0.052	0.037	0.015
56	0	0	10	9	0.052	0.037	0.015
57	0	7	10	16	0.052	0.066	0.014
58	3	3	13	19	0.067	0.078	0.011
59	1	10	14	29	0.072	0.119	0.047
60	15	30	29	59	0.149	0.242	0.092
61	2	4	31	63	0.160	0.258	0.098
62	5	13	36	76	0.186	0.311	0.126
63	10	8	46	84	0.237	0.344	0.107
64	13	20	59	104	0.304	0.426	0.122
65	21	29	80	133	0.412	0.545	0.133
66	2	5	82	138	0.423	0.566	0.143
67	28	28	110	166	0.567	0.680	0.113
68	6	4	116	170	0.598	0.697	0.099
69	13	17	129	187	0.665	0.766	0.101
70	4	8	133	195	0.686	0.799	0.114
71	3	8	136	203	0.701	0.832	0.131
72	24	26	160	229	0.825	0.939	0.114
73	2	1	162	230	0.835	0.943	0.108
74	7	4	169	234	0.871	0.959	0.088

	観察度数 (出現回数)		累積度数		累積相対度数		
階級 (音の種類)	第 1 群 (マイルス版) 出現回数	第 2 群 (チャーリー版) 出現回数	第 1 群	第 2 群	第 1 群	第 2 群	差
75	2	3	171	237	0.881	0.971	0.090
76	8	3	179	240	0.923	0.984	0.061
77	4	2	183	242	0.943	0.992	0.049
78	1	2	184	244	0.948	1.000	0.052
79	9	0	193	244	0.995	1.000	0.005
80	0	0	193	244	0.995	1.000	0.005
81	1	0	194	244	1.000	1.000	0.000
82	0	0	194	244	1.000	1.000	0.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
128	0	0	194	244	1.000	1.000	0.000
合計	194	244					

（※文責: 北田明日香）

7.2 RNN を用いたシステムの入出力

リカレントニューラルネットを用いた機械学習による作曲システムを作るにあたって、ニューラルネットを用いた機械学習による作曲システムの入出力に問題があり、まずはこれを改善することを目指した。

以前の作曲システムでは、入力について、学習させる楽譜を四分音符、四分休符のみで構成するように編曲し、それを本プロジェクトが定義したものに沿って数値化し、その数値を入力しなくてはならなかった。また出力について、システムが学習して出力するのは本プロジェクトが定義した数値であり、出力がどのような曲になったか聴くには、その出力を定義にそって楽譜に書き起こさなくてはならなかった。出力は四分音符、四分休符のみだった。

これらの問題点を解決すべく、システムの入出力は MIDI データで行うこととした。MIDI とは”Musical Instruments Digital Interface” の略で、1981 年に策定された、電子楽器同士を接続するための世界共通規格のことである。特徴として、実際の音のデータではなくどのように演奏するかという文字列データであるため、オーディオデータに対してデータサイズが非常に小さいことが挙げられる。MIDI データで入出力が行えるなら、以前の作曲システムのような手間もかからず、四分音符、四分休符以外の音価の学習も行えると考えた。

MIDI データを入出力に用いるうえで、MIDI の中身の文字列データを理解する必要がある。『偏った DTM 用語辞典 カナインデックス』[35] を参考に文字列データを読み解いたところ、その内容は複雑だが、今回プロジェクトで扱う部分としては NoteOn、NoteOff、velocity、tick、resolution などといった信号がある。NoteOn は鍵盤を押すという信号で、MIDI の文字列データのなかで最も基本的といえる、発音を指示する文字列である。NoteOff は鍵盤を離すという信号で、発音をやめさせる指示をする文字列である。NoteOn、NoteOff には velocity、pitch、tick といった情報が付加されており、velocity は 0〜127 までの値で音の大きさ、pitch は 0〜127 までの音の高さ、tick は 0 以上の値で直前の指示からの経過時間を表す。この経過時間だが、resolution

という値によってデータごとに定義が異なる。resolution は分解能とも呼ばれ、四分音符をどれくらいの数字として扱うかをデータごとに定義するものである。480、960 などにするのがメジャーで、たとえば resolution 480 なら二分音符は 960、付点四分音符は 720、八分三連符は 160 といった具合に音の長さを定義できる。これらをまとめた一例として、resolution 480 で NoteOn、velocity 80、pitch 60、tick 0 なら直前の指示から間を開けずに 80 の大きさ、60 の高さの音を発する。この次に NoteOff、velocity 0、pitch 60、tick 480 なら四分音符分の時間を空けたのちに音は止まる。この連続で MIDI データは曲を演奏する。ただし、NoteOff の velocity は鍵盤を離す速度をあらわすが、多くの MIDI キーボードでは鍵盤を離す速度のセンサーを持たないことがあり、NoteOff の代わりに NoteOn、velocity 0 の信号を使用することが多い。こうする事により、鍵盤の押し離しだけならば NoteOn 信号のみになるため、ランニングステータスと呼ばれる第 1 バイト省略の手法によってデータ量が削減できるためである。今回我々が入力データとして楽譜から MIDI データに打ち直したものは、DTM ソフトの Cubase で出力しており、その MIDI データも NoteOn 信号のみで行われていた。この方法で構成される MIDI データでは、1 つの発音に対して 2 つの NoteOn 信号が使われることがわかる。

ここで注意するべきなのが、MIDI データは休符のデータを持たないことである。RNN を用いた機械学習による作曲システムでは、以前のニューラルネットを用いた機械学習による作曲システムと同じように休符も学習したい。そこで、休符もデータとして読み込めるように、入力の MIDI データを本プロジェクトが定義する文字列データへ変換することを目指す。

本プロジェクトが定義する文字列データは、MIDI データを楽譜に直したときの音符、休符ひとつ分を、0 と 1 の数字のみで構成される 140 字の文字列とした。その中身は、1 番目〜128 番目は pitch、129 番目は音符か休符か、130 番目〜140 番目は音の長さをあらわしている。1 番目〜128 番目について、例えば pitch の値が 60 であったなら、1 番目〜60 番目は全て 0 とし、61 番目を 1 とし、62 番目〜128 番目は全て 0 とする。このとき、pitch の値が 0〜127 であり、文字列データは 1 番目〜128 番目としているので、数字がひとつずれていることに気を付ける。129 番目について、この 140 字の文字列データが表したいものが音符であるなら 0 とし、休符であるなら 1 とする。休符である場合、音の高さはないので 1 番目〜128 番目は全て 0 であり、つまり、いかなる音を表す場合も 1 番目〜129 番目において 0 は 128 個、1 は 1 個である。130 番目〜140 番目について、前述した resolution の値を re とすると、130 131 132 133 134 135 136 137 138 139 140 番目についてそれぞれ $re \times 4$(全音符) $re \times 2$(二分音符) re(四分音符) $re \times \frac{2}{3}$(二拍三連符) $\frac{re}{2}$(八分音符) $\frac{re}{3}$(三連符) $\frac{re}{4}$(十六分音符) $\frac{re}{5}$(五連符) $\frac{re}{6}$(六連符) $\frac{re}{8}$(三十二分音符) $\frac{re}{16}$(六十四分音符) の長さを用いた音符、休符であることを表す。例えば全音符の長さをあらわす場合、130 番目〜140 番目の文字列は 10000000000 となる。また、付点四分音符の長さをあらわす場合、付点四分音符は四分音符と八分音符を足し合わせた長さの音符なので、文字列は 00101000000 となる。これらをまとめた一例として、入力された MIDI データから最も低い音を付点二分音符の長さで演奏するという情報を受け取った場合、文字列に変換すると、100……0(127 字分 0 を打つ)0(音符なので 0)01100000000(付点二分音符は二分音符と四分音符を足し合わせた長さの音符である) となる。次に、具体的なアルゴリズムについて説明する。

まず入力の MIDI データを読み込み、NoteOn の信号を探す。その NoteOn 信号について文字列に書き起こしたなら、次の NoteOn 信号を探す。その繰り返しで MIDI データの全ての NoteOn 信号を文字列に書き起こすまで続ける。

NoteOn の信号を見つけたとき、その信号のもつ velocity の値を調べる。velocity が 0 より大きいなら発音信号で、0 であるなら NoteOff の代わりに用いられている消音信号とわかる。発音

信号であった場合、まずは tick の値を調べる。tick の値が 0 より大きい場合、この直前に休符があったことがわかる。これは前述のように一般的な MIDI データでは、1 つの発音に対して 2 つの NoteOn 信号が用いられ、発音信号と消音信号が対になってできていることに由来する。今回は発音信号の tick の値が 0 より大きい場合を考えているため、直前の信号は必然的に消音信号であり、前の音が消音信号により止まってから tick の時間分経過し、今回の発音信号が実行されるため、MIDI データ上で扱われていない休符がここに存在するとわかる。なので、velocity が 0 より大きく、tick の値が 0 より大きい場合、まず休符の文字列を書き起こす。それが終わると今回は発音信号であるので、この発音信号についての音符の文字列を書き起こす。そのために pitch の値を変数に一時保存しておく。音符の長さについては今回の NoteOn 信号だけではわからないため、次の NoteOn 信号を探す。

次の NoteOn 信号は必然的に velocity 0 の消音信号である。tick の値を調べて、先ほど一時保存した pitch の値と今回の tick の値を用いて文字列を書き起こす。このとき、普通有り得ないが、tick の値が 0、すなわち直前の発音信号から間髪入れず今回の消音信号となった場合、文字列化不可能としてエラーを返すこととする。

以上のようにして入力データを本プロジェクトが定義する文字列データに変換した。

また、この文字列データで学習した作曲システムは同じ定義の文字列データを出力し、その文字列データを全く逆のアルゴリズムで MIDI データへ変換した。

これら入出力部分の実装は Python により行った。また MIDI データを扱ううえで、midi パッケージを用いた。

（※文責: 伊藤祐大）

7.3 RNN を用いたシステムの概要

今回作成したリカレントニューラルネットワーク (以下、RNN) システムの概要を説明する。

まず、音楽生成システムの前段階として、sin 波を予測する RNN を作成した。sin 波を予測するにあたって、入力データは sin(0)〜sin(2000) までのノイズ入りデータを使用した。また、中間層のノード数は 10、活性化関数はシグモイド関数であり、出力層の活性化関数は線形活性である。損失関数は二乗和誤差である。図 7.1 は学習の様子を示す。

横軸はエポック数、縦軸は損失関数の値である。エポックが進むに連れて誤差が少なくなっている、つまり学習している様子がわかる。こうして学習した RNN モデルに実際に sin 波を予測させてみる。まず、最初に教師データである sin 波の値を 3 つほど与えて、その後にどうなるかをグラフで見た。グラフを図 7.2 に示す。赤い点線は最初に与えた教師データを示し、青の点線が本物の sin 波、青の実線が予測した sin 波である。

sin(100) まで予測させてみた結果、ある程度の精度で予測できることがわかった。このシステムを用いて、音楽生成システムを作成することを試みた。

sin 波を予測したシステムを使用して音楽生成システムの作成を行う。sin 波を予測した時同様、隠れ層のノード数は 10、活性化関数はシグモイド関数である。まず最初に音程のみを予測するシステムを作成しようと試みたため、出力層の活性化関数は線形活性である。入力データは MIDI から取り込んだ note のナンバー一曲分である。損失関数は sin 波同様二乗和誤差を用いた。図 7.3 は「かえるの歌」を入力データとしたときの学習の様子である。

横軸はエポック数、縦軸は損失関数の値である。学習は成功しているように見える。学習した

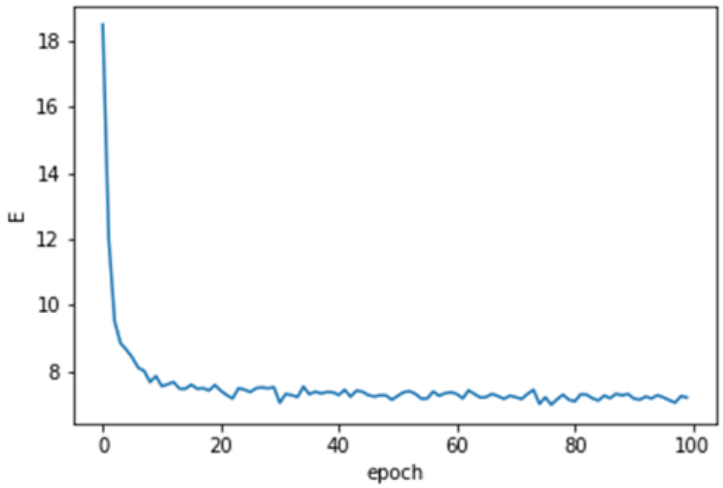


図 7.1　sin 波の学習

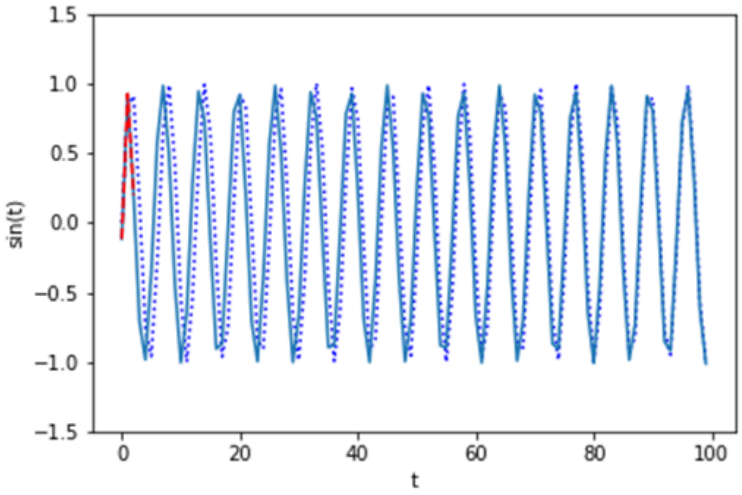


図 7.2　sin 波の予測

RNN モデルに教師データを 2 個与えて予測結果を見てみると、図 7.4 のようになった。

縦軸が note ナンバーに 0.01 をかけた値、横軸は時間である。最初はそのまま予測を行うが最終的に一つの音のみを出力してしまうことがわかった。また、他の曲で学習させてみても同様の結果が得られた。重みの初期値による失敗かと思い、何回か試してみたが、同じ結果しか得られなかったため、モデルの正当化の検証を行った。

まず最初に、与えるデータを”10, 50, 90, 50, 10, 50…” と続く 3 つの値しか無いものにした。結果としては図 7.5 のとおりになった。データは 0.001 をかけて与えている。

図 7.5 から、うまく出力できていないことがわかる。

また、2 値のものを与えてみても同様の結果を得た。この失敗する原因については今期では解明することができなかった。7.2 にて説明したように、音楽 MIDI データをワンホット型にして与えたが、それもやはり失敗した。

ところで、失敗の原因を探る中で、その理由の解明こそ至らなかったが、入力データをより小さ

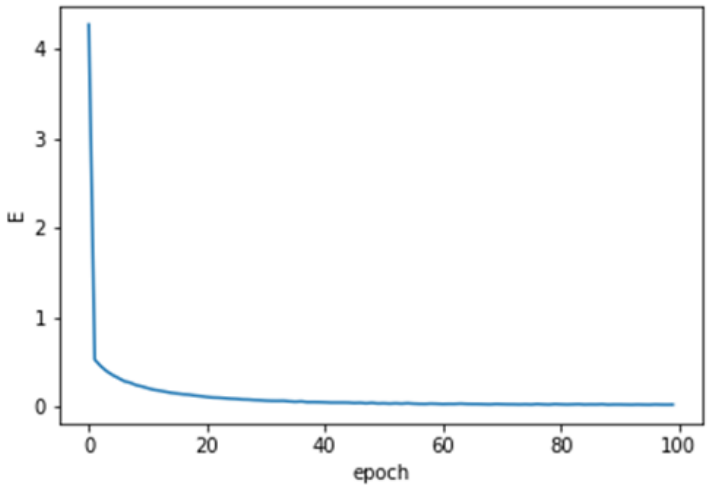


図 7.3　かえるの歌の学習

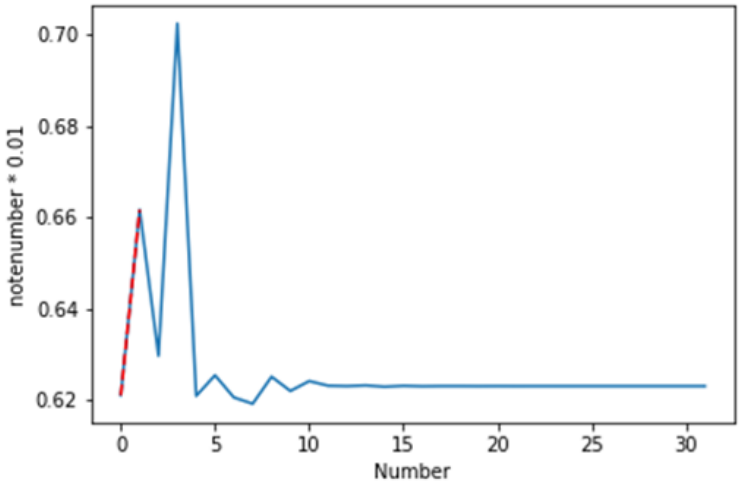


図 7.4　かえるの歌を学習させたシステムの予測

い値にしたり、0 に寄せたりすることにより、望ましい学習が行われやすいことに気が付いた。そこで、入力と出力を見直し、少し改良したシステムを用いて、再び学習を行った。

入力と出力について、音程のみの学習を目指した。まず、7.2 にて説明した、Midi データに用いられている pitch(0〜127) の値を与える。与えた pitch の値すべての平均を求め、それぞれの pitch の値から平均を引く。これにより、入力データは 0 を平均とした値となる。次に、最も大きな値と小さな値との差が、例として 20 を超える場合、それを下回るまですべての値を、例として 2 で割る。このようにして得た入力データを学習する。出力データは、入力時に割った数だけ、この場合なら 2 をかけ、平均した値を足すことにより得る。

まず、2 章で説明した基礎的な音型の中で、数字のうえで最も学習が簡単にみえる、半音階の学習を行う。入力は”60, 61, 62, 63, 64, ..., 71, 72, 71, 70, ..., 61, 60, 61, 62, ...” と、60 から 72、つまり 1 オクターヴのドからドの間を上昇、下降する半音階を用いた。隠れ層のノード数、活性化関数、損失関数などについては先ほど説明したものと同様である。図 7.6 は学習の様子である。

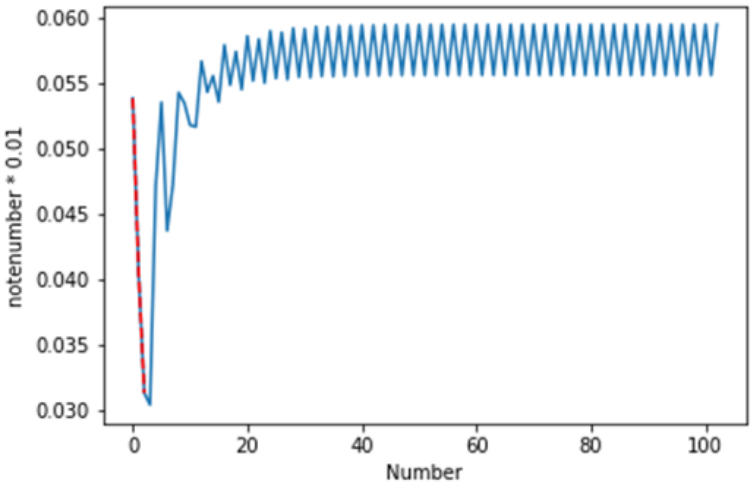


図 7.5　3 種の値のみのデータを与えた結果

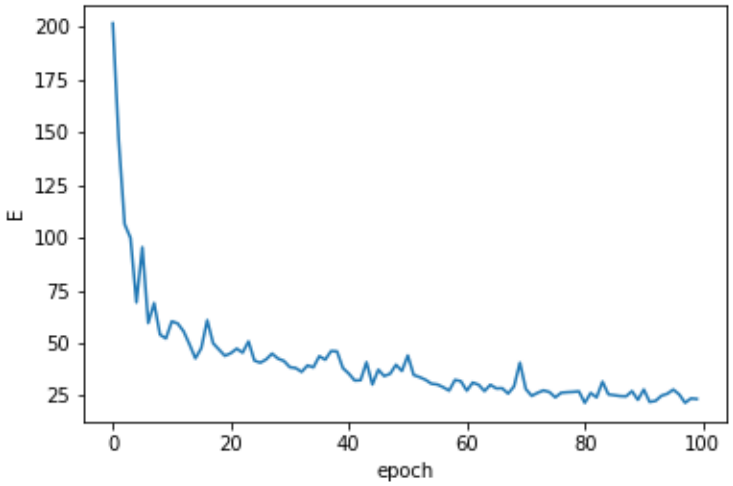


図 7.6　半音階の学習

横軸はエポック数、縦軸は損失関数の値である。学習は成功しているように見える。学習した RNN モデルに教師データを 2 個与えて予測結果を見てみると、図 7.7 のようになった。赤い点線は最初に与えた教師データを示し、青の点線が入力データ、青の実線が予測した出力である。

縦軸が note ナンバーから全体の平均を引いて調整した値、横軸は時間である。先ほどの結果のように 1 つの音を出し続けることはなく、ある程度の精度で予測できているようにみえる。入力データを楽譜に起こしたものと、出力を小数点第一位で四捨五入して整数にし楽譜に起こしたものとの、最初の数小節がそれぞれ図 7.8、図 7.9 である。

これらを比べると、音の上昇下降のタイミングなどはほぼ一致している。反面、同じ音が続いたり、半音以上の幅で音の動きがあったりなど、半音階の音型を模倣できたとは言い難い。

次に、スケールの学習を行う。入力は”60, 62, 64, 65, 67, 69, 71, 72, 74, 72, 71, 69, 67, 65, 64, 62, 60, 62, 64...”と、ドから 1 オクターブ上のレの間を、C メジャースケールの音のみを用いて上昇、下降するスケールを用いた。隠れ層のノード数、活性化関数、損失関数などについては先ほど

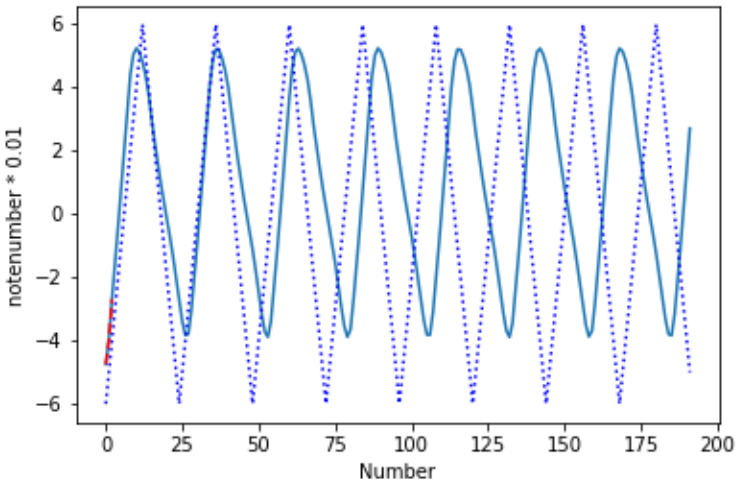


図 7.7　半音階を学習をさせたシステムの予測

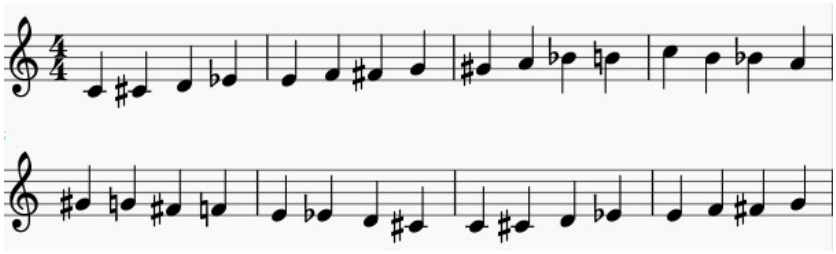


図 7.8　半音階の楽譜

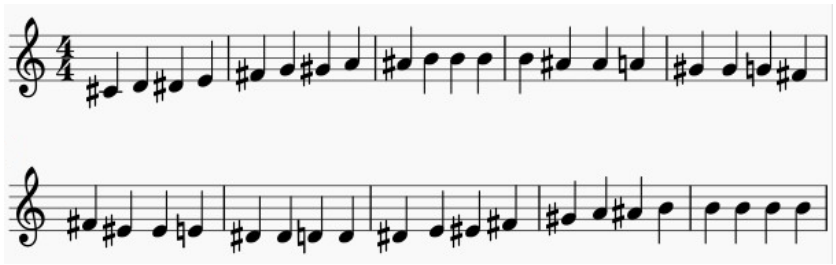


図 7.9　半音階を学習したシステムの出力

説明したものと同様である。図 7.10 は学習の様子である。

横軸はエポック数、縦軸は損失関数の値である。こちらも学習は成功しているように見える。学習した RNN モデルに教師データを 2 個与えて予測結果を見てみると、図 7.11 のようになった。

縦軸が note ナンバーから全体の平均を引いて調整した値、横軸は時間である。こちらもある程度の精度で予測できているようにみえる。入力データを楽譜に起こしたものと、出力を小数点第一位で四捨五入して整数にし楽譜に起こしたものとの、最初の数小節がそれぞれ図 7.12、図 7.13 である。

これらを比べると、音の上昇下降のタイミングなどはほぼ一致している。反面、やはり同じ音が続いたり、C メジャースケールの音以外の音があったりなど、スケールの音型を模倣できたとは言い難い。

次に、アルペジオの学習を行う。入力は C メジャースケールのダイアトニックコードの上昇ア

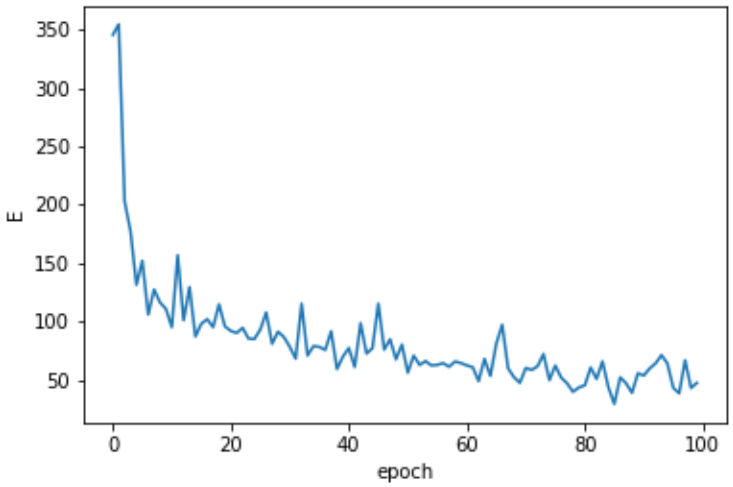


図 7.10　スケールの学習

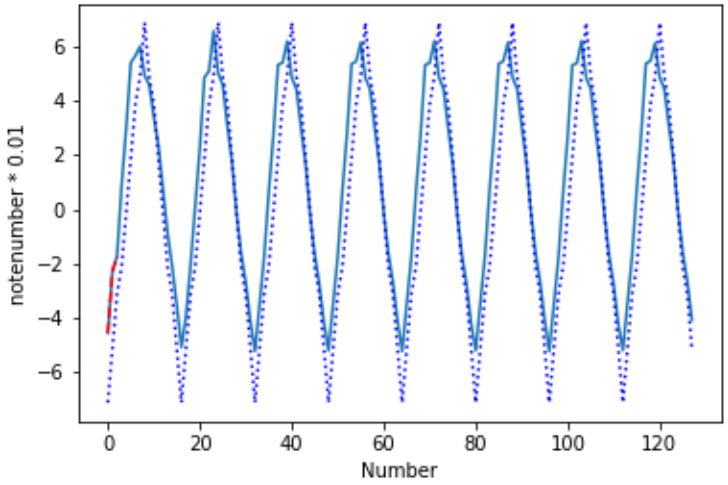


図 7.11　スケールを学習させたシステムの予測

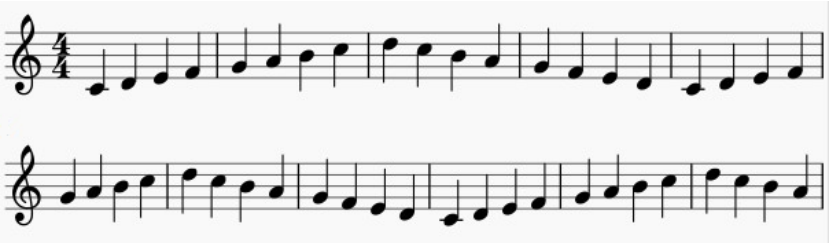


図 7.12　スケールの楽譜

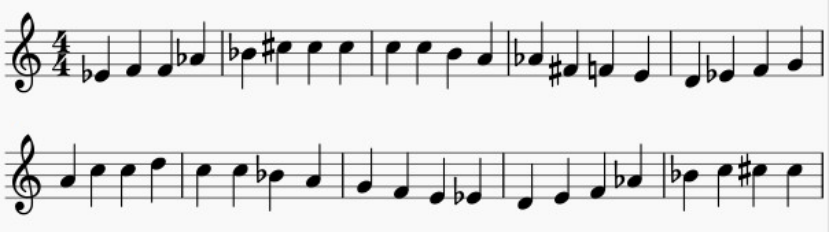


図 7.13　スケールを学習したシステムの出力

ルペジオを順番に並べたもので、これを繰り返したものを用いた。隠れ層のノード数、活性化関数、損失関数などについては先ほど説明したものと同様である。図 7.14 は学習の様子である。

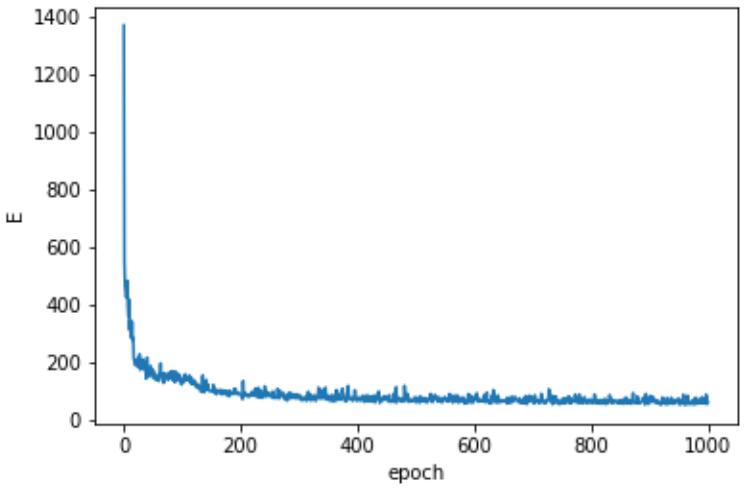


図 7.14　アルペジオの学習

横軸はエポック数、縦軸は損失関数の値である。半音階やスケールと比べて少し複雑になったため、エポック数を 1000 とした。こちらも学習は成功しているように見える。学習した RNN モデルに教師データを 2 個与えて予測結果を見てみると、図 7.15 のようになった。

縦軸が note ナンバーから全体の平均を引いて調整した値、横軸は時間である。こちらもある程度の精度で予測できているようにみえる。入力データを楽譜に起こしたものと、出力を小数点第一位で四捨五入して整数にし楽譜に起こしたものとの、最初の数小節がそれぞれ図 7.16、図 7.17 である。

これらを比べると、音の上昇下降のタイミングは合っていないうえ、C メジャースケール以外の音が多く現れているため、アルペジオの音型を模倣できたとは言い難い。

結果として、これら基礎的な音型の学習について、同じ音を出し続けるようなことはなくなったが、音型の模倣ができたとは言い難い。

しかしここで、マイルス・デイヴィス、チャーリー・パーカーの”K.C.Blues”のアドリブパートを学習させてみる。これまでの章で述べてきたように、ジャズの、とりわけブルースのアドリブパートは、音程を選ぶうえでの制約が少なく、またその都度変化するものであるため、演奏者の特徴が表れやすい。そのため、音型の学習が正確でなくても、演奏者の特徴を模倣できる可能性があると期待した。

入力はマイルス、チャーリーそれぞれの”K.C. Blues” におけるアドリブパートを用いる。隠れ

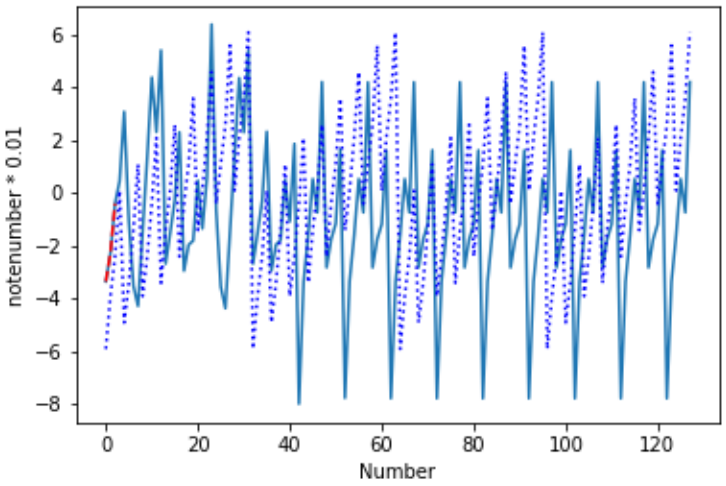


図 7.15　アルペジオを学習させたシステムの予測

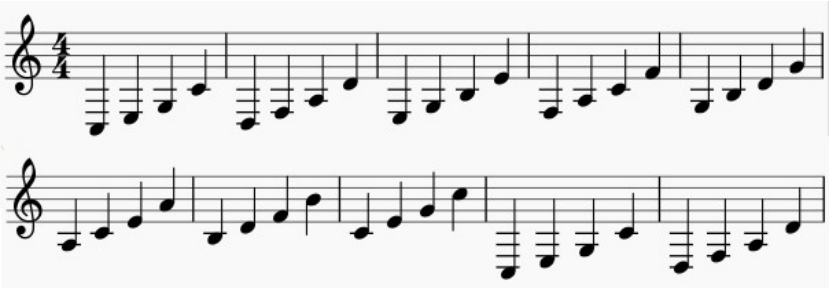


図 7.16　アルペジオの楽譜

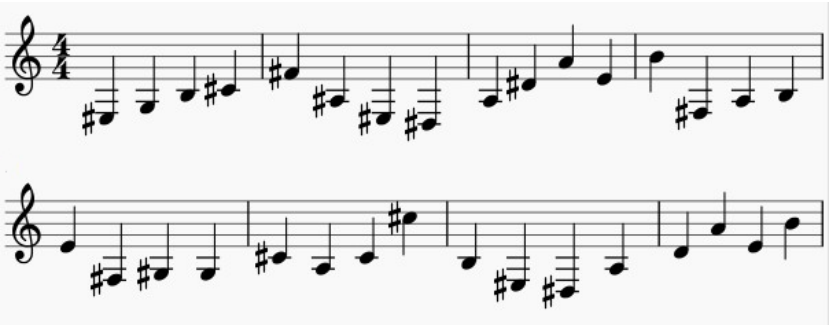


図 7.17　アルペジオを学習させたシステムの出力

層のノード数は 50 とした。活性化関数、損失関数などについては先ほど説明したものと同様である。図 7.18、図 7.19 は学習の様子である。

横軸はエポック数、縦軸は損失関数の値である。こちらも学習は成功しているように見える。学習した RNN モデルに教師データを 2 個与えて予測結果を見てみると、それぞれ図 7.20、7.21 のようになった。

縦軸が note ナンバーから全体の平均を引いて調整した値、横軸は時間である。音型のみの学習ほどではないが、ある程度の精度で予測できているようにみえる。入力データを楽譜に起こしたものと、出力を小数点第一位で四捨五入して整数にし楽譜に起こしたものとの最初の数小節が、マイ

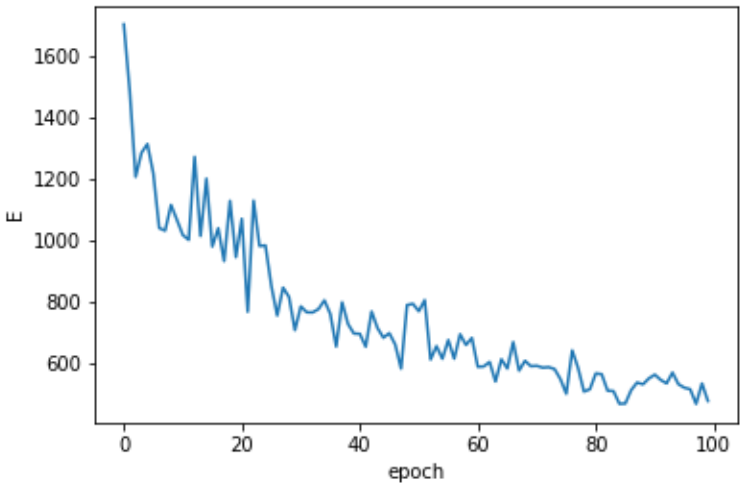


図 7.18　K.C. Blues(マイルス版) の学習

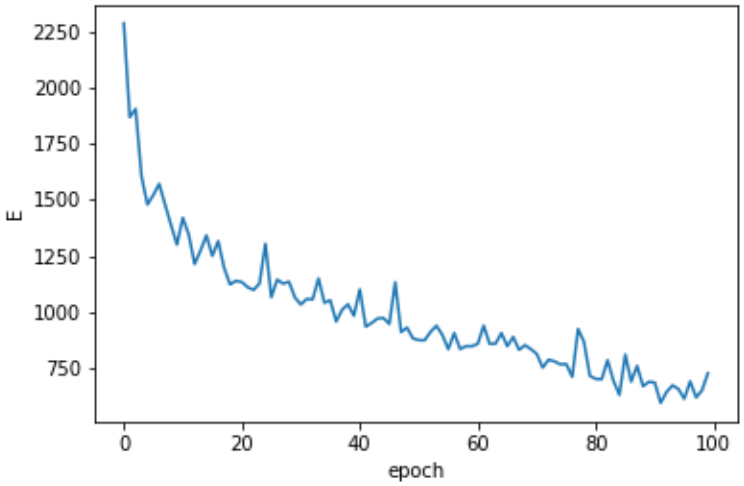


図 7.19　K.C Blues(チャーリー版) の学習

ルスについては図 7.22、図 7.23 で、チャーリーについては図 7.24、図 7.25 である。

先ほどまでの基礎的な音型のみの学習とは異なり、今回はジャズの、とりわけ C ブルースである”K.C.Blues” のアドリブを学習している。これまでも述べてきたように、このようなアドリブパートでは音楽的に使うことを控えなければならない音が少なく、演奏者の特徴が表れやすいことや、どのようなコードに対しても音楽的に正しく聴こえるスケールがあることなどの特徴がある。

そこで、7.1.2 で説明したように、マイルスを学習したシステムの出力とチャーリーを学習したシステムの出力でコルモゴロフ・スミルノフ検定を行う。

帰無仮説　H_0:マイルスを学習したシステムの出力とチャーリーを学習したシステムの出力に差はない

対立仮説　H_1:マイルスを学習したシステムの出力とチャーリーを学習したシステムの出力に差がある

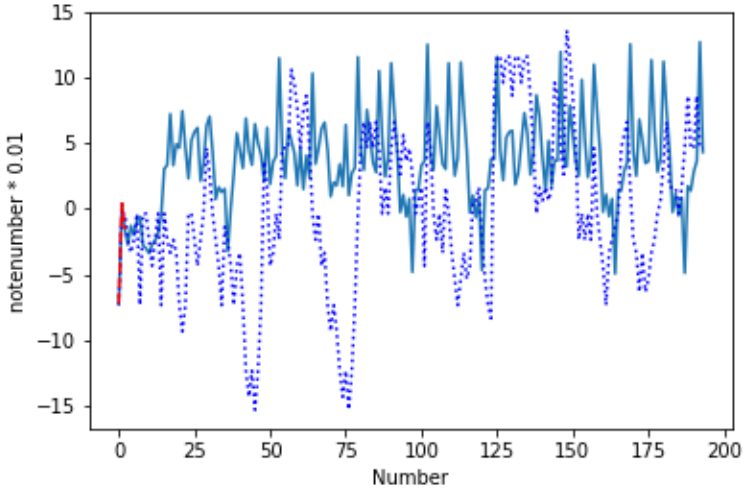


図 7.20　マイルス版を学習させたシステムの予測

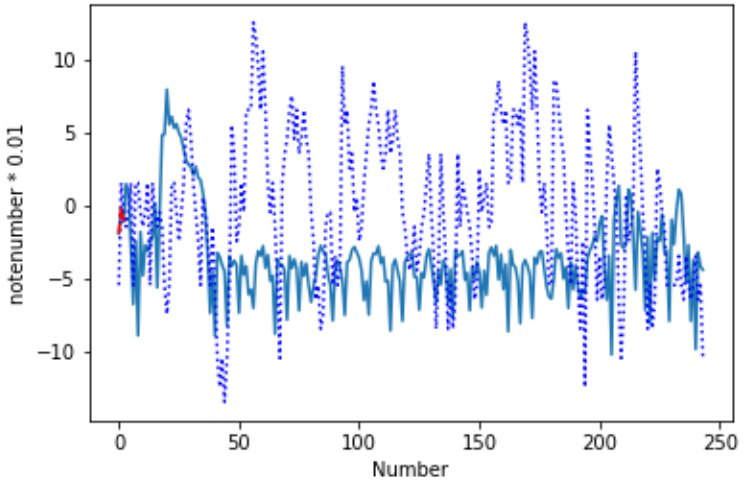


図 7.21　チャーリー版を学習させたシステムの予測

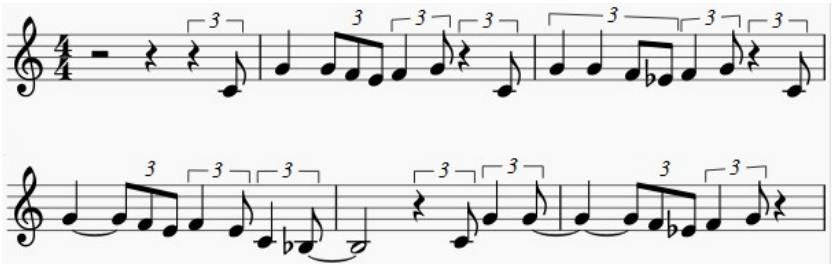


図 7.22　マイルスのアドリブ

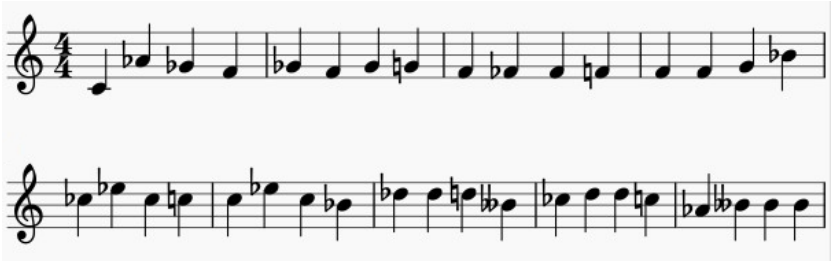


図 7.23　マイルス版を学習させたシステムが作成したアドリブ

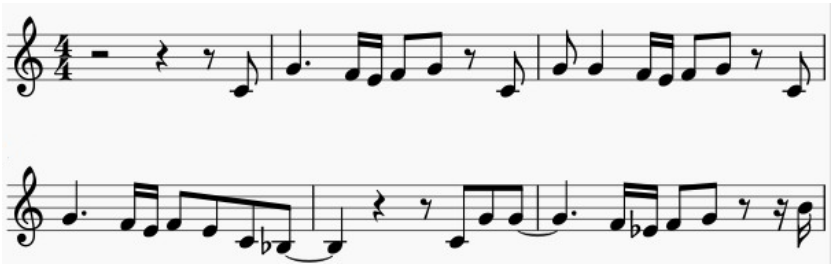


図 7.24　チャーリーのアドリブ

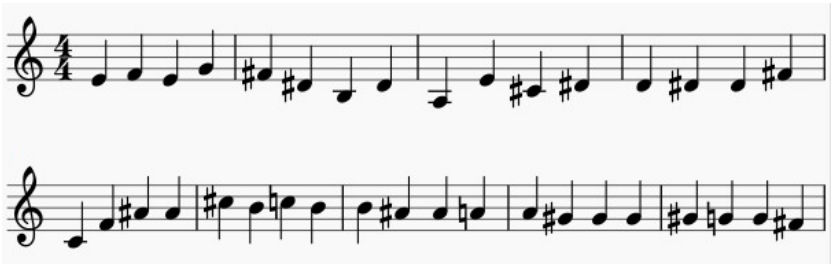


図 7.25　チャーリー版を学習させたシステムが作成したのアドリブ

この帰無仮説を棄却することが出来れば、マイルスを学習したシステムの出力とチャーリーを学習したシステムの出力に差があると言える。また、RNN を用いた機械学習による作曲システムは、成功したといえるだろう。

表 7.4:

	観察度数 (出現回数)		累積度数		累積相対度数		
階級 (音の種類)	第 1 群 (マイルスシステム) 出現回数	第 2 群 (チャーリーシステム) 出現回数	第 1 群	第 2 群	第 1 群	第 2 群	差
1	0	0	0	0	0.000	0.000	0.000
2	0	0	0	0	0.000	0.000	0.000
3	0	0	0	0	0.000	0.000	0.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
54	0	0	0	0	0.000	0.000	0.000
55	0	1	0	1	0	0.004	-0.004
56	0	1	0	2	0	0.008	-0.008

	観察度数 (出現回数)		累積度数		累積相対度数		
階級 (音の種類)	第 1 群 (マイルスシステム) 出現回数	第 2 群 (チャーリーシステム) 出現回数	第 1 群	第 2 群	第 1 群	第 2 群	差
57	0	8	0	10	0	0.041	-0.041
58	0	18	0	28	0	0.115	-0.115
59	0	12	0	40	0	0.165	-0.165
60	1	27	1	67	0.005	0.276	-0.271
61	0	50	1	117	0.005	0.481	-0.476
62	2	61	3	178	0.015	0.733	-0.717
63	2	23	5	201	0.026	0.827	-0.801
64	3	9	8	210	0.041	0.864	-0.823
65	6	7	14	217	0.072	0.893	-0.821
66	3	6	17	223	0.088	0.918	-0.830
67	11	6	28	229	0.144	0.942	-0.798
68	11	4	39	223	0.201	0.959	-0.758
69	23	2	62	235	0.320	0.967	-0.647
70	22	4	84	239	0.433	0.984	-0.551
71	31	3	115	242	0.593	0.996	-0.403
72	23	1	138	243	0.711	1.000	-0.289
73	19	0	157	243	0.809	1.000	-0.191
74	10	0	167	243	0.861	1.000	-0.139
75	10	0	177	243	0.912	1.000	-0.088
76	1	0	178	243	0.918	1.000	-0.082
77	1	0	179	243	0.923	1.000	-0.077
78	5	0	184	243	0.948	1.000	-0.052
79	7	0	191	243	0.985	1.000	-0.015
80	3	0	194	243	1.000	1.000	0.005
81	0	0	194	244	1.000	1.000	0.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
128	0	0	194	244	1.000	1.000	0.000
合計	194	244					

表 7.4 から $P = 5.50236 \times 10^{-65}$ となり、$P < 0.05$ より有意差が示され、帰無仮説を棄却する。したがって、マイルスを学習したシステムの出力とチャーリーを学習したシステムの出力に差があると言えた。

結果として、システムは成功した。しかし現状では音程の学習のみを行うシステムであり、今後更なる改良も考え得る。今後は基礎的な音型の学習を失敗したことの原因究明と、より長期のデータを与えることのできる LSTM(Long Short Time Memories) を実装することが目標になるだろう。

（※文責: 近藤渚紗）

第 8 章　中間発表・成果発表の評価

本章では、中間発表および成果発表で記入してもらった評価シートの集計結果とコメントを参考にした今後の改善点を記述する。

評価シートの評価項目は「発表技術」と「発表内容」の二つを用意した。そして、それぞれについて 1（非常に悪い）から 10（非常に優秀）までの間で点数を付け、それぞれについてのコメント（評価理由やアドバイスなど）を書く欄を用意した。

（※文責: 吉田周平）

8.1　中間発表

8.1.1　評価点数の集計

中間発表で記入してもらった評価シートは計 58 枚だった。シートを記入した人の所属の分布は表 8.1 のようになった。

表 8.1　評価者の分布

所属	学年	人数
教員		8
一般		4
学生	院 1 年	1
	学部 4 年	7
	学部 3 年	30
	学部 2 年	5
	学部 1 年	1
不明		2
合計		58

中間発表の評価は、目的である「学部 1 、2 年生への紹介」という点で見ることとする。学部 1 、2 年生は、 6 名が発表を見に来てくれた。この後の「コメントの解析と改善点」で、興味を示してくれたか否かの是非を記述することにする。

次は、それぞれの評価項目についての平均点を表 8.2 に記す。

全体の平均は「発表技術」については 6.78、「発表内容」については 6.91 となった。「発表技術」を最も高く評価したのは「一般」の人だったが、同時に「発表内容」を低く評価したのも「一般」の人だった。これは、学内の人と学外の人との評価基準が違うという考えることが出来る。

次に、ヒストグラムを用いて比較する。発表技術と発表内容の評価をヒストグラムにすると、図 8.1 のようになった。

どちらのヒストグラムについても、 6 ～ 8 の間に票が集中していることがわかる。また、評価出

表 8.2　各評価項目の平均点

所属	学年	発表技術	発表内容
教員		7.14	6.75
一般		8.00	6.66
学生		6.79	7.09
	院 1、４年	6.87	7.00
	学部 3 年	6.76	7.07
	学部 1 、２年	6.83	7.33
全体		6.78	6.91

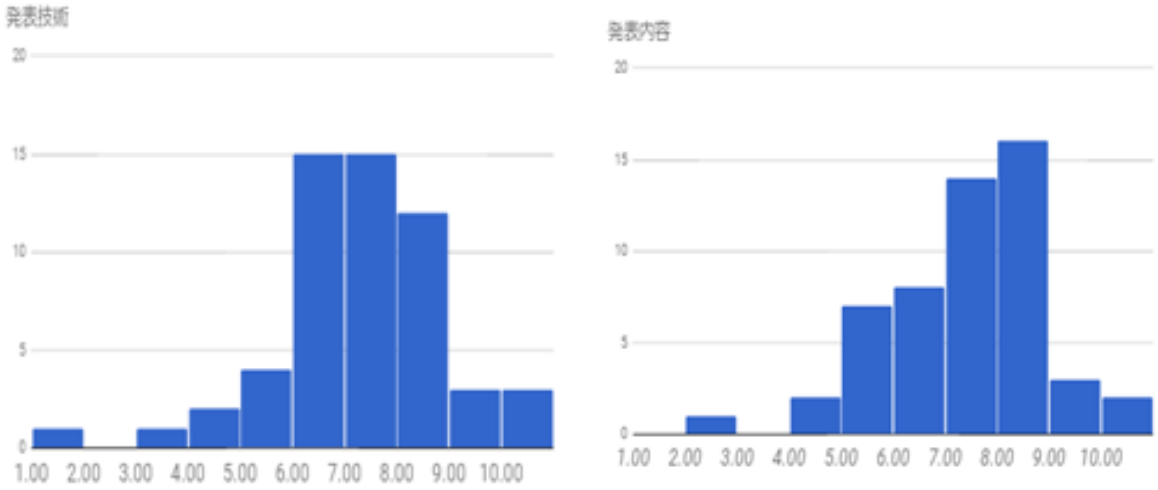


図 8.1　発表技術（左）と発表内容（右）の評価のヒストグラム

来ないという理由で評点を記入しなかった人がいたため、データの個数が「発表技術」にういては２個、「発表内容」については５個不足している。故に、「発表技術」は 56 個、「発表内容」は 53 個のデータで集計したものである。

次に、散布図を見ていく。散布図は図 8.2 のようになった。

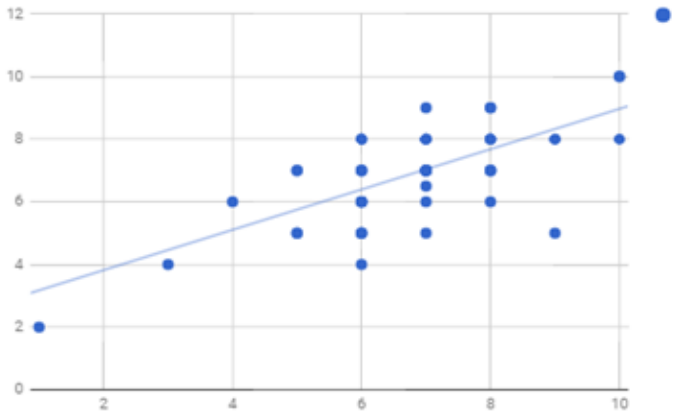


図 8.2

また、「発表技術」と「発表内容」の評点の相関係数は 0.68 となった。これは、二つの評価項目が少しだけ関連していると言える数値である。

では、次にコメントについて解析する。

（※文責: 吉田周平）

8.1.2 コメント解析と改善点

まず、「発表技術」について、多かったコメントをポジティブなコメントとネガティブなコメントに分けて並べる。

- ポジティブ
  - 声が聞きやすい（大きさ、話し方）
  - 実際の音があって良い
  - 専門用語の説明が分かりやすい
- ネガティブ
  - 練習、準備不足を感じた
  - 早口で聞き取りにくい
  - スライドが見えにくい（色、大きさ、文字量）

ポジティブなコメントと同じ理由でネガティブなコメントを書いている人もいた。また、小数派の意見だったが、「スライド番号」や「マウスの使い方」についての意見もあり、それらについても鑑みて発表技術の改善点を考えた。

- 改善点
  - スライドをもっと早い段階から作る。
  - 時間を考慮して早口にならないように調整する。
  - 無線マウスを活用する。
  - 指示棒の導入。
  - 説明にアクセントを加えて聞きやすくする。

などがあげられる。以上の点から、発表技術の改善・向上に取り組むことにする。

次に、「発表内容」についても同様にして、多かったポジティブなコメントとポジティブなコメントを並べる。

- ポジティブ
  - 手法（アプローチ）が面白い。
  - 成果物もあり、楽しく聞けた。
  - 興味深い。
- ネガティブ
  - 目標が明確ではない。
  - 難しい。
  - わかりにくい。

「発表内容」に関しては、感想よりもアドバイスや質問が多かった。重要なアドバイスが沢山あったため、これについても並べていく。

- プロやジャズに詳しい人に聞いてみたらどうでしょうか。
- 評価方法を改善すべき。
- 過学習対策は？
- 模倣が成功の基準は？
- 中間発表の目的に則していない。

これらの意見に関しても、後期の活動に反映することとする。

以上より、中間発表の評価コメントは賛否両論であり、様々な意見も多く、とても参考になった。

（※文責: 吉田周平）

8.2 成果発表

成果発表会において評価者に配った評価シートでは、「発表技術について」および「発表内容について」を 1(非常に悪い) から 10(非常に良い) までの 10 段階で評価者に評価してもらい、それらを集計し平均を計算した。

成果発表で記入された評価シートは 49 枚であり、評価シートに評価を記入した人の所属の分布は表 8.3 の通りである。

表 8.3　評価者の分布

所属	学年	人数（前半/後半）
教員		6 （3 / 3）
一般		5 （2 / 3）
学生	院 2 年	1 （1 / 0）
	学部 4 年	1 （1 / 0）
	学部 3 年	30（14/16）
	学部 2 年	5 （5 / 0）
不明		1 （1 / 0）
合計		49（27/22）

表 8.3 のから分かるように、主に評価を行ったのは学部 3 年生であり、そのほかの学年の学生は後半の発表に来ていなかった。なお、評価できないという理由で一部評価が書かれていない評価シートもあった。そのため統計に使用された実際の枚数は発表技術についてが 48 枚、発表内容についてが 47 枚である。

発表技術についての評価の平均は、表 8.4 の通りである。

評価シートの統計を取った結果、発表技術についての評価の平均は全体で前半が 6.667 点、後半が 7.59 点、総合で 7.082 点となった。なお、平均は (評価の合計点)÷(評価の記入された枚数) で算出した。そのため、アンケートの評価欄に数値が記入されていないものも時折見受けられたがそれらは計算に入れてない。なお、数値は小数点以下第 4 位で四捨五入したものである。

発表技術を最も高く評価をしたのは後半の「一般」の人で 8.5 点であったが、同時に最も低く評価をしたのも前半の「一般」の人で 5 点であった。総合では不明のシートを除いた場合「一般」の人からの評価が最も低い 7.2 点であり、「学生」の学部 2 年生、学部 4 年生からの評価が最も高い 8

表 8.4　発表技術のについての評価の平均点

所属	学年	前半	後半	総合
教員		8.00	7.333	7.6
一般		5.00	8.5	7.2
学生		7.333	7.438	7.378
	学部 4 年	8.00		8.00
	学部 3 年	7.143	7.438	7.3
	学部 2 年	8.00		8.00
	不明	7.00		7.00
全体		6.667	7.59	7.082

点であった。

発表技術についてネガティブなコメントで特に多く見受けられたものは、「声が小さくて聞き取れなかった」というものであった。これらの原因として考えられるものは、事前の発表練習において周りでも発表が行われていることを考慮していなかったため、結果として周囲で行われているほかのプロジェクトの発表の声にかき消されてしまっていたからなのではないかと推測できる。また、発表者がスライドの方を見ながら発表する回数が多かったことも、このコメントをもらうことになっ一因だろう。また、「字が小さい」、「周りが明るいせいで、スライドが白くて見づらい」などのコメントもあった。これは、スライドと評価者の距離感を事前に把握しきれなかったことが原因だと考えられる。

ポジティブなコメントとしては、「評価者からの質疑応答に適切に答えていた」「身振り手振りがあってわかりやすかった」などがあり、発表者自体の技術を高く評価する意見が多く見られた。しかし、人によっては上記の「声が小さくて聞き取れなかった」というコメントとは逆の「声の大きさ、スピード共にちょうどよかった」というコメントも見受けられ、評価者内で評価に大きなばらつきが見られた。

発表内容についての評価の平均は、表 8.5 の通りである。

表 8.5　発表内容のについての評価の平均点

所属	学年	前半	後半	総合
教員		8.50	7.333	7.8
一般		5.50	9.50	7.5
学生		7.714	8.20	7.917
	学部 4 年	9.00		9.00
	学部 3 年	7.57	8.20	7.897
	学部 2 年	8.00		8.00
	不明	7.00		7.00
全体		7.04	7.81	7.375

発表内容についての評価の平均は、全体で前半が 7.04 点、後半で 7.81 点、総合で 7.375 点となった。なお、これらの数値も発表技術についての評価と同様の式で算出したものである。

発表内容を最も高く評価をしたのは後半の「一般」の人で 9.5 点であったが、同時に最も低く評価をしたのも前半の「一般」の人で 5.5 点であった。総合では不明のシートを除いた場合「一般」の人からの評価が最も低い 7.5 点であり、「学生」の 4 年生からの評価が最も高い 9 点であった。

発表内容についてのコメントは発表技術についてのコメントに比べて数が少ないが、その分書かれたコメントの文章量は多かった。

ネガティブなコメントとしては「専門用語に対する説明が少なかった」というものがあったが少数であり、他のコメントのほとんどはポジティブなものであった。

ポジティブなコメントを具体的に挙げると、特に多かったコメントが「よくまとまっている」というものであり、専門的な知識が多く難しいはずの内容でありながら評価者に伝わるものであったと考えられる。コメントの中には「前期に比べてすごく進歩している」「失敗の原因について良く考察できていた」といったものも見受けられ、「成果物の完成形が楽しみである」というコメントもあった。

（※文責: 中川哲哉）

第 9 章　まとめと今後の展望

さて、本プロジェクトがおこなってきたことを簡単に振り返ってみたい。

本プロジェクトでは、プロのミュージシャンのジャズ・アドリブ演奏を模倣するアドリブ自動生成プログラムの作成を目指した。そのために、まず基礎的な音楽理論について学習した。このとき、特にジャズ、なかでも F ブルースにおける音楽理論について理解を深めた。そのうえで、私たちはフラクタルの性質をもつ 1/f ゆらぎに着目した。私たちが日ごろ耳にする音楽の多くは 1/f ゆらぎに従っており、人に心地よさを与えると言われているからである。そこでガードナー [16] を参考にして、白色雑音、褐色雑音、1/f ゆらぎをそれぞれ用いた乱数生成方法によるアドリブ生成システムを作成した。またそれらを実験により評価し、作成した曲の優位性を検証した。結果として、使うオクターヴを増やす、褐色雑音を反射壁と吸収壁に分けるなど試行錯誤を繰り返すも、どの雑音にも優位性は認められなかった。このとき、乱数の分布から自己相関を求めると、それぞれの雑音の特徴は反映されており、乱数生成そのものは成功しているといえた。また、使用する音の長さは四分音符、四分休符のみで統一し、休符の出現確率は雑音に関わらず 30% 程度で統一し、音ごとの長さや音楽的役割などについては考慮しなかった。さらに、用いるスケールをメジャースケールとブルーノートスケールの二種類としたが、こちらについても優位性は認められなかった。

プロのジャズミュージシャンのアドリブ演奏を模倣する作曲システムを作成するにあたって、近年の自動作曲システムが深層学習を用いていることに目を付け、その足掛かりとしてニューラルネット、リカレントニューラルネットを用いた機械学習による学習を試みることにした。斎藤 [?]、巣籠 [23] を参考にして、ニューラルネット、リカレントニューラルネット、機械学習の基礎的な理解を深め、それらを用いたアドリブ生成システムを作成した。これらのシステムは、ある音の次に来る音の確率を学習し、出力するものである。ニューラルネットを用いたシステムについて、入力は音の高さを 0 から 36 の整数に定義して数値化したものを用いた。数字が大きくなるにつれて高い音を表し、36 を休符とした。重みの初期値は、合計が 1 となるように 0 から 1 までの数からランダムに決定した。活性化関数は、ソフトマックス関数を用いた。損失関数は、交差エントロピー誤差を用いた。重みの更新には勾配法を用い、学習率は 0.01 とした。収束判定条件は、前の交差エントロピー誤差の絶対値と、今の交差エントロピー誤差の絶対値の差が 0.0001 以下になったときとした。このとき、学習回数の上限は 10000 回とした。以上のようなプロセスを、用いる全ての音に対しておこない、0 から 36 の整数に数値化されたものを出力した。このとき、学習は一層でおこなわれた。

まとめると、まず前期では全ての作曲システムにおいて、音程を選ぶことのみをおこなっており、音価や音量、音色、楽器なども自動で選べるようなシステムを実装することが考えられる。このとき、ジャズの音楽的理論から、三連符や五連符などのリズムを使うこと、裏拍に強拍を持ってくることなどを考慮するとさらにシステムの改良を図れるかもしれない。また、このような実装をするさい、3 種の雑音によるアドリブ生成システムにおいて、音程を含む全ての項目で同種の雑音でアドリブ生成をすることで、さらに雑音の特徴が曲にあらわれ、前期のように比較実験をおこなうとき、1/f ゆらぎに優位性が示されることが期待できるかもしれない。別のアプローチとして、2 章で述べたような音楽理論を用いて音を出力するシステムの実装も考えられる。加度 [17] で説明されている、スケールやアルペジオ、半音階などの音の音楽的役割を明確にし、プロのジャズ・ア

ドリブもこのような音楽的役割から読み解き、それをもとにしたシステムの実装方法なども模索できるかもしれない。現行システムの改善点としては、前期の作曲システムの出力は全て音程と休符を数値化したもので、MIDI データで出力するなどがある。また、ニューラルネットを用いた機械学習によるアドリブ生成システムについて、完成したシステムで生成した曲がどの程度の完成度であるか検証するため、6 章で紹介した編曲した Bags' Groove にどのくらい似ているか検定すること、音の散布図を三種雑音と比較、検定することなどが考えられる。このアドリブ生成システムをより正確にするために、層を増やすことや、用いる関数を変えることなどが考えられる。同じ機械学習でも、リカレントニューラルネットワークを用いた機械学習によるアドリブ自動生成システムを作成することも考えられる。他に、本プロジェクトの目標であるジャズ・アドリブの模倣について、模倣とはどのようなことであるか、模倣できたかどうかの判断をする基準を明確にすることなども課題となっていた。

後期では、リカレントニューラルネットワークを用いて機械学習を行い、プロのジャズミュージシャンのアドリブ部分を教師データとして学ばせてアドリブ演奏を模倣させるアドリブ生成システムの構築を目標とした。まず、マイルス・デイヴィスやチャーリー・パーカーが作った 110 曲の音楽の中から、ジャズの音楽的理論より教師データとしてふさわしいものを最終的に 2 つ選び、前期のアプローチとして考えていた音楽理論や音楽的役割によってプロのジャズ・アドリブを読み解き教師データを作成した。学習システムは Sin 波の予測をする RNN を基盤に、音楽生成システムの作成を試みた。まず、テストとして Sin(100) まで予測させた結果、ある程度予測できることが分かった。次に、教師データをかえるの歌にして予測を行った。しかし、教師データを読み込んだ結果、最初のうちはしばらく予測を行うが、最終的に一音のみを出力し続けることが判明した。失敗の原因については解明することができなかった。問題の原因として、リカレントニューラルネットワークでは長期の学習ができない欠点があり、それが関係しているのではと考えられる。結果として、音程だけを学習し、入出力を MIDI にするシステムを作成することが出来た。この点においては前期の目標を一部達成したといえる。

今後の展望は、この不具合の解決と、リカレントニューラルネットワークでは出来ない長期の学習が出来る LSTM を用いてシステムを構築し、アドリブを生成することである。

（※文責: 伊藤祐大）

参考文献

[1] Doya, K. 2002*Recurrent Networks: Learning Algorithms*. Handbook of Brain Theory and Neural Networks, 2nd edition.

[2] Gardner, M. 1991. *Fractal Music, Hypercards and More....* W.H.freeman & Co Ltd.

[3] Levine, M. 1995. *THE JAZZ THEORY BOOK*. ATN.

[4] 1982. *Charlie Parker Omnibook: For C Instruments. (Treble Clef Version)*. Criterion Music Corp.

[5] Spi 版. 2015. *Miles Davis Omnibook: For C Instruments*. Hal Leonard Corp.

[6] Blip Interactive Ltd.「Nano Studio Version 1.44」, ⟨http://www.blipinteractive.co.uk/⟩.

[7] MI7 Japan(2017)「Finale」, ⟨https://www.finalemusic.jp/⟩

[8] steinberg(2015)「Cubase 8」, ⟨https://japan.steinberg.net/jp/products/cubase/start⟩.

[9] ta-ka(2016)「python で RNN 実装 - Qiita」, ⟨https://qiita.com/ta-ka/items/1e4f3414d478618c373c⟩(参照 2018-1-17).

[10] Takabo Soft(2013)「Domino - MIDI 音楽編集ソフト Version 1.43」, ⟨http://takabosoft.com/domino⟩.

[11] Werner Schweer ほか「MuseScore 2.1」, ⟨https://musescore.org/ja⟩

[12] 伊藤雄哉・山西良典・加藤昇平・伊藤栄則 (2011)『楽曲に対する感性評価と音響ゆらぎ特徴の対応付け』, 日本感性工学会論文誌 Vol.10 No.3 pp.341-348.

[13] 薄浩之・乾伸雄・野瀬隆・小谷善行 (1998)『ニューラルネットワークを用いたアドリブ生成のためのリズム学習』, 情報処理学会研究報告音楽情報科学（MUS）1998(74(1998-MUS-026)), 39-44, 1998-08-07.

[14] 梅村祥之 (2014)『規則的に生成した 4 音符からなる楽曲を用いた楽曲の心地よさに関する客観評価指標』, 情報処理学会研究報告 Vol.2014-MUS-104 No.17.

[15] 大場稜也・村瀬慶介・高橋淳二・戸部義人 (2017)『深層学習による楽曲分類の検討』, 平成 28 年度電子情報通信学会東京支部学生会研究発表会. 大矢健一 (1994)『ニューラルネットワークのダイナミクスによるリズム認知モデル』, 音楽情報科学 7-1

[16] ガードナー, マーチン (1996)『ガードナー数学マジック フラクタル音楽』(一松信訳), 丸善.

[17] 加度克紘 (2015)『ドレミで覚えるジャズ・アドリブの法則』, リットーミュージック.

[18] 金森光平・星野准一・浜中雅俊 (2016)『クラスタリングと機械学習を用いた音楽理論 GTTM に基づく楽曲構造分析システム』, 情報処理学会研究報告 Vol.2016-MUS-110 No.18.

[19] 熊本忠彦 (2004)『印象に基づく楽曲検索のための個人適応手法の設計と評価』, 第 3 回情報科学技術フォーラム.

[20] 斉藤康毅 (2016)『ゼロから作る DeepLearning - Python で学ぶディープラーニングの理論と実装』, オライリー・ジャパン.

[21] 齋藤創・大場みち子 (2017)『機械学習を利用した DTM 音色検索フィルタの提案と音色づくり支援システムへの適用』, 情報処理学会研究報告 Vol.2017-MUS-114 No.26.

[22] 杉本知仁・太田晶大・森山甲一・栗原聡・沼尾正行 (2008)『自動作曲システムにおける協調フィルタリングを用いた楽曲評価予測』, The 22nd Annual Conference of the Japanese Society

for Artifical Intelligence.

[23] 巣籠悠輔 (2017)『詳解ディープラーニング TensorFlow・Keras による時系列データ処理』, マイナビ出版.

[24] 東条敏・平田圭二 (2017)『音楽・数学・言語 情報科学が拓く音楽の地平』, 近代科学社.

[25] 友田勝久「関数グラフソフト GRAPES 7.35」, ⟨http://www.osaka-kyoiku.ac.jp/ tomodak/grapes/⟩(参照 2018-1-17).

[26] 野村亮・栗田多喜夫・相原玲二 (2015)『楽器音辞書を用いた非負値疎分解による楽譜の生成』, 情報処理学会研究報告 Vol.2015-MUS-109 No.6.

[27] 玉木明和・姚鳳会・加藤清史 (1996)『フィードバック機能をもつ自動演奏システムの試み』, 音楽情報科学 15-13.

[28] 矢向正人・土屋景一・荒木敏規 (1996)『旋律パターンの分類 - 類似性判断と分析例 -』, 音楽情報科学 16-5.

[29] 横岡ゆかり (2016)『大好きな曲で楽しくレッスンジャズピアノはじめよう！』, 日本放送出版協会.

[30] 吉野巌・阿部純一 (1993)『メロディの調を認定する過程の計算モデル』, 音楽情報科学 3-4.

[31] 力徳正輝 (2016)『楽曲生成モデリングのための音符クラスタリング』, 情報処理学会研究報告 Vol.2016-MUS-113 No.18.

[32]「Ampermusic」, ⟨https://www.ampermusic.com/⟩(参照 2017-12-15).

[33]「Deep Jazz」, ⟨https://deepjazz.io/⟩(参照 2017-12-15).

[34]「SPIRIO」, ⟨http://www.steinway.co.jp/spirio⟩(参照 2017-12-15).

[35]「偏った DTM 用語辞典 カナインデックス」, ⟨https://www.g200kg.com/jp/docs/dic/⟩(参照 2017-12-15).

[36]「ジャズ資料館 - Jazz Database for Musicians -」, ⟨https://jazzshiryokan.net/index.html⟩(参照 2017-12-15).

[37]「ニューラルネットと深層学習」, ⟨http://nnadl-ja.github.io⟩(参照 2017-12-15).