GPT-4による対話的音楽生成のための プロンプトエンジニアリング

川口 竜斉[1,a)]　片寄 晴弘[1,b)]

概要：
生成系 AI 技術の進展の一つの形として Text to music スタイルによる自動作曲システムの実用化が進んでいる．このシステムの使用により，簡易な操作によって素人レベルのユーザが想像を超えるレベルの音楽を作ることができるようになったが，ユーザ自身の意図をどのように作品に反映させるかという観点からは，解決すべき課題が残されている．我々は，GPT-4 を基盤としたプロンプトエンジニアリングの枠組みを利用し，一部のパラメータによる意図指定手法も用いて，対話的な作曲作業のフレームワークのデザインに取り組んでいる．本稿では，この取り組みをデモを交えて紹介する．

1. はじめに

深層学習による生成 AI の発展により，コンテンツ制作の形が大きく変わりつつある．自動作曲においても text-to-music 系の生成 AI の性能は実用レベルにまで向上しており，たとえ音楽的知識を持たないユーザであっても，即座に音楽を作り出せる時代になりつつある．text-to-music 系生成 AI では，単語や簡単な文章の入力によっての作品制作上の意図を設定することができる．ジャズ風，激しい等の曲調の指定による作曲作業の代替が可能となったが，細かい意図指定や指定した部分の曲調を少し変えるといった指示を伝達して所望の制作物を得ることは困難である．つまり，現状の生成系 AI においても **directability** の実現（実装）という課題が残されている．**directability** は生産性向上という視点だけではなく，制作における本質的な楽しみも増強されると期待される．[6] 大規模言語モデル（以下, **LLM**）GPT-4 [2] の応用領域の一つとして物語の自動生成がある．この領域では「推敲」「後続文の作成」「あらすじの生成」に留まらず，一部に編集者想定での利用の検討も進みつつある [1]．音楽とのバインディングをどう実現するかという問題があるが，このことは，大規模言語モデルを **directability** の実現（実装）できる可能性を示唆している．

GPT-3 の枠組みを使って音楽生成を行うものとしては海老ら [5] の研究が挙げられるが，この研究では歌詞に対するコード進行の生成を対象としており，メロディを含む音楽生成を対象とはしていない．さらに Directability を持つものとしては，[3] と呼ばれる DAW が挙げられるが，こちらはユーザの音楽制作を支援する形で GPT を活用する．それに対して本研究では，ユーザは指示を与えるのみにとどまり GPT に直接音楽を制作させることを目的としている．他にも，GPT 作曲くん [4] と呼ばれる ChatGPT を活用した音楽自動生成ツールが存在する．こちらでは，Diffusion モデルに対して与えるパラメータを列挙するというタスクにおいて GPT が活用されており，本研究の活用方法とは異なる．

本稿では，GPT-4 を基盤としたプロンプトエンジニアリングの枠組みを利用し，音楽制作における **directability** の実現（実装）に向けての取り組みを紹介する．

2. アプローチ

本章では，**LLM** を利用した作曲支援のアイデア，実装アプローチについて述べる．**LLM** 利用の有無にかかわらず，作曲プロセス，およびその支援においては，対象とする対象音楽ジャンルにおいて破綻のない生成物を確保することが目標となる．近年の音楽ジャンルにおける生成 AI は **wav ファイル**に代表される信号出力のものも増えてきているが，ここでは，ABC 記譜法の利用を想定する[*1]．ABC 記譜法はアスキー文字で譜面を表現できるように考案された記法であり，web 上でも多くの事例が集積されており，**LLM** においても学習対象になっていることが想定

[1] 関西学院大学
[a)] hcu83327@kwansei.ac.jp
[b)] katayose@kwansei.ac.jp

[*1] このノーテーションをベースとしたポストプロダクションとしての編曲の実施，さらには生成系 AI の適用を想定している．

される．物語の生成タスクにおいては，**directability** 視点での生成 AI の活用法が模索され，一部で活用も始まっている．一方で，作曲プロセスにおいての試みについては現時点でほとんど報告例がない．我々は，作曲プロセスにおける **directability** の確保の視点として以下のような事項を想定している．

(1) 自然言語の指定による楽曲生成
(2) 形容詞と関連づけた制御
(3) 指定された区間に関するフィードバック受理と修正
(4) 指定した雰囲気に合致した音源や楽器の自動選定
(5) 楽曲に合致した音響や MIX に関する自動調整

これらのうち, (1) については既に, text-to-music 系生成 AI において実現されてされているものの，滑らかな意図の伝達をどう実現するかという課題がある．本稿では，GPT-4 と連携した (1) から (3) までの実装について取り扱う[*2]．

3. ユーザ視点からみたシステム利用イメージ

前章で述べた目標を達成するものとして，ここでは，図 1 に示すような構成のシステムの開発に取り組んでいる．

初めに，ユーザが指示を自然言語や感情パラメータの形式でシステムに入力を与える．ここでは,「雨を題材にした曲を作って」というテーマを指定する指示だけではなく,「前よりもっと明るめなワルツを作って」といった前の生成結果を参照するような指示や,「明るさ 0.7 / 1.0, 躍動感 0.2 / 1.0」というような形容詞とその度合いの組み合わせによって作りたい曲を指定できるものとする．次に，システムがその入力内容を送信プロンプトに埋め込み，GPT-4 に送信する．この送信プロンプトはあらかじめ入力欄が空白になっているプロンプトのテンプレートであり，入力欄以外の箇所には，GPT-4 に行わせる音楽生成の手順やその制約が記述されている他，few-shot learning を行うための入力例-出力例の組がいくつか与えられている．出力例の楽譜は ABC 記譜法によって記されている．そして，GPT-4 の API からレスポンスを受け取り，その中に含まれる ABC 記譜法での楽譜出力を抽出する．抽出した結果はシステムが wav ファイルに変換し，ユーザーに提示されることとなる．以降，結果を聴いたユーザは出力された楽曲に対して様々な要望を自然言語で入力することができるようになる．この状態で再度指示を与えることで，すでに生成された作品に対してフィードバックを与えられる．例えば，生成された曲が暗すぎれば「明るさ」のパラメータを 0.1 ほど高めるよう修正したり，生成された曲の長さが足りなければ「さらに曲を伸ばしてください」という指示を与える．この過程を繰り返して生成作品を改良し続ければ，ユーザは所望の楽曲を生成することができる．

[*2] (2) に関連して，GPT で感情を取り扱うことができるかどうかという議論もある．入力された文章をもとに執筆者の感情を推定しパラメータとして提示できる事例報告がある．

4. GPT-4 上での directability 実現のためのプロンプトエンジニアリング

この節では，2 節の仕様を GPT-4 上で実現するにあたって発生する問題と，その解決法について取り上げる．

事前知識を与えていない GPT-4 にメロディの生成を行わせようとすると，おおよそ次のような傾向がみられる．

(1) 順次進行が著しく長く続く．曲の大部分がほとんどが一度の進行で構成されている．
(2) 中盤と終盤ともに，序盤に提示したモチーフからかけ離れたメロディが展開されている．
(3) モチーフの展開やリズムパターンが楽曲を通して一通りしかない．
(4) 1 小節に全音符以上の音価を詰め込む．
(5) 一種類の音符だけを使っている．
(6) 全音符, 二分音符だけで曲を構成する．(一般的な楽曲は四分音符を基本的な音価として作られる．)

これらの傾向は実際の楽曲にはあまり見られない．よって，要求事項 (1) を達成するためにはこの傾向から脱却している楽曲を生成するプロンプトを発見することが肝要となる．以降，そのようなプロンプトを「性能が良い」と評価することとする．以下では，これらの問題を解決するためのプロンプト構成法について述べる．

まず，実験的結果からプロンプト中に”piano”という文言を挿入するだけで生成性能が大幅に向上したことが判明した．つまり，ほぼ同じプロンプトであっても”piano”を入れ込んだプロンプトの方がより傾向 (1), (2), (3) を回避しやすいことを確認した．これは，トークン”piano”および ABC 記譜法のトークンの入力によって，Transformer の生成単語の探索範囲が大規模言語モデルのコーパスに含まれる楽譜データが存在する領域に限定されたことが原因だと推察する．楽曲の生成プロンプトだけでなく，与えられたメロディの拡張タスクにおいても同様の性能向上が確かめられた．また，GPT-4 のプロンプトエンジニアリングには zero-shot と few-shot の二通りが存在する．今回の生成タスクにおいては，few-shot の方がより安定した生成出力が得られることがわかった．それに加え，GPT の出力する ABC 記譜法の形式も例示に示した書式に一致するという効果も確認されている．すなわち，few-shot をプロンプトに採用することで，単純に性能を高めるだけでなく，クライアントアプリケーションにとってパースのしやすい楽譜を扱えるというメリットもある．さらに，本研究で使用しているプロンプトでは傾向 (1) - (6) と真逆のことを行うよう，具体的に記述してある．今回採用した few-shot learning では生成楽譜の具体例もプロンプトにいくつか記述する．そのため，プロンプトが長くなることが避けられず与えた制約が無視される可能性が高くなる．この問題に対しては，プロンプトを分割した上で各部分プロンプトを

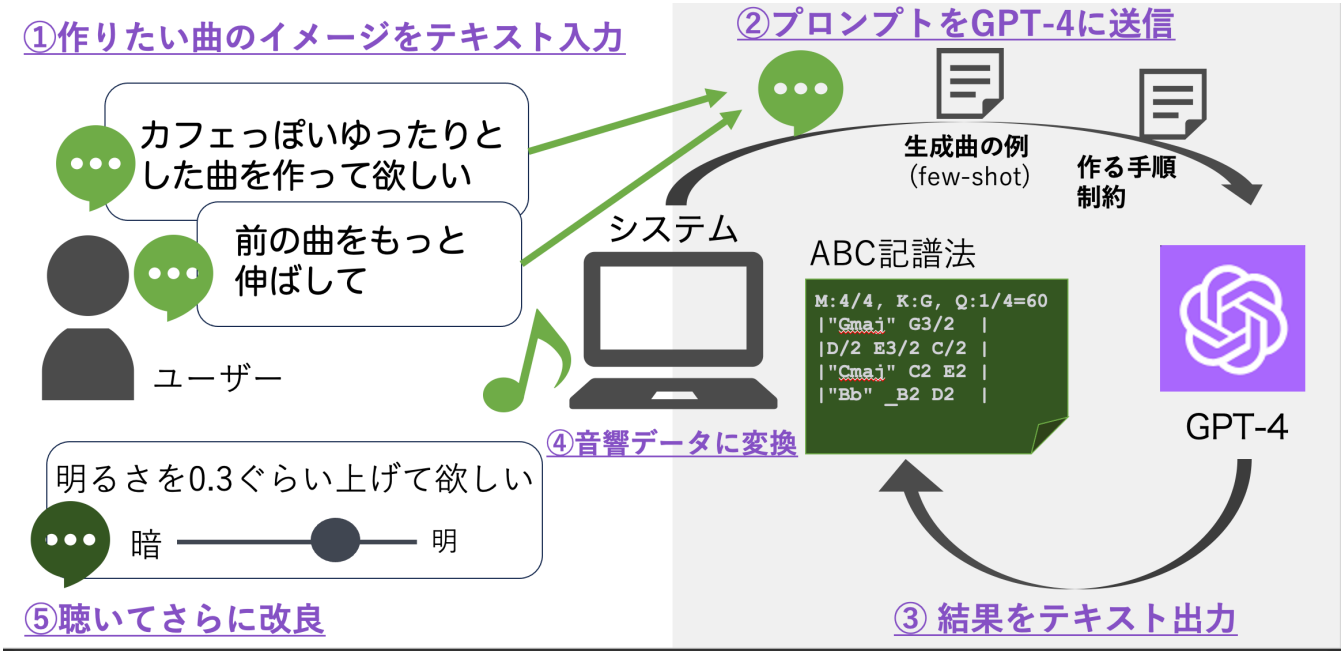


図 1 ユーザの指定を反映した楽曲

読み終えるたびに要約させる refine パターンという対策が考えられる．しかし，こちらでも与えた指示を無視する頻度が頻度が高まったため，この方法を採用できなかった．代わりに，一度生成した楽曲について自身で与えた問題点を指摘させ，その上で一部を修正させるという方法をとっている．これにより，比較的安定した出力結果を得ることに成功した．

以上のことを踏まえ，今回の楽曲生成プロンプトは次のような構造を持つ．なお，性能向上を図り実際のプロンプトは入力部を除き全て英語で記述されている．

- (条件指示)「ピアノ曲」であること, (1)-(6) とは真逆のことを具体的に指定する場所．他にもノンダイアトニックコードや C 以外のキーの仕様を推奨している．
- (入力) ユーザの入力が埋め込まれる箇所
- (最終出力) 店舗情報を含めた ABC 記譜法で出力することを指示．
- (例示) 入力に対する最終出力の例示を 5 例ほど列挙 (few-shot learning)
- (重要指示) 後述

重要指示部には次の内容が記載されている．

(1) 指示に従いピアノ曲を制作せよ．
(2) 生成した楽曲に対して，条件指示部の指示の中で満たしていない点を数点発見せよ．
(3) その問題点を解決するためにはどうすれば良いか具体的に記述せよ．
(4) その記述した内容を生成楽曲に適用し，出力せよ．

以上の形式をした英文のプロンプトを GPT に入力することで，先に挙げた傾向をある程度回避した音楽を生成させることができる．

また，GPT は執筆された文章を読んで，その文章に含まれる形容詞を元に執筆者の感情を推定しパラメータとして提示できる可能性が示唆されている．この性質を活用することで，要求事項 (3), (4) に関する制御の容易さを大幅に引き上げることができる．

5. 実施

この章では，第 2 章で取り上げた要求事項のうち，事項 (2), (3), (4) がどれほど達成されるかについて，実際に GPT-4 に第 4 章のプロンプトを与え検証する．そのうちある入力に対して性能が良かった点と悪かった点を取り上げ，考察を行う．図に全ての実施結果がまとまっている．

(2) ユーザの自然言語による指定を反映した楽曲生成

初めに，第四章のプロンプトと入力を与えた際に生成される初期楽譜について検討する．例えば，入力「お日様のように明るく活発な曲を作り出してください.」に対しては図 2(a) の楽譜”Bright Sun”が生成された．

この楽曲 Bright Sun では C 以外のメジャーキーを選択しており，強い緊張を生まないコード進行を主に使っている．このため，楽曲全体については明るい雰囲気に仕上がっており確かにユーザが与えたプロンプトの内容に概ね合致した出力が得られている．このことから，GPT-4 は文章中の形容詞をもとに，それに関連のある調やコード進行を選択する能力があると言える．ただし，問題点として $B\flat$ が周囲のコードとコード進行として合致していない点が挙げられる．この $B\flat$ は，条件指示部のノンダイアトニックコードを利用する制約に従って修正段階の際に挿入されたものである．しかし，その挿入方法が周辺のダイアトニッ

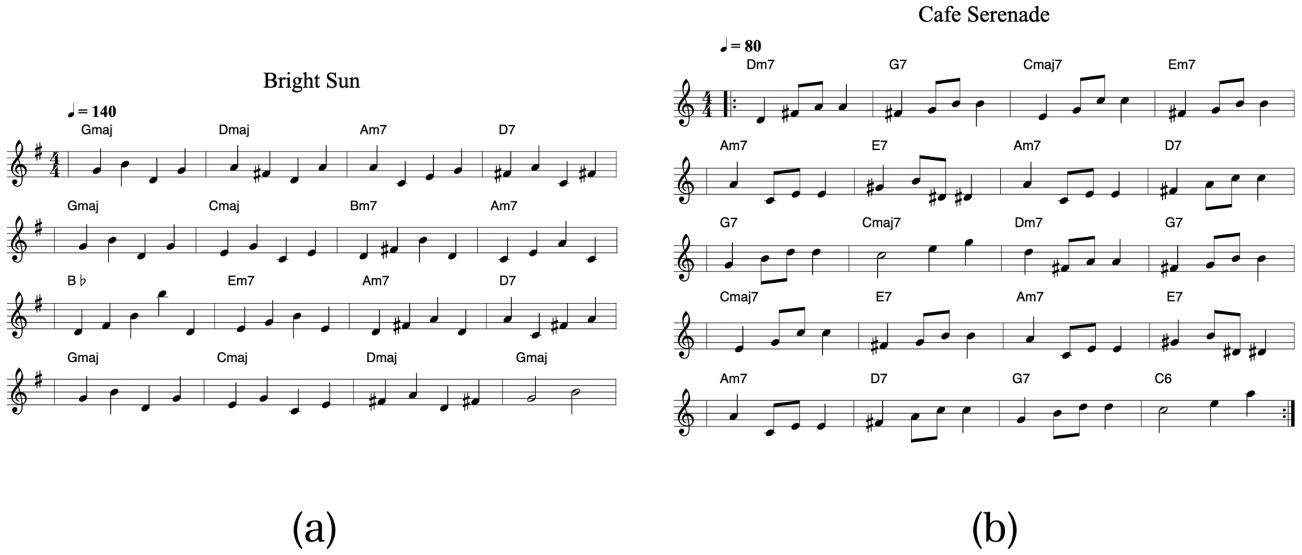


図 2 ユーザの指定を反映した楽曲

クコードを考慮できておらず，視聴者にとって軽い違和感を覚えさせる要因となっている．このノンダイアトニックコードの制約は楽曲の緊張弛緩関係を深める方向に働くこともあるものの，今回の例のように初期生成ではまだノンダイアトニックコード埋め込みの制御が難しい場合がある．そして，指示に音楽理論知識や，関連知識の記述を行うような命令を含めた場合，性能が低下しプロンプトに与えた制約を無視するという現象が多発した．注意機構の注意が与えた知識や自身の記述した知識に行ってしまい，与えた制約を反映しなくなることが原因だと推察される．この問題に対しては，重要な指示を最後に行ったり，プロンプトをいくつかのタスクに分割した上で要約を最後に行わせたり，あるいは Code Interpreter による長期記憶の実現などが対抗策として考えられる．本研究では，重要な指示を最後に行うことで部分的に解決を図った．失敗した事例についても検討する．入力「カフェっぽいゆったりとした音楽を作ってほしい」に対しては図 2(b) の”Café Serenity” が得られた．今回のプロンプトでは「一小節に適切な音価を入れること」という指示を与えており，かつヘッダー上に M:4/4(曲が 4/4 拍子であることを意味する) と記載されているにも関わらず，ほとんどの小節においてそのメロディの音価の合計が 3 拍子分しかない．つまり，一部 3/4 拍子のメロディが混じって生成されてしまっているということになる．一小節に適切な音価を入れるという指示があるにも関わらず反映されていないことを踏まえれば，これは GPT の学習データ内にある 3/4 拍子の曲のデータの影響を受けてしまい，該当する小節部だけ 3/4 拍子だと誤認識されていることが要因だと思われる．そのため，制約に 4/4 拍子のみの曲を作成する制約を加えたり，あるいは 3/4 拍子の際には拍子の切り替えを明示する制約を加えることで解決すると思われる．

(3) 形容詞と関連づけた細かな制御

ここでは，次のような形容詞と 0 から 1 までの中の数値を結びつけて曲の雰囲気を表現する「感情パラメータ」を導入することで，パラメータを通じた音楽的制御がある程度の範囲でできるということを示す．

躍動感: 0 / 1
明るさ: 1 / 1

本稿では，まず GPT-4 にこのパラメータに従って曲を作らせるよう初期楽曲を生成させた上で，(躍動感, 明るさ) を，A:(0, 1) → B:(0.3, 1) → C:(1, 1) → D:(1, 0.3) → E:(1, 0) の順に遷移させ，それぞれの過程でどのように出力楽曲が変化したかについて簡単に述べる．生成された楽譜は図 3 に記載する．

- $A \rightarrow B$: メロディに対して新しい三連桁を数箇所に挿入し，テンポを 80 から 85 へと上昇させた．
- $B \rightarrow C$: テンポを 85 から 120 へと大きく向上させた．
- $C \rightarrow D$: 前状態の楽譜の調を D マイナーへ変換し，コード進行を短調のものへ修正．
- $D \rightarrow E$: コード進行の一部を Bb7 や A7sus4 に置き換え，より緊張を増すように修正．

このように，躍動感と明るさを少しずつ移動させることで，確かに徐々に曲の雰囲気を変え，最後には躍動感の大きさと明るさの大きさを入れ替えることができた．この結果から，パラメータを用いれば生成音楽を少しずつ変化させていくような音楽自動生成 AI の操作形態が実現しうると言える．

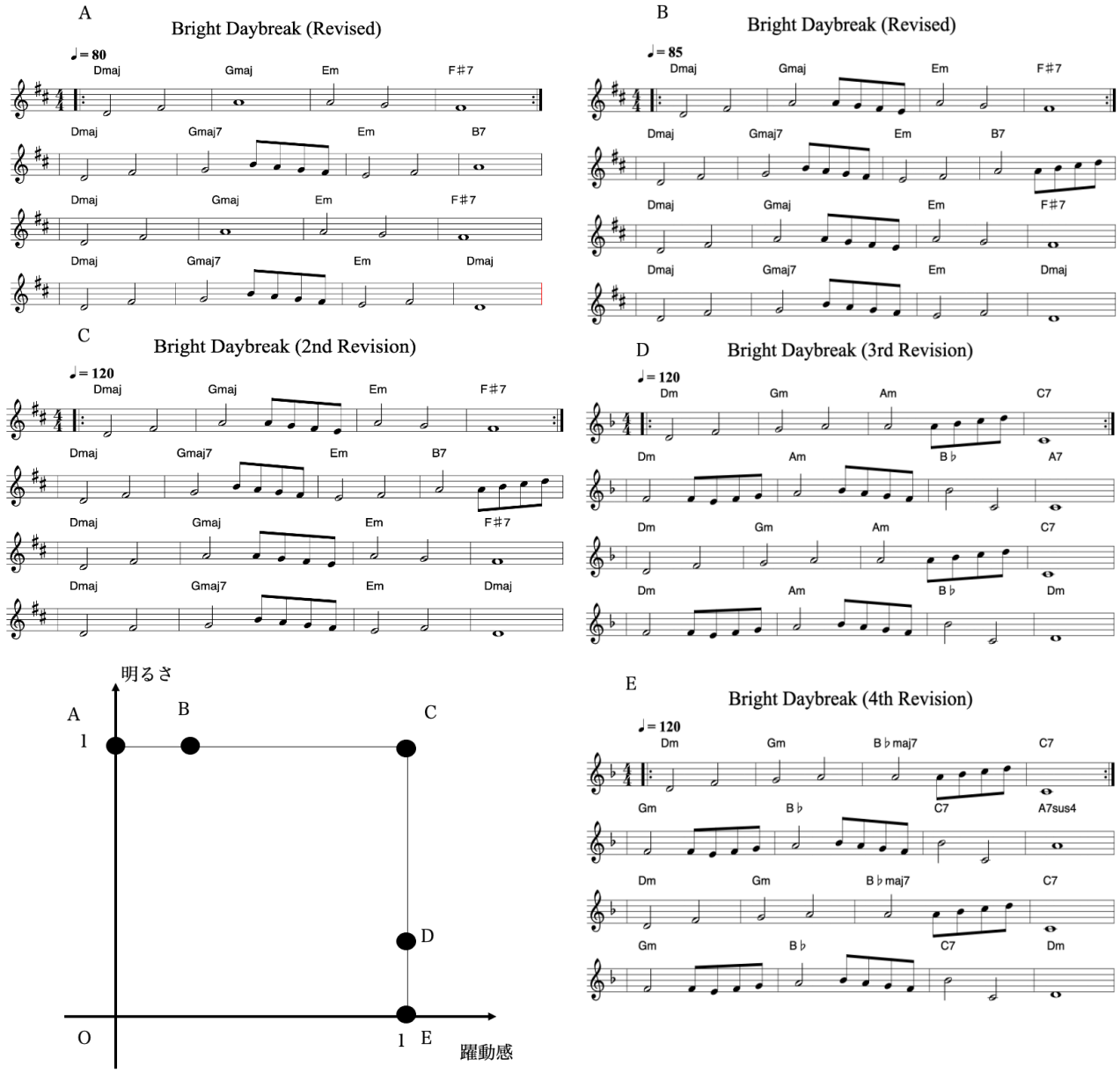


図 3 感情パラメータの変更と出力の変遷

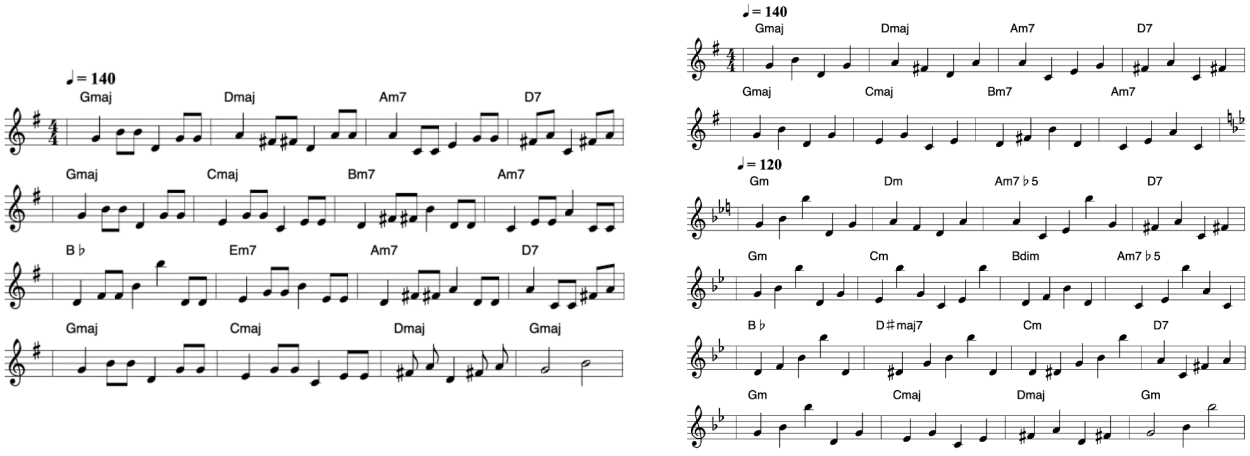
その一方でこの制御システムには課題がある．今回の結果ではパラメータの移動方向自体は適切に反映されていた．しかし，移動の大きさ自体は適切に反映されているかは疑問だ．0.3 程度の大きさで移動した $A \rightarrow B$ ではメロディとテンポに変化があったが，0.7 程度の大きさで移動した $B \rightarrow C$ においてはテンポのみが大きく変化した．2 倍程度の大きさを持つ同方向への移動である以上，この場合にはメロディでも何らかの変化を起こすことが期待される．このことから，複数回パラメータを移動させ続けた時にその移動の大きさまでが相対的に正しく反映することが課題として残る．

また，初期楽曲のテンポがかなり遅かったため，躍動感を最大値 1 としてもテンポが 120 程度の値で止まってしまった．ユーザによってはこのテンポがまだまだ遅いと感じることもある．このような感情パラメータを扱うインタフェースを用意する場合には，それぞれのパラメータの最大値がどんな状態を指し示しているのかをユーザが調節できるようにする必要があると思われる．

(4) 生成された楽曲に関するフィードバック受理と修正

ここでは，図 2(a) で取り上げた楽譜に対してフィードバックを与え，適切な修正が行われるかを確かめる．

初めに，「四分音符が多すぎる．リズムの変化をもっと多様にしてほしい.」というプロンプトを英文で送付した．すると，図 4(a1) に示す楽譜が出力された．この例では，四分音符を八分音符の連桁に分割することで様々な音価を使



(a1) 「四分音符が多すぎる。」　　(a2)「少しだけ曲調を暗く」

図 4　生成された楽曲に対するフィードバックの結果

うように改良された．音符を同一音高の連桁に置き換えるだけではなく，音高の異なる連桁へ置換されたセクションも存在し，傾向 (5) から脱却することに成功したと言える．

その一方で，「ここから少しだけ曲調を暗くしてみてください.」というプロンプトを与えた場合，図 4(a2) に示す楽譜が出力された．全体的にマイナー調に変換され，ノンダイアトニックコードもうまく活用できたアレンジに仕上がっている．しかし，ユーザは「少しだけ」と指定したに過ぎず，このアレンジはその命令に対してあまりにも暗いものとなっている．「暗く」という語に GPT-4 が引っ張られてしまい，マイナーへの転換を行うに至ったのだと考察した．この例から，「少しだけ曲のニュアンスを変更してほしい」というフィードバックに対応することは現状困難だと思われる．この課題に対応するためには，例えば先ほどの感情パラメータの移動量と「少しだけ」といった表現を結びつけるよう指示したり，コード進行の明暗の程度のマッピングを few-shot で例示するといった方法が考えられる．

6. おわりに

本稿では，物語執筆の分野で **directability** を要するタスクが開拓されつつある GPT-4 を活用して **Directability** のある自動楽曲生成のデザインを行うための取り組みについて紹介した．現地点で，GPT-4 を用いた **directability** の実現要求事項を部分的に達成している．ユーザーの指示から調性やテンポ，コード進行を選び出し，適切な音楽を作り出したり，あるいはその生成した楽曲に対してパラメータを操作することで楽曲の雰囲気を変えたり，指示を与えることでより良い楽曲へ改良したりすることが部分的に可能となった．しかし，現状のシステムには，生成される楽曲のリズムの制御が難しく，ユーザーから指示された音楽的操作の程度を実行時に正しく反映できないという課題もある．その改善のために，事前に与えるプロンプトの制約を変化させたり，あるい与えられた文章に対してすべき操作の程度を few-shot learning で調整するといったことが考えられる．

参考文献

[1] Chung, J. J. Y., Kim, W., Yoo, K. M., Lee, H., Adar, E. and Chang, M.: TaleBrush: Sketching Stories with Generative Pretrained Language Models, *CHI '22: Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, Vol. 54, No. 209, pp. 540–545 (2022).
[2] Report, G.-. T.: OpenAI, (online), available from ⟨https://openai.com/research/gpt-4⟩ (2021).
[3] WavTool: WavTool, (online), available from ⟨https://wavtool.com⟩ (2023).
[4] Yakura, H.: GPT 作曲くん，（オンライン），入手先 ⟨https://compose.yumetaro.info⟩ (2023).
[5] 海老惟槻，　白松俊：GPT-3 を用いた歌詞付きコード譜生成をユーザーの意図通り制御する手法の試作，人工知能学会全国大会論文集，Vol. JSAI2023 (2023).
[6] 片寄晴弘：音楽における自動処理と Directability, オペレーションズ・リサーチ = Communications of the Operations Research Society of Japan : 経営の科学，Vol. 54, No. 9, pp. 540–545 (2009).