音楽創作におけるコンピュータと演奏者との関係性についての考察

― Antescofo を応用した自作品の分析を通して ―

大 野 雅 夫

Abstract

"Etude" for piano and live electronics(2015) is my composition using Antescofo, which is a real-time score following and coordination language for computer music composition and performance, and is developed by Arshia Cont in collaboration with Marco Stroppa.

In this composition, Antescofo follows the performer, who controls the tempo. I tried to find a way to make the system work and clarified the problem through analysis.

キーワード……コンピュータ　リアルタイム・スコアフォローイング　Antescofo　Max

1　はじめに

1-1　研究動機

コンピュータ音楽による音楽作品の制作を行っていると、音楽がコンピュータの中で完結することが基本的な前提として当たり前のようになっていた。そこには、コンピュータ音楽の演奏に人間が介在するという選択肢が存在していなかったのである。その点に違和感を抱いてはいたが具体的な手立てが思いつかず、その違和感を解消する突破口がないかと常に探していたところ、最新のテクノロジーとしての Antescofo の存在を知り、実際に自作品 : “Etude” for piano and live electronics (2015)に応用したのが、本研究題目を設定するに至ったきっかけである。

1-2　Antescofo とは

Antescofo とはリアルタイム･スコアフォローイングと作曲と演奏のためのコーディネーション言語(ライブラリ)であり[1)]、Max(後述)のエクスターナル・オブジェクトである。このオブジェクトを用いることによって､Max 上でのリアルタイム･スコアフォローイングが可能になる。

Antescofo は Max のパッチ上で基本的に antescofo~という名のオブジェクトとして機能する。antescofo~単体ではスコアフォローイングの動きを視覚的に確認することはできないが、Ascograph（図 1）というアプリケーションを使用することで、antescofo~の動作をインターフェ

イス上に表示することができる。antescofo~に楽譜のデータを取り込む場合には、最も簡単な手段としては楽譜のデータを MusicXML の形式に変換し、それを antescofo~で開くという方法があるが、その際の問題点は、MusicXML の形式だと antescofo~から Max の「receive」オブジェクトに「send」の機能でその引数を送信したりする際に用いるアクションランゲージを MusicXML のファイル内に保存できない点である。そのため MusicXML で antescofo~に楽譜のデータを取り込む場合には、アクションランゲージは Ascograph のエディタの機能を用い手動で入力するしかない。その問題点を解決できるのが NoteAbilityPro(後述)である。このアプリケーションはノーテーションソフト[2)]であるが、アクションランゲージも内包する antescofo~用の楽譜データのテキストを生成できるので大変効率的である。

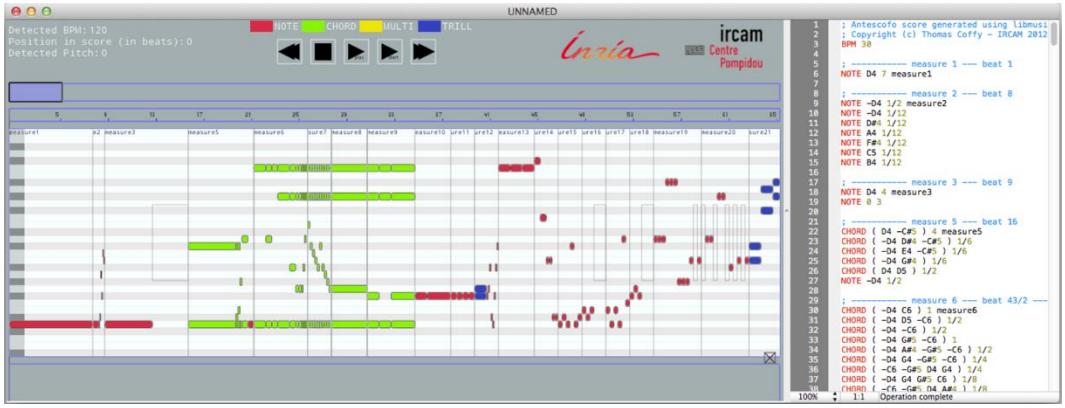


図 1. Antescofo のインターフェイス･エディタとしての Ascograph の画面
（出所）筆者による作成

1-3 Max とは

Max とは、1988 年に IRCAM の Miller S. Puckette によって開発された、ビジュアルプログラミング言語である。Max を用いることにより、従来の、楽譜を用いた演奏者と楽器を通した演奏を前提とした音楽に限らず、MIDI[3)]やデジタル信号処理のテクノロジーによって、音の波形そのものを内部処理し、リアルタイムでスピーカー等から出力することができる。音とは物理的には空気の振動であり、その音の波形と周波数を自由にコントロールできる Max を用いることによって、理論上は、如何なる音色をも作り出すことが可能である。その際の音色は実際に存在する既存の楽器の音色にとどまらず、加算合成・減算合成・AM 合成・FM 合成・RM 合成等が代表する様々な波形合成・音響処理を施すことによって、未聴の音色を作り出すことが可能なソフトウェアである。既存の楽器の音色を用いる場合にはもちろん、MIDI を用いることも可能である。音のダイナミクスやデュレーションも自由にコントロールすることができ、音域やピッチも自由に指定することができる。同時発音数・連続的なアタック・音の跳躍等も基

本的には制限が無い。以上のような機能を持つ Max だが、エクスターナルのライブラリとしてオブジェクトを追加することにより、更に機能を拡張させることができる。その機能の一つが上述した「Antescofo」である。

一般的に知られているテキストベースのプログラミング言語とは違い、Max のようなビジュアルプログラミング言語は、あらかじめコンパイルされたオブジェクトと、オブジェクト同士をパッチコードと呼ばれる線で繋ぐことによってプログラミングを行う。このプログラミング方法だと、一からテキストを入力する必要がなく、作曲家に取って必要な機能がオブジェクトという形であらかじめ提供されているのに加え、パッチコードで繋いでいくという視覚的な方法によりプログラミングを行うので直感的なプログラミングが可能になり、実際の音のイメージとプログラミングの内容とをマッチさせやすいというメリットがある。

デメリットとしては、視覚的なプログラミング方法であるが故に、パッチャー等を用いてプログラムを整理しないと、視覚的に煩雑になり、いわゆるスパゲティコードの状態に陥ってしまう点があげられる。

1-4　Max 上で機能する Antescofo

Antescofo は Max 上で機能するので、基本的に Max との連携が想定された仕様になっている。そのため、Max の noteout のオブジェクトを用いて MIDI の音源を再生したり、アクションランゲージを用い Max の持つ本来のプログラミングの機能と連携させたり、といったことが容易に可能になっている。

1-5　スコアフォローイングと音楽創作との関係性

コンピュータ上でスコアフォローイング[4)]を行おうとする試みは、Antescofo が開発される以前からも行われていた。

それは例えば、Max 上で ctlin オブジェクトを用い MIDI フットコントローラーからテンポやリズムの信号を出したり、デジタル信号処理において音のアタックを検出し音楽のリズムを検出しようとしたりする試みであったが、それらは「楽譜」という概念に対応したものではなく、あくまでも、プログラム上の値としてあらかじめデータを用意しておく必要があるものであった。その点において Antescofo は、スコアを参照しながら、スコア上でコンピュータとの関わりを構築できるので、そういった意味ではただコンピュータ上の操作が簡単になるというだけではなく、作曲家にとってより合理的で、場合によってはより複雑な、人間による演奏とコンピュータの演奏との関係性を構築できるテクノロジーの一つとなる可能性がある。

「楽器としての」コンピュータを用いた作曲活動は 20 世紀後半に入ってから盛んに行われた[5)]が、Pierre Boulez の“Répons[6)]”(1981-1984)で行われたような、生の楽器の演奏とコンピュータ音楽とが組み合わされた作品はあったにせよ、その 20 世紀のいずれの作品においても、コン

ピュータが有機的に(自立的に)実際に演奏される音楽を理解し、生身の演奏者と音楽を共有する技術が確立されることはなかった。21 世紀に入り、Marco Stroppa とのコラボレーションにより Arshia Cont によって開発された、リアルタイム･スコアフォローイングと作曲と演奏のためのコーディネーション言語としての Antescofo により、上記の技術が確立されつつあり、現在もそういったテクノロジーの技術的開発は盛んに行われている(大塚他 2011)(中村他 2013)。

1-6 本研究のねらい

上記のようなテクノロジーが開発されたとして、それが、開発者レベルではなく、一人の作曲者によって実際に創作に応用されるレベルにおいて、これまでは不可能であった「コンピュータを独立させた伴奏者として機能させる」といった音楽的概念を、実際の時間軸を伴った音楽として如何にして具現化することができるのか、その過程における問題点とその改善点をあげつつ、自作品の制作を通して明らかにする。

また、これは Max を用いた創作においても同様であるが、創作の段階においては、テクノロジーと作曲者との関係性は個々人で様々であるため、本論文では Antescofo の一般的な使用法についての方法論の確立を目指すのではなく、あくまで筆者と Antescofo との関係性においての創作活動における使用法について、その実用性を検証する。

自作品において Antescofo のテクノロジーを用いるにあたり、Antescofo を実際に用いた作品創作についての先行研究や、公に発表された作品として、Antescofo の使用を謳っているものは現時点では多くは見かけられない[7]。これまでの筆者の Max 及びライブ・エレクトロニクスに関しての研究に基づいた自作品においては､コンピュータを独立した伴奏者として機能させる､ということは困難であったが、Antescofo を用いることによりそれが可能になった。

本論文は、清水・大野(2015)からの継続的研究として位置付けている。

1-7 先行作品と自作品との相違点

Marco Stroppa、Vassos Nicolaou らによる作品も、自作品も、人間による演奏とコンピュータ上での処理における時間軸をスコアフォローイングによって同期させるという点においては同じだが、それによって Arshia Cont、Vassos Nicolaou らが、リアルタイムで人間による演奏にコンピュータ上の処理でエフェクトをかけたり、一旦録音したものを時間軸をずらして再生したりしたのに対し、筆者はここでは、人間による演奏にコンピュータによって手を加えることはせず、あくまでコンピュータを一人の「演奏者」として捉え、人間による演奏はコンピュータが演奏を行う「きっかけ」として用いることに終始し作品を制作した。この点においては､Arshia Cont、Marco Stroppa らによる作品とは異なっている。何故なら筆者にとって今回は、コンピュータを独立した伴奏者として機能させる、という概念が重要であったからである。

2　Antescofo を用いて創作を行う際の前提条件

2-1　Antescofo を機能させる基本的手順

antescofo~の操作は、antescofo~のインレットに様々なメッセージやシグナルを送ることによって行う。基本的には以下の順序で操作を行うことによりスコアフォローイングが可能になる。

① 演奏音源のシグナルのパッチコードを繋ぐ。
② 「read」のメッセージを送り、MusicXML か Antescofo 用のアクションランゲージを含んだテキストデータを選択し、追従するスコアのデータを読み込む。
③ 「suivi 1」(「suivi $1」にし、toggle を用いてコントロールしても良い)のメッセージを送る。
④ 「start」のメッセージを送る。

この操作を行うことにより、演奏音源と antescofo~に読み込まれたデータによってスコアフォローイングを行う。但し、このためには様々な事前の準備と操作が必要になる。

2−2　設営に関して

antescofo~にシグナルデータを入力する際には、マイクを用い dac~のオブジェクトからシグナルのパッチコードを antescofo~のインレットに繋ぐのが通常的な方法だが、この際のマイクの使用法についても、正確な結果を得るためには工夫が必要である。

まず、マイクは基本的には有線のものを用いた方が効果的である。Bluetooth 等の無線の技術を用いることも可能だが、無線送信技術の仕様や、高負荷によるスペックの問題等により入力される音のデータにレイテンシー(遅延)が生じ易いので、そういった面で有線の方が安定していると言える。

当然だが、antescofo~を用いないクラシックのコンサートでは、楽器のすぐ近くにマイクが設置されているということは録音が行われる場合以外あまり考えられない。今回の自作品ではピアノを用いたが、ピアノの演奏の妨げにならないことが大前提である。楽器の響きに影響が出ないようマイクスタンドを用い、楽器のピアノが直接触れることがないようにした。

また別の問題として、演奏中にマイクでピアノの音を拾う際には、どうしてもスコアフォローイングによる自動伴奏の音源の音もマイクに入ってしまうので、antescofo~のスコアフォローイングの処理にエラーが起こらないよう、マイクの指向性を考慮し、できるだけピアノの音だけが良く入るよう工夫する必要がある。

この問題に対する解決策として、通常のマイクを使用する際には、マイクを出来るだけ楽器(音源)に近づける必要がある。今回(グランドピアノ)の場合は、ピアノの蓋の内側に見えるハンマーのすぐ上に配置した。またマイクの上・横から入ってくる自動伴奏の音源の音を少しでもカットするため、マイクの上に布を被せた。

設営の際には、

① 演奏の妨げにならないマイクの設置と配線
② マイクを出来るだけ音源に近づけ、音源以外からの音は出来るだけ遮断する。

上記二点が考慮されることが望ましいが、超指向性マイクやレイテンシーのない無線技術を用いることが可能な場合、もしくは録音された音源によりスコアフォローイングを行う場合については勿論この限りではない(図 2)。

図 2. セッティングされたマイク （出所）筆者による撮影

2-3 入力シグナルレベルとピッチの調節

スコアフォローイングを行う楽器の音源は、マイクによって入力され、dac~からシグナルデータとして antescofo~に送られるが、このシグナルの音圧のレベルと、基本となるピッチが Antescofo 上で設定された正常な値に調節(チューニング)されていないと、スコアフォローイングの正確な結果を得ることができない。

シグナルレベルとピッチについては、Antescofo の calibrate の機能を用い視覚的に確認することができる。ここではシグナルレベルは時系列に沿って生成されるグラフによって表示され、ピッチはリアルタイムに変化するメーターによって表示される。シグナルレベルは音のアタックの検出にも関わるので、時系列上で確認できるのは便利な機能である。

シグナルレベルを適正な値に調節するためには、「どこかの段階で」何らかの調整を行うことが必要である。考えられるのは、

1．マイクで音を拾う段階
2．オーディオインターフェイスでデジタルデータに変換する段階

３．Max 上で dac~からシグナルデータとして出力された段階

４．antescofo~に入力された段階

以上それぞれの段階だが、音源が動くこと(ピアノの場合は、叩く鍵盤の場所が変わればそれに伴って必然的に音源も移動する)、一定の音量で演奏されるとは限らないこと(p や f、フレージング、イントネーションなどの音楽的な演奏上の強弱変化)また、それらに伴い自動的、連続的にシグナルレベルの調節を行う必要性があることを考慮すると、今回は dac~から出力されたシグナルデータを Max 上の処理で自動的に適正なシグナルレベルに調節し、antescofo~に入力するのが望ましいと考えられる。

そのような処理を行う Max 上のパッチのプログラミングの方法としては様々な手段が考えられるが、自作品の演奏の上で、筆者自身がプログラミングを行ったパッチが図 3 である。

このパッチは、現在のシグナルのレベルが適正レベルと比べて低いか高いか判断し、低ければレベルを上げ、高ければレベルを下げるという処理を自動で、連続的に行うパッチである。

チューニングについては演奏中でなく、事前の準備段階で調整しておく必要がある。そもそも、チューニングを行うピッチを変更したい場合は、antescofo~の tune の機能を用い、チューニングする周波数を指定することができる。

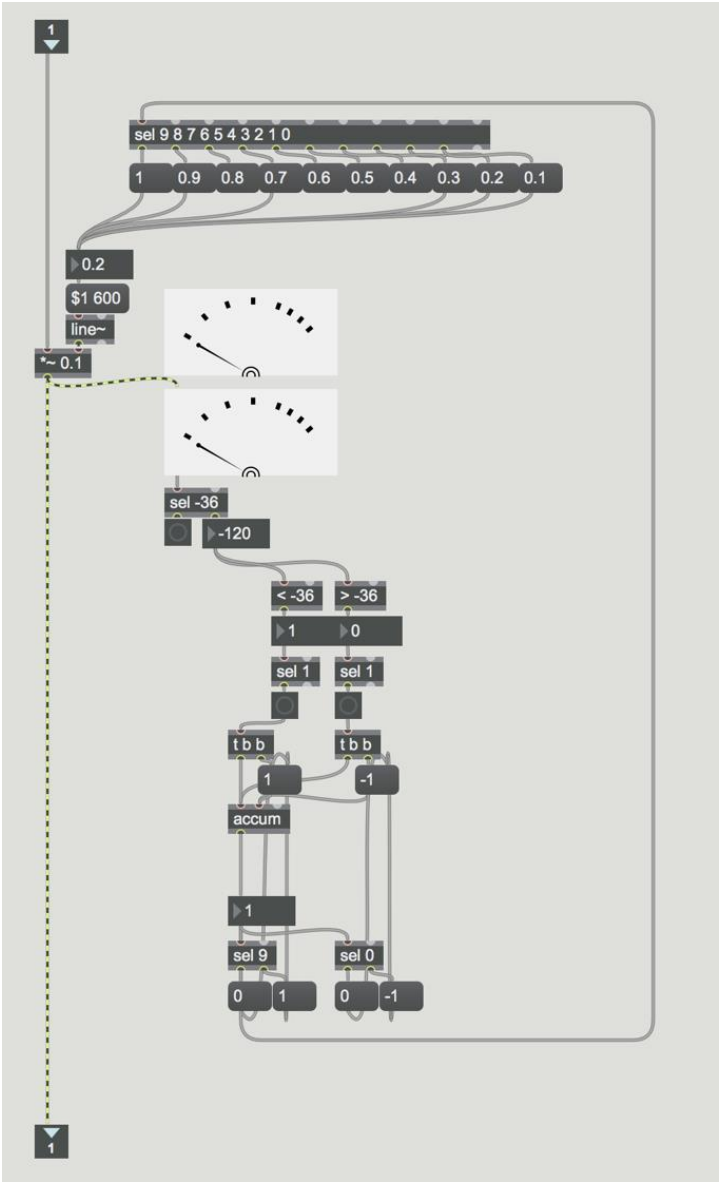


図 3. シグナルレベルを調節するパッチ（出所）筆者による作成

2−4　NoteAbilityPro を用いたアクションランゲージの作成

NoteAbilityPro とは、Antescofo 用のアクションランゲージを含んだテキストデータを作成することができるノーテーションソフトである(図 4)。また、アクションランゲージは、antescofo~がスコアフォローイングを行う際、Max の「send」「receive」の機能を用いて antescofo~から「メッセージ」を送る際に用いる。

既述した通りアクションランゲージは、NoteAbilityPro を用いなくとも Ascograph 上で手動により入力することも可能だが、NoteAbilityPro の GFWD Note Editor(図 5)の機能を用いることにより、アクションランゲージを書き込むことが可能になるので、こちらの手段の方が格段に効率的であると言える。但し全自動ではないので、ある程度の手作業が必要となる。Antescofo の技術を用いたスコアフォローイングによる自動伴奏を実現するには、どの音を「きっかけ」とし、それに対しどのような「反応」をするかという指示は、全て事前に人手によって指示しておく必要がある。この点が、人間とコンピュータとの認識の境界線とも言える。

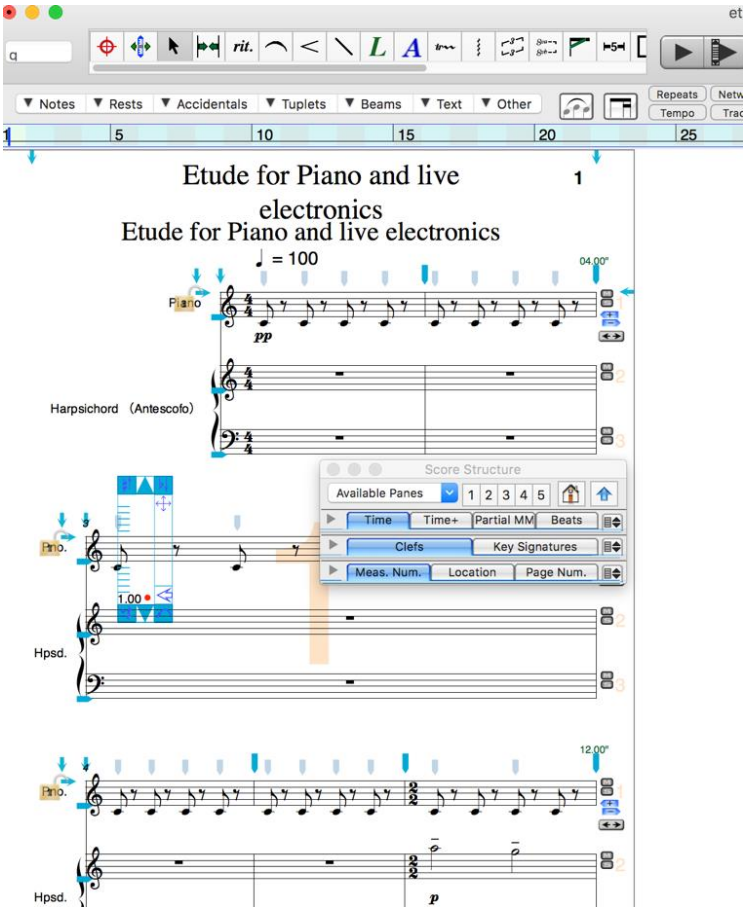


図 4. NoteAbilityPro のエディタ（出所）筆者による作成

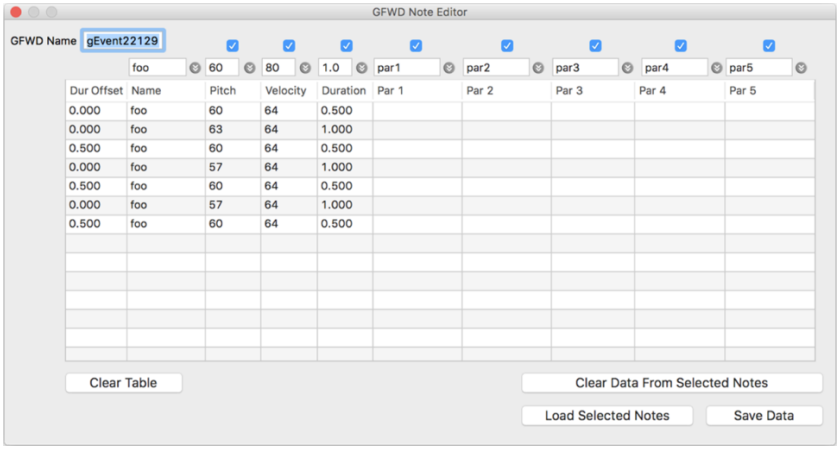


図 5. GFWD Note Editor（出所）筆者による作成

2-5　演奏音源の用意と出力先の選択

今回の自作品では Antescofo を自動伴奏者として用いた。伴奏者として用いるためには音源を用意する必要があるが、Mac の内臓の MIDI 音源では、求めていたリアリティのある音源とは言えなかったので、今回はノーテーションソフトの Sibelius に搭載されている音源を用いた。

再生する音源の MIDI のノートナンバーやベロシティー、デュレーション等は Max 上で antescofo~から receive のオブジェクトを用いることにより受け取ることができるので、その値を noteout のオブジェクトに送り、出力先を仮想の「from Max 1」に指定し、Sibelius 上で「from Max 1」を入力装置として指定することにより、Sibelius の音源を使用することができる。Windows では noteout からの仮想の出力先が Max の機能としては用意されていないので、別途設定を行う必要がある。

3　自動伴奏者として用いる Antescofo の自作品への応用

3-1　自作品：“Etude” for piano and live electronics (2015)について

3-1-1　作曲経緯

この作品は、2015 年 8 月 6 日に行われた新潟大学教育学部音楽科における作曲セミナーでの発表のために作曲された。

3-1-2　作品概説

編成はピアノとハープシコードになっており、人間がピアノのパートを演奏し、Antescofo がハープシコードのパートを演奏する。この二つのパートの関係性としては、人間の演奏するピアノパート(連続する一点ハ音)に Antescofo によるハープシコードパート(伴奏付け)が合わせ

る、といった一方的な関係性となっている(図 6・図 7)。また、ハープシコードパートの伴奏の元になっているフレーズの作成には Max のパッチを用いた。

エチュードという言葉には様々な解釈あるが、今回は主に２つの意味を込めている。１つはリアルタイム・スコアフォローイングという自分にとって新しい技術への挑戦の意味、もう１つは生身の演奏者とコンピュータとのファーストコンタクトのための技術的な練習のための作品という意味である。

図 6. 人間と Antescofo との一方的な関係性（出所）筆者による作成

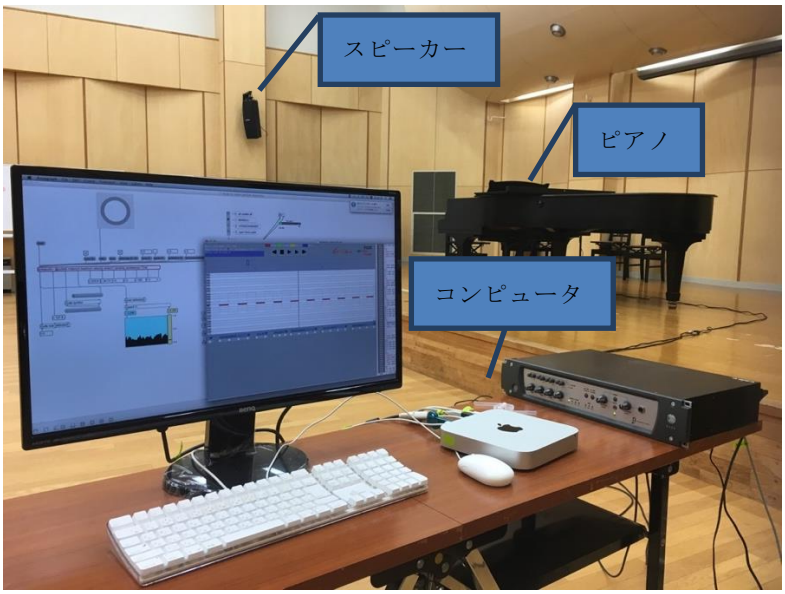


図 7. ピアノとコンピュータとスピーカー（出所）筆者による撮影

3-2　作品のアイディアと構成

3-2-1　作品のアイディア

この作品の端緒は「曲のテンポを司る人間とそれに追従するコンピュータ」という構図を明確に打ち出そうとするアイディアからである。

演奏におけるテンポを明確にするため、人間の演奏するピアノパートは、曲の始めから終わりまで一点ハ音で均等なリズムを刻み続ける。テンポに関しては Tempo rubato として、演奏者による即興的なテンポの動きを認めている。そのテンポの動きに対してハープシコードパートを追従させるために Antescofo のテクノロジーを用いている。ハープシコードパートはピアノパートとは対照的に、比較的複雑な音型となっている。拍が意図的に半拍ずらされたり、裏拍から連符が始まったりしている箇所などは、ピアノパートが表拍で一定のリズムを刻んでいるということもあり人間が演奏する場合には正確に拍感を感じることが非常に難しくなるが、Antescofo での処理においてはそのような点について考慮する必要はない。またその点が Antescofo による演奏の特徴であるとも言える。テンポの動きに対する追従に関しても同様である。

3-2-2 構成

全体の構成として、この作品は以下の６つの部分から成っている。

序奏 – A – B – A’ – B’ – 後奏

以下それぞれの部分の役割について簡単に述べる。尚、A’B’は A B のヴァリエーションであるため本項での言及については省略する。

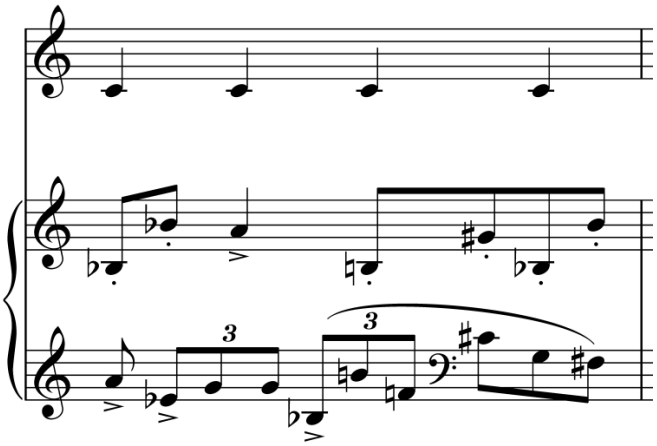
3-2-2-1　序奏

序奏の始めの部分においては、ハープシコードパートは演奏せず、ピアノパートのみが演奏し一定のリズムで一点ハ音を刻み続ける。これは、少ない音数からだんだんと音の数が増えていくという、音楽的な曲の仕組みとしての意味もあるが、Antescofo での処理の上で「曲のテンポを確定させる」ために一点ハ音を刻み続けるという技術的な意味もある。ピアノパートはこの作品の全体にわたって一点ハ音を刻み続ける。テンポが確定した後にハープシコードパートが入り、最初はゆったりとした二分音符によるリズムを刻む。その後音価が八分音符に変わり、拍子のある部分に移行し A の部分に入る。

3-2-2-2　A

A はハープシコードの左手と右手のパートがそれぞれ別のフレージングで演奏され、またピ

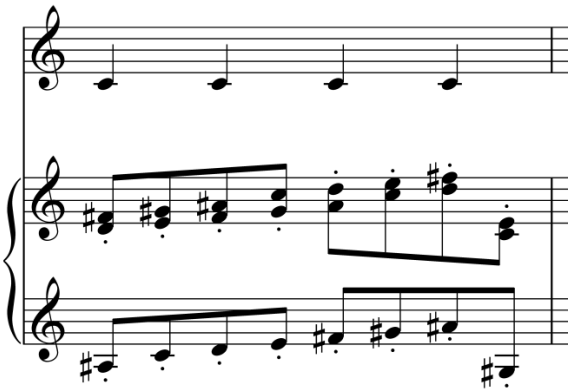
アノパートとも拍がずらされている箇所が多く、場合によっては譜例 1 のように比較的複雑なリズムになる箇所もあるが、上述したようにそのような複雑性は Antescofo の処理においては演奏の妨げになるものではないので、そのような箇所もコンピュータとして、あくまで正確なリズムで演奏される。



譜例 1. ピアノパートによる連続する一点ハ音とハープシコードパートによる伴奏付け
（出所）筆者による作成

3-2-2-3　B

B は A とは対照的に、ピアノパートとハープシコードパートで拍がずらされている箇所があまりない。そのような場合では、ピアノパートのテンポの動きがハープシコードパートの演奏に如実に反映され聴こえることとなる。B ではそういった、ピアノパートがテンポの変化により、ハープシコードパートをコントロールする表現を行なっている(譜例 2)。



譜例 2. ピアノパートと拍がずれていないハープシコードパート（出所）筆者による作成

3-2-2-4　後奏

後奏では、序奏と逆の行程を行なっている。音の数も多く、音量的にもピークを迎えた B’から、だんだんと音の数を減らし、長い音価の音符を用いたゆったりとした曲の終わりへと、序奏よりも時間をかけて移行させている。そうすることによって、曲の始めから一定のリズムでずっと鳴り続けていた一点ハ音が、曲の終わりに止まった際の感覚を強調させるよう工夫している。

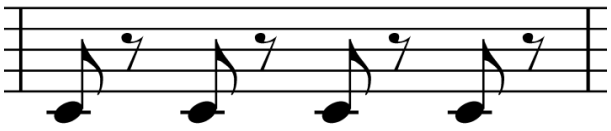
3-3　作品における技術的制約とその解決方法

楽譜に記譜されたこの音楽を実際に演奏する際に必要な Antescofo での処理における技術的課題は、楽譜を Antescofo 用のデータに変換することと、ピアノパートの実際の演奏を元にして正しくスコアフォローイングを行うことである。

楽譜を Antescofo 用のデータに変換することは、2-4 で述べた NoteAbilityPro の GFWD Note Editor を用いることにより、比較的容易にクリアすることができた、しかしこの作品においては、ピアノパートの音声の認識が困難であった。

この作品においてピアノパートはテンポを司っているため、四分音符の連続で尚且つピッチが変化しない。人間がこのような演奏を聴けば、当たり前のように一つ一つの音のアタックを認識し、たとえそれが連続していようと最初の音とその次の音とを認識する。テンポが四分音符＝100 の場合で、四分音符が連続されて演奏される場合、ハンマーによって打弦された音は、次の四分音符までに完全には減衰しない。つまり音量的にはアタックの音量とさほど変わらないまま四分音符が連続していくのである。Antescofo はこの微妙な音量の減衰を認識しなかったため、四分音符の連続においてそれぞれの音のアタックを認識することができなかった。ピッチが変わっていないというのも一つの要因である。

この Antescofo の技術的制約の問題を解決するためには、NoteAbilityPro を用いて Antescofo 用のデータを作成する際、譜例 3 のように音符と音符の間に休符を挟んだ。このように休符を挟むことにより、Antescofo 上の処理としては、減衰を考慮することなくアタックを検出することができるので、結果として正しくスコアフォローイングを行うことが可能になった。



譜例 3．八分音符と八分休符の連続（出所）筆者による作成

4　総括及び今後の展望

以上で述べたように、コンピュータを一人の「伴奏者」として機能させるためには、いくつかの課題があった。

人間であれば、ある程度の喧騒の中でも、その中からピアノの音を聞き取るということは可能である。また、大きい音であれ小さい音で聴取できればそれを旋律として認識することができる。アタックの検出についても、四分音符が連続しており、たとえ間に休符がなくとも、それを周期的なリズムであると認識することは容易に可能である。しかし、これらのことをコンピュータが認識できるようにすることは容易ではない。その問題を解決するための、Antescofo を用いる、という手段の有効性について、本論文を通して確認することができた。

Antescofo を用いて自作品を製作するにあたっては、マイクへの音声の入力の問題、シグナルレベルとピッチの調整の問題、アタックの検出に関する問題等様々な課題が出てきたが、本論文における様々な解決策を講じることにより、自作品の制作に成功した。

今回は Antescofo を用いることにより、コンピュータを自動伴奏者として機能させたが、今後の可能性としては、人間の「共演者」としてのコンピュータとの、よりインタラクティブな関係性の構築が考えられる。人間と「伴奏者」としてのコンピュータとの関係性は、本論文で述べたように、人間の演奏をコンピュータが認識するという一方的な関係性であったが、コンピュータの演奏にも何らかの可変的な要素を加えることにより、コンピュータの演奏も、人間の演奏に何らかの影響を与え得るのではないかと考える。

また、Antescofo はもちろんその性質上、Max の機能を用いることにより機能を果たすことが可能になっているので、Max の機能を用いる、という点に着目すると、また他の可能性も考えられる。今回の自作品においては、人間による楽器の演奏をきっかけに Max 上で、MIDI の値を外部のソフトウェアに出力したが、 Max そのものの機能を用い、あらかじめ Max 上で用意しておいた波形データを音源として用い再生するということも可能である。その他にも、演奏に合わせ、ある時点まで演奏が進むと、事前に録音しておいた音源が再生されたり、ある音をきっかけに Max で用意しておいたパッチャーを作動させ、Max 上の処理による音楽が始まったり、といったことが可能性としてあげられる。

今回の自動伴奏者としての Antescofo の使用方法はそのような可能性の内の一つであり、また、コンピュータを一人の演奏者として扱うというのは一見シンプルな概念だが、その実現過程において、音楽創作における様々な可能性が見えた。今後も自作品制作を通して、音楽創作におけるリアルタイム・スコアフォローイングの可能性について継続して研究を行う必要がある。

＜注＞

1) Ircam Forum“Products-Antescofo” http://forumnet.ircam.fr/product/antescofo-en/ (2017 年 1 月 14 日アクセス)
2) コンピュータ上で楽譜データを作成するソフトウェア
3) Musical Instrument Digital Interface
4) 実際の演奏に合わせて、スコア上のその演奏箇所をコンピュータ上で自動的に追従していく機能
5) Iannis Xenakis(1978)“Mycenes alpha”、湯浅譲二(1991)「UPIC のための『始原への眼差し 第一番』」等
6) 「レポン」と読む。「応答」「答え」の意。６人のソリスト、室内オーケストラとライブ・エレクトロニクスのために作られた曲である。コンピュータの操作は人間が行う。
7) Antescofo 開発者らによる作品、Marco Stroppa(2007)“… of silence”、Vassos Nicolaou(2009)“Otemo”等がある。

＜引用文献＞

大塚琢馬、中臺一博、高橋徹、尾形哲也、奥乃博(2011)「累積頻度重みを適用したパーティクルフィルタによる実時間楽譜追従」『第 73 回全国大会講演論文集』情報処理学会

中村友彦、中村栄太、嵯峨山茂樹(2013)「誤り・任意の弾き直し・引き飛ばしを含む演奏音響信号への高速な楽譜追跡」『情報処理学会研究報告』Vol.2013-MUS-99、No.40

清水研作、大野雅夫(2015)「コンピュータ音楽とその作曲への応用」『新潟大学教育学部研究紀要 人文・社会科学編』第 8 巻第 1 号、PP.81-94

主指導教員（清水研作教授）、副指導教員（伊野義博教授・田中幸治准教授）