University of California
Santa Barbara

Composing (with) Interfaces: Analog and Digital Feedback Loops and the Compositional Process

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Music

by

Marc Paul Evanstein

Committee in charge:

Professor Clarence Barlow, Chair
Professor Curtis Roads
Professor Benjamin Levy
Professor Joel Feigin

September 2019

The Dissertation of Marc Paul Evanstein is approved.

Professor Curtis Roads

Professor Benjamin Levy

Professor Joel Feigin

Professor Clarence Barlow, Committee Chair

September 2019

Composing (with) Interfaces:

Analog and Digital Feedback Loops and the Compositional Process

by

Marc Paul Evanstein

Dedicated to the Sport of Volley-Pong

Acknowledgments

The work presented here owes a tremendous debt to my composition teachers at UC Santa Barbara: Joel Feigin, Clarence Barlow, and Curtis Roads. From Joel, I learned to treat each new piece with humility, asking it what it needs and listening to the answer. From Clarence, I learned to lean into all of my crazy ideas and to follow each to its logical conclusion. From Curtis, I came to appreciate that the process of trimming is a creative, rather than a destructive, act. Most of all, I am grateful to have had such different musical perspectives coexisting so congenially and with so much mutual respect.

An enormous part of my education has come from my close friends and colleagues in the graduate composition program. We come from very different backgrounds, have very different experiences and perspectives, and make very different music. Much of what is written here has been influenced either directly or indirectly by our conversations.

I would like to express my gratitude as well to the many other professors and staff members at the UC Santa Barbara Departments of Music and of Media Arts and Technology who have made my time here so enjoyable and enriching: Charles Asche, Benjamin Levy, Karl Yerkes, Marko Peljhan, Carly Yartz, Adriane Hill, and Anthony Garcia, among many others.

Finally, my parents and family, who have always encouraged me to choose my own path, my wife Emily, and our cat Minuet, are the unseen champions behind all of my efforts, big and small. I love you guys.

Curriculum Vitæ

Marc Paul Evanstein

Education

2019	Ph.D. in Music (Expected), UC Santa Barbara.
2019	M.S. in Media Arts and Technology, UC Santa Barbara.
2018	M.A. in Music Composition, UC Santa Barbara.
2011	M.A. in Music, Science and Technology, Stanford University.
2010	B.A. in Music, Stanford University.

Employment

2018–19	Teaching Associate in Music Composition, UC Santa Barbara.
2017	Director, UCSB Summer Music Festival.
2016–17	Teaching Assistant in Musicianship, UC Santa Barbara.
2014–15	Teaching Assistant in Piano, UC Santa Barbara.
2011–13	Academic Tutor, College Track, East Palo Alto, CA.

Publications

2019	"SCAMP: Suite for Computer Assisted Music in Python," Unpublished thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Media Arts and Technology, UC Santa Barbara, 50 pp.
2019	"Clockblocks: A Pure-Python Library for Controlling Musical Time, *Proceedings of the 2019 International Computer Music Conference*.

Awards

2019	Outstanding Graduating Graduate Student, UC Santa Barbara Department of Music.
2019	Outstanding Graduate Service Award, UC Santa Barbara Department of Music.
2013–18	Chancellor's Fellowship, UC Santa Barbara.
2017	Composition Fellow, Atlantic Music Festival.
2015	Composition Competition Winner, Bowdoin International Music Festival.
2010	Patrick Butler Undergraduate Prize in Piano Performance, Stanford University.
2009	Humanities & Sciences Undergraduate Prize in Music Composition, Stanford University.

Abstract

Composing (with) Interfaces:

Analog and Digital Feedback Loops and the Compositional Process

by

Marc Paul Evanstein

Most, if not all, composing is done through some form of interface, in which the composer makes modifications to, and receives feedback about, the composition in progress. Such interfaces can take a wide variety of forms, including both analog interfaces, such as pencil and paper, and digital interfaces, such as notation software and Digital Audio Workstations. This document presents a theory of these 'compositional interfaces,' and of their effect on the compositional process. Starting with a review of the literature about interfaces in general, and musical interfaces more specifically, the anatomy of the compositional interface is then examined, with a detailed discussion of forms of input, representation, and feedback. Important properties of the interface as a whole—such as latency, openness, and scope—are then discussed, after which two case studies are analyzed. In the final chapter, the role of interfaces is considered within the context of the author's own work.

Contents

Chapter 1

Introduction and Outline

During my first lesson with Prof. Clarence Barlow—an early pioneer in algorithmic composition and in the design of generative computer music systems—I opened up my laptop to show him a new program I had created. The application featured a graphical interface for creating networks of interconnected pitches and durations; when executed, it would chart a path (or several parallel paths) through the network, exporting the result as a score in MusicXML format[1].

I was excited to show him all of the program's bells and whistles: the ability to have multiple layers of networks running simultaneously, the system of cues and triggers that could be used to coordinate multiple voices, the ability to use variable expressions instead of fixed values for pitches and durations, etc. I had spent a great deal of time expanding the program's capabilities, in the way that one does when one is not entirely sure which capabilities are truly important.

After I demonstrated all of this, Barlow responded with a simple question: "Could you make it so that it plays back immediately, rather than having to export a file and open it up in Sibelius?" He also suggested that the nodes might light up as they played, so that we could see, as well as hear, the path that the program was taking through the

[1]Essentially, it was a program in which you could design your own first-order Markov chains, specify starting points for the various voices, and generate the resulting music.

network.

It wasn't the advice I was expecting, but in retrospect it was a profoundly important suggestion. If I truly wanted to use what I had built for writing music, it would mean engaging with a very particular workflow:

1. Design/adjust the network.
2. Run the program and export a MusicXML file.
3. Leave the program and find the generated file.
4. Open that file in Sibelius.
5. Listen to and evaluate the result.
6. Repeat steps 1-5 until satisfied.

What Barlow's comment made me realize was that, while I had been focused on the interface *within* my computer program and on the possibilities that it afforded, I had utterly neglected the overarching feedback loop in which it was embedded. Despite the fact that my main mode of evaluation was auditory (hearing the playback from Sibelius), that loop included the generation of notation, resulting in extra, unnecessary steps that slowed down my workflow[2]. Moreover, beyond issues of efficiency, I was limiting the music my program could produce to that which could be easily notated.

If time were an unlimited resource, and if western music notation were a neutral, lossless translation of musical content, such considerations might not matter all that much. However, given these conditions are both far from the truth, it becomes crucial to consider how the interfaces involved in the compositional process—the tools we use, the feedback loops we engage in—affect the music we produce.

[2]This is not to say that I wouldn't *ultimately* want to generating notation. It just didn't need to be embedded in every iteration of the composing process.

All music is composed via some kind of interface. These interfaces are manifold, varying by culture, genre, medium, composer, and individual composition. Many interfaces are so basic as to escape our attention; others are more visible and highly elaborate. Some analog interfaces have been used in much the same way for centuries; other digital interfaces undergo constant change and development. Far from mere technical contrivances, all of these interfaces reflect the values with which composers approach the process of composition.

The following chapters represent an investigation into the many questions surrounding such interfaces. We begin in Chapter 2 with a review of the research on interfaces in general, and on musical interfaces more specifically. Chapter 3 then proceeds to outline a practical theory of what I term the 'Compositional Interface,' modeling the process of composition as a feedback loop, examining each element of that loop systematically, and then considering several important general properties. Finally, Chapter 4 considers this model within the context of my own compositional practice.

Chapter 2

Interfaces and Creative Work

2.1 What is an Interface?

2.1.1 Interfaces as Translators

Steven Johnson, writing at the cusp of the 21st century as the digital revolution was coming into full force, defined an interface as something that "serves as a kind of translator, mediating between two parties, making one sensible to the other" [1]. Immediately we see the relevance to music: music-making, and the formation of musical meaning, often involves many layers of translation, and each point of translation is both a potential impediment and a potential opportunity.

For instance, in programs that output western music notation (such as my own program described in the introduction), ideas are forced into discrete expression in both pitch and time[1]. In order to be notated, a note 1.27 seconds long must (unless at a very specific tempo) be approximated by multiple tied rational durations. Likewise, unless great care is taken in the translation process, microtonality and glissandi become deeply awkward to notate and to read[2]. Such translations can destroy the integrity of an idea

[1]Interestingly, this was not as true when western notation was first coming into being: the gestural nature of early neumatic notation suggests continuous rather than discrete information.

[2]It was exactly these kinds of issues of translation between the continuous and discrete musical domains that inspired my creation of SCAMP, a Suite for Computer-Assisted Music in Python[2].

that originated in a more continuous context. Perhaps more insidiously, the anticipation of such translations can lead a composer to avoid musical ideas that do not lend themselves to easy discrete expression.

On the other hand, there is great opportunity in translations of a more poetic nature. For instance, to take a non-musical example, Roberto Benavidez' "Piñatas of Earthly Delights" re-renders the bizarre creatures from Hieronymus Bosch's famous "Garden of Earthly Delights" as meticulous, life-sized Piñatas [3]. Here, the process of translation (from small to large, from 2D to 3D, from paint to Piñata) creates new layers of poetic meaning, and an entirely new spatial experience.

2.1.2 Interfaces, Technology, and Dimensionality

Although the term 'user-interface' immediately conjures up images of websites and computer programs, the term 'interface' can, and will here, be applied much more broadly. I take some inspiration in this regard from the Technology-Related Assistance for Individuals with Disabilities Act (or "Tech Act"), where technology is defined as "any item, piece of equipment, or product system (whether acquired off the shelf, modified, or customized) that is used to increase, maintain, or improve the functional capabilities of a child with a disability" [4]. In the same way that a pencil grip can be considered instructional technology, different-sized pieces of music paper can be considered different interfaces for composition.

That said, interfaces do play a particularly vital role where computers are concerned, since these layers of translation are the only thing making the vast complexity of such systems comprehensible. As Johnson writes [1]:

> Our only access to this parallel universe of zeros and ones runs through the conduit of the computer interface, which means that the most dynamic and

> innovative region of the modern world reveals itself to us only through the anonymous middlemen of interface design.

This points to one of the key roles of interfaces in modern life: the collapsing of unwieldy, high-dimensional spaces into lower-dimensional spaces that can be more easily comprehended an explored. Indeed, as we will see, this is an important role played by musical interfaces. (Every time I compose, I am reminded just how astoundingly complicated and multi-dimensional the space of musical possibilities is.)

2.1.3 "Doing, Feeling, and Knowing"

In his "Interaction Design Sketchbook," Bill Verplank describes the interaction between a human and an interface as consisting of three main elements: "doing, feeling, and knowing"[5]. Thus, in order to understand any given interface, we must pay attention to the paths of action, the paths of perception, and to the conceptual model that bridges the two. This framework will serve us well in the ensuing discussion of compositional interfaces in Chapter 3.

The conceptual model linking perception and action often takes the form of one or several metaphorical mappings. For instance, the "desktop" and "window" metaphors have become ubiquitous in personal computers. We employ the metaphors unthinkingly, organizing icons in virtual space, opening and closing windows, throwing things into a trash can that we later empty. It could well have been different: for instance, Apple briefly experimented with a 3D navigational structure in which files were organized as a galaxy of stars and planets [1]. Nowadays, however, the window and desktop metaphors have become so ingrained that many of us think of them as part and parcel of what a computer *is*, forgetting that the original human computer interfaces were textual[3].

[3]Interestingly, mobile devices organize the same hardware according to somewhat different conceptual metaphors, which is one of the reasons why they do not quite feel like "computers."

The use of one metaphor over another can have a profound impact on how we approach an interaction. For instance, in their seminal work, "Metaphors We Live By," Lakoff and Johnson note the abundance of expressions in the English language that establish the conceptual metaphor of *argument as war*: we speak of "attacking" someone's "weak point," of "indefensible positions," of a criticism being "right on target" [6]. They then invite the reader to consider the effect of a different kind of metaphor:

> Try to imagine a culture where arguments are not viewed in terms of war, where no one wins or loses, where there is no sense of attacking or defending, gaining or losing ground. Imagine a culture where an argument is viewed as a dance, the participants are seen as performers, and the goal is to perform in a balanced and aesthetically pleasing way. In such a culture, people would view arguments differently, experience them differently, carry them out differently, and talk about them differently.

It is not hard to make the leap from this description to the realm of musical expression. Instrumental teachers, for example, frequently use metaphors to convey complex physical motions (e.g. comparing the sensation of resistance in a violinist's bow arm to that of dragging the arm through water). As we interact with metaphors in performance, instrument design, and composition, what effect are they having on the music that results?

2.1.4 Expression: Buttons vs. Handles

Verplank divides input controls into two main categories: *buttons*, which represent discrete control, and *handles*, which represent continuous control. For instance, a light switch is a button, as it alternates between two discrete states, whereas the heat knob on

a stove is a handle, since it offers a continuous range of settings. Of particular importance to music are the different functions that these two types of control typically offer [5]:

> Buttons are more likely symbolic. Handles can be "analogic". With buttons, I am more often faced with a sequence of presses. With a handle a sequence becomes a gesture. I use buttons for precision, handles for expression.

Gesture is a central concern in musical expressivity, so the incorporation of handles is often of great value to musical interfaces. However, even instruments that seem to consist exclusively of buttons—such as the harpsichord—can be expressive. How is this possible? The reason is that time is the ultimate handle, the ultimate form of continuous control[4].

Interestingly, the use of language as an input to an interface seems to defy easy categorization into button or handle, to the point that it might therefore be considered an entirely different category. The associative power of language, and the ability of words to evoke a web of connotations, makes it a tremendously powerful expressive tool, one which we will come back to in Chapter 3 in our discussion of compositional interfaces.

2.2 Musical Interfaces

Turning our attention now to musical interfaces, much of the research that has been done has focused on the study of real-time musical expression and of musical instruments (both traditional and newly created). While instrumental performance differs in significant ways from composition, there is much we can learn from these investigations.

For instance, in a study of the relationship between technique and expression at the piano, McPherson et al. demonstrated that, while the mechanism of the piano seems to

[4]Unless, of course, it is quantized.

only provide two dimensions of musical expression—key velocity and timing—performers are consistently operating in a higher-dimensional conceptual space [7]. By analysing the *continuous* key position and velocity data produced by an optical sensor (as opposed to merely the note onsets and velocities), they found that pianists were, in fact, manipulating at least 5 or 6 independent dimensions, including velocity, percussiveness, rigidity, weight, and depth.

This accords with my own experience as pianist; indeed, every pianist and piano teacher I have known sees the space of touch and gesture as significantly more intricate than the mere combination of timing and key velocity. While the extra dimensions that McPherson et al. have identified may not express themselves in the sound of a single note[5], I believe that they come into play quite significantly in the execution of multi-note gestures. Moreover, when the visual presence of the performer is taken into account, perception may be affected via a musical version of the McGurk effect [8].

Accepting that the higher-dimensional space in which a pianist feels herself to be operating affects the musical result, we have found a clear example in which it is not merely the modes of input and output of a musical interface that matter, but also the mental models with which we approach the interaction.

2.2.1 High-Tech Musical Interfaces

The question of interface design is perhaps most inescapable in the context of electronic and computer music. Indeed, in this context, a musician must often play the role not just of composer, but of performer and instrument designer. The use and design of interfaces comes into play at each of these stages of creation.

A wide variety of methods for control and feedback of electronic instruments have

[5] Although one could make an argument for it: there is an audible percussive sound produced by the finger hitting the key, and the speed of release has an audible effect as well. Add to this the nuance of different levels of pedaling, and the possibilities become significantly richer.

been explored, ranging from auditory to graphical to haptic to textual [9], and one of the central design questions inevitably becomes how to establish musically effective mappings between controls and feedback. While acoustic interfaces naturally feature multiple, complex input and feedback relationships, these relationships must usually be more explicitly constructed in designing computer music interfaces. As Sergi Jordà points out, this sometimes leads to simplistic interfaces that lack the subtlety of their acoustic counterparts [10]:

> Blowing harder on many wind instruments not only affects dynamics but also influences pitch, and in these control difficulties lie much of their expressiveness... For the sake of simplicity, many electronic instruments use basic one-to-one linear mappings.

In addition to the issue of instruments themselves lacking subtlety, the development of performance mastery is another concern, given the constant technical innovation and evolution inherent to the discipline. As Mari Kimura writes [11]:

> The degree of excitement caused by new inventions that can "do anything" functionally surprises me, especially in comparison to the lesser interest in mastering these new devices.

There is nothing inevitable about these issues; indeed, Kimura herself has devoted her career to exactly the kind of mastery whose absence she laments. Likewise, there are plenty of musicians designing electronic instruments with subtle, non-linear mappings. The important thing is to be aware that some of what we take for granted in the analog domain must be explicitly accounted for in high-tech music-making.

2.2.2 Beyond the Instrument

While some computer music interfaces fall squarely within the category of musical instrument, others occupy more ambiguous categories. For example, in some cases the user acts as a conductor, adding various forms of live expression (e.g. dynamic and tempo fluctuations) to pre-composed material. In other cases—which we might call "performance environments"—algorithmically-generated material is created in response to live gestural or acoustic input [9]. These interfaces grow closer to what one might call a 'composition'.

However, through what iterative creative processes are compositions themselves brought into being? Such a process is, itself, a kind of interface, featuring various modes of input and feedback and acting as a translator between imagination and concrete musical representation. As early as 1985, Bruce Pennycook commented on the diversity of such interfaces and their importance in the creative process [12]:

> For some, paper and pencil suffices; for others a piano serves the purpose. Many contemporary composers work with multitrack sound recording systems and a complex of sophisticated electronic devices... Identifying composition tasks in terms of goals seems fruitless. We can, however, identify the kinds of tools that composers find useful: musical instruments, music manuscript, graphics, sound recording systems, programming languages, device controllers, etc. Thus the criteria for evaluating the effectiveness of the user interface must be based on a measure of its capacity to adapt to the needs of the composer throughout the compositional process.

Jonathan Savage comments on the power of such interfaces to shape the nature of the resulting music [13]:

> All musical technologies, and one can include musical instruments in this category, exercise a form of control over the composer. This can be positive and enabling or negative and limiting. In practice it means that the rigidity of the technology's functions can limit forms of expression. A simple example would be the use of a computer-based sequencing environment such as Cubase VST... Compared to a tape recorder, it is very easy to move through the 'blocks' of a composition within this environment and adopt metaphors such as 'cut, copy and paste' to extend and develop musical material.

Given the central role that these 'compositional interfaces' play in the creation of a new musical work, it behooves us to study their inner workings, charting their anatomy and investigating which properties best serve which musical goals. The following chapter represents just such an investigation.

Chapter 3

A Theory of the Compositional Interface

The following represents my own theory of the role of interfaces in music composition, a theory that—while informed by the context of the previous chapter—derives to large extent from reflection on my own compositional practice (specific examples of which will be discussed in the following chapter). However, my thinking on this matter has also been shaped by countless (and ongoing) conversations with teachers, colleagues, and friends. As such, I believe it has relevance well beyond the context of my own work.

The model presented here is almost certainly reductive at times. It almost certainly overgeneralizes and overlooks crucial aspects of the creative process. Nevertheless, as a lens through which to view the process of composition, I believe it offers something of significant value.

3.1 Overview

The process of composition can be thought of as a feedback loop, in which a human being iteratively modifies and refines some store of musical information, which we call the 'composition'. Over the course of this process—or many different such processes—the composition gradually takes shape.

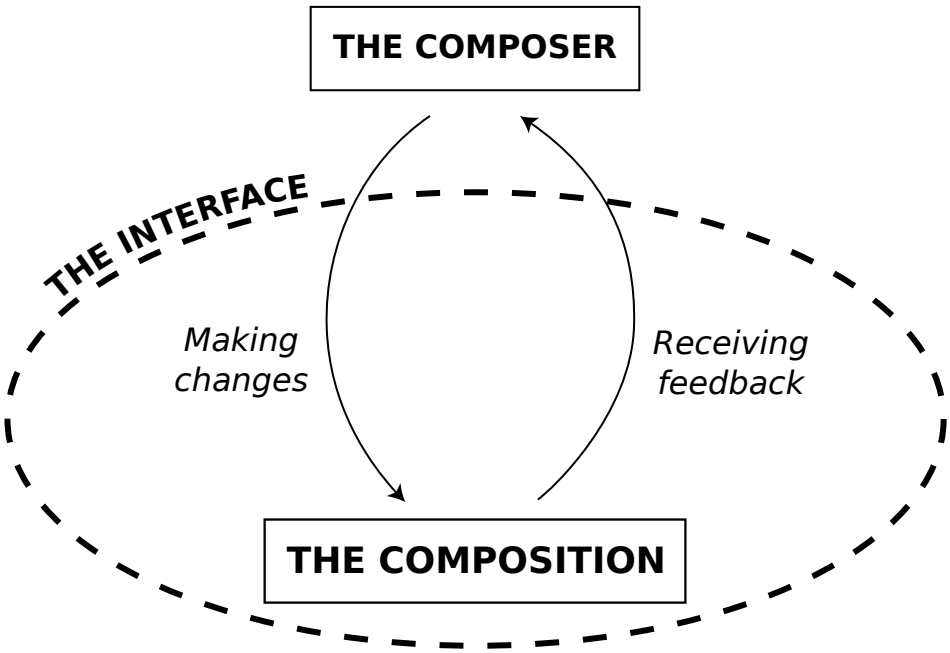

Figure 3.1: Diagram of the process of composition as an interface between the composer and the composition.

Figure 3.1 illustrates this feedback loop. What I will henceforth refer to as the 'compositional interface' encompasses the form in which the composition is stored or represented, the various ways of altering that representation, and the various ways of receiving feedback about the work in progress. Note that the key difference between a compositional interface and an instrumental interface is the presence or absence of a store of musical information. The feedback loop of instrumental performance involves only the manipulation of the instrument's controls and the resulting auditory, visual, and tactile feedback[1].

[1]Of course, some pieces are composed directly through improvisation on an instrument and never written down; in this case, one might think of the 'composition' as existing in the memory of the improviser.

3.1.1 Examples of Compositional Interfaces

Compositional interfaces can take all sorts of forms, ranging from low- to high-tech, and everything in between. Some examples might be:

- A pencil and music paper, while sitting at a piano
- Colored pencils on plain white paper, without any (external) auditory feedback
- A tablet notation program in which handwriting with a stylus is converted into a fixed vocabulary of musical symbols. (Perhaps the program also offers a form of synthesized playback.)
- One of the many available Digital Audio Workstations (DAWs), complete with a suite of plug-ins, virtual instruments, etc.
- Patching at a modular synthesizer[2]
- Coding a generative musical process with direct sonic output (e.g. in SuperCollider)
- Coding a generative musical process that outputs western music notation

It is important to note that, as far as the definition represented by Figure 3.1 is concerned, even a slight alteration of the tools involved constitutes a different interface. This is because seemingly subtle changes may yet affect the means of modification, the nature of the feedback, and/or the way the musical information is represented. This, in turn, may affect the mental model by which the composer approaches the interface (see Sec. 2.1.3).

[2]This could be considered an instrumental interface as well, depending on the context. In live performance, the synthesizer acts as instrument; in the studio—perhaps as a process of preparation for a live set—it is more of a compositional interface, since the configuration of patch cords is a store of musical information that is being iteratively developed.

For example, writing with a pen rather than a pencil may seem like a minor variation; however, the permanence of the medium might lead to a more deliberate approach, in which careful consideration is taken before committing an idea to the page[3]. Or, alternatively, perhaps the inability to erase would lead to a carefree attitude, in which mistakes are simply accepted instead of corrected, and in which passages are restarted from the beginning when too many mistakes have accumulated. While the exact result would surely depend on the composer, it is likely that even this slight modification would have an effect on the feedback loop.

It is also important to note that even within what appears to be a single interface, there are often multiple interfaces lurking. For instance, a composer may enter notes and chords into notation software via the computer keyboard, via clicking with the mouse, or via a piano keyboard. Since these represent different modes of making changes to the composition, they are really different interfaces. However, as they all are contained within the overarching structure of a single program, it might be natural to term them 'sub-interfaces'.

The lines here are admittedly blurry: as with any question of taxonomy, there is a tension between 'lumping' and 'splitting' [14]. For instance, one could consider the use of two different digital keyboards—say, one that uses springs and one that has weighted keys—to be separate sub-interfaces, since they feature different tactile feedback. Or, on the other hand, we could interpret the use of a particular piece of notation software as a sub-interface of the broader interface of "composing at the computer." Although there is an unavoidable subjectivity to this kind of categorization, I will generally use the term sub-interface to describe a mode of working within an overarching interface, especially in the case where one can fluidly switch between several different modes.

[3]A good friend of mine actually has the practice of tearing the corner of the page before he sits down to write, as a way of dispelling the notion that he is creating a perfect artifact.

3.2 Elements of the Interface

With the general framework now in mind, let us now consider the individual components of a compositional interface in more detail.

3.2.1 The Composition

I defined the term 'composition' above simply as 'a store of musical information'. This is a deliberately vague definition, and I mean for it to include any form of intentionally constructed musical object. For example, George Lewis describes his work *Voyager* "not only as an environment, but as a 'program,' a 'system' and a 'composition,' in the musical sense of that term" [15]. It is this broad meaning of the term 'composition' as the product of an iterative process of design and improvement that I am intending.

For instance, the composition might take the form of handwritten notation on a sheet of music paper, as well as the half-formed ideas, shapes, and words jotted hastily in the margins. In a Digital Audio Workstation, it might consist of the positioning of various clips, the selection of plug-ins and automation curves applied to those clips, and even the color-coding and organization of different tracks into musically significant groups. On a modular synthesizer, it might be represented by a particular configuration of patching cables and knob settings[4].

Of course, in reality there is rarely (if ever) a single, simply defined location in which the musical information of a piece of music is stored. Even in a finished, traditionally notated work, performance practice is arguably part of the work's identity. In pieces that incorporate live electronics, the code and associated documentation become an integral part of the work. In other cases, such as in much of the work of Meredith Monk, the work

[4]Again, this raises the question of where the difference between composition and performance lies. In this case, if the modular synth patch is performed live, perhaps it is only those connections and knob settings that are *invariant* over the course of the performance that represent the composition.

is conveyed in large part via oral transmission. Finally, and perhaps most obviously, works that combine different media—such as opera—necessarily combine different varieties of information storage.

When a work is still in progress, the distributed nature of the musical information is particularly salient. Pre-compositional diagrams, snatches of music notation, sketches of chord progressions, bespoke instruments, specific wirings of effect pedals, snippets of computer code, collections of audio clips—all of these may be loosely understood to be part of the same work. This points to perhaps the most important of the locations in which a composition is stored: in the memory structures of the composer's brain. It is there that the myriad partial representations come together to form a unified concept.

In considering a compositional interface, it is generally one particular representation of the work that is being iterated over, with the understanding that this representation is part of a larger, unified conception. Over the course of work on a piece, the composer will likely switch between several different interfaces, each featuring its own partial representation of the work in progress. For instance, Beethoven famously did much of his initial sketching on a single line staff, even when working on large orchestral works like the Eroica Symphony [16]. As I will discuss below, these partial representations often allow the composer to focus on a particular aspect of the work in progress.

3.2.2 Input Paths

Different interfaces offer different options for modifying the work in progress, and one important distinction is between those input paths that offer discrete control and those that offer the possibility of continuous shaping. As discussed in Section 2.1.4, Verplank refers to these as 'buttons' and 'handles' respectively.

For instance, one of the key features of composing with paper and pencil is that it

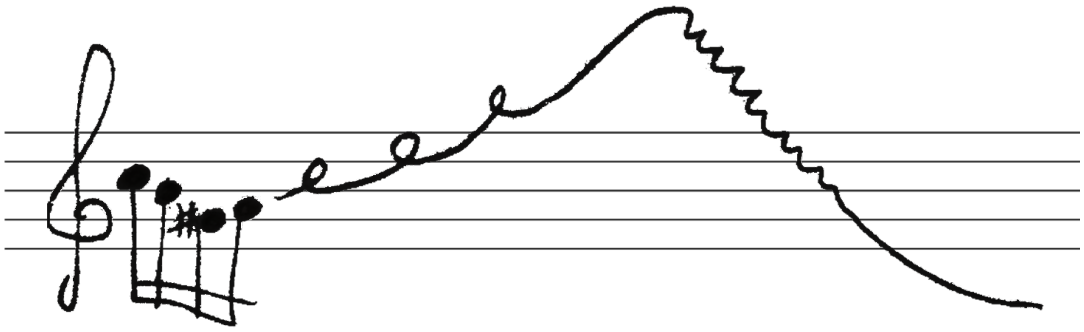

Figure 3.2: Sketching the contour of a gesture by hand on music paper.

presents the composer with continuous control at all levels. Even when working with traditional western notation, with its discrete treatment of pitch and rhythm, continuous ideas can be easily conveyed via impromptu graphic notations. For instance, when I wish to indicate the contour of a gesture, but am unsure of the precise notes and rhythms I want to use, I will sometimes draw something along the lines of Figure 3.2, with the understanding that the precise (discrete) details will be filled in later. This is a continuous mode of input, and one which I know many composers make use of when writing by hand.

By contrast, entering music into score-writing software—such as Sibelius, Finale or MuseScore—is a thoroughly discrete affair (all buttons, in Verplank's terminology). Every note must have a clearly defined pitch and duration. Graphical symbols are selected from a fixed set of options. This is not to say that score-writing software does not offer certain advantages; after all, the legibility of the resulting notation is in part due to the consistency that results from using discrete representations. However, particularly when something of a continuous nature is to be communicated, such programs can be deeply frustrating to the creative process (as any composer that has tried to input proportional rhythmic notation will tell you).

Digital Audio Workstations generally offer a mix of discrete and continuous control. For instance, a virtual instrument plug-in might receive discrete-pitch MIDI data, but in

addition offer a continuous pitch bend knob. Likewise, whether or not a plug-in is active on a track is a discrete setting, but the plug-in itself frequently features settings that take a continuous range of values. In general, automation curves represent continuous control, and a popular method of composing such curves is through the use of connected hardware knobs and faders.

One of the most notorious ways in which continuous control is ceded to discrete control in DAWs is through quantization (or "snapping") to a metric grid. Indeed, in many cases such behavior is the default setting, which leads to important questions about the role such defaults in creative applications.

Words, Words, Words...

As alluded to in Section 2.1.4, in addition to the discrete input of buttons and the continuous input of handles, verbal description seems to fall into an entirely separate category. Words offer the power of connotation, of association, and cut across many musical parameters at once.

For example, a sketch for the sixth movement of Ligeti's *Ten Pieces for Wind Quartet* consists of the following, type-written text [17]:

> "Oboe Concerto" short, medium-fast. Note repetitions—staccato—clockwork, precision mechanism, polyrhythm-polymeter, internally otherwise elegant legato-staccato, alternation, mannerist oboe solos also, here and there aggressive deep oboe register (with bassoon in unison).

While some words (e.g. "staccato") have precise musical meanings, others are richly ambiguous (e.g. "precision mechanism"). The eighth movement of the same work contains associations of a different kind:

> "Horn Concerto," initially slow solo horn introduction with echoes (= Mahler 7^{th} Symphony, 1^{st} Nachtstück), soon ex abrupto: fast (short) brutal, really a caccia piece, without the horn caccia in some places, caccia-like oboe (Pastorale scherzo) but only with short entrances.

This web of association contains not only musical terms and suggestions of character, but a reference to a particular model, from which Ligeti borrows, among other things, the idea of stopped echos.

Defaults, Dark Patterns and Discoverability

How many pieces use quantized timing simply because that was the default in the DAW used to create them? For that matter, how many songs are in 4/4 time simply because that that was the default time signature?

Some evidence of the effect of interface design decisions on user behavior can be found in recent attention to the use of 'dark patterns' in the design of games and other popular user interfaces. Generally, the term 'dark pattern' refers to an instance in which the game or interface designer intentionally leads the user towards or away from certain actions, in a way that is detrimental to the user but beneficial to the designer [18]. For instance, some actions may be discouraged through needless difficulty ('obstruction'), or relevant information might be obscured or hidden ('sneaking') [19]. A typical example might be a paid subscription service that places the "unsubscribe" page in an unintuitive location, accessible only through a maze of links. If such design strategies did not affect user behavior, they would hardly have become so widespread.

While the intentional undermining of user intent is less likely to be an issue in the case of compositional interfaces, it is nevertheless the case that some patterns are 'brighter' than others. This brightness is typically referred to as *discoverability* and is a key feature

of interface design [20]. Consider, for instance, the difference between a large, permanent button at the top of the screen with the words "REVERSE SOUND CLIP" and a nested menu item found under "Edit > Clip Options > Temporal Transformations"; the former is far more discoverable than the latter. Since there is limited screen space, some input options will inevitably be more discoverable than others.

If we know ahead of time what we wish to accomplish, we will be motivated to discover the means to do so, however hidden it may be. However, much creative work relies on open-ended exploration, in which the goal is not known in advance. In my experience, such work is highly influenced by the relative discoverability of different options[5].

3.2.3 Feedback Paths

The final link in the chain of a compositional interface is the feedback by which the composer assesses the work in process. We can broadly categorize such feedback into *auditory*, *visual*, and *tactile* channels, as illustrated by Figure 3.3.

Auditory Feedback

Auditory feedback, unsurprisingly, plays a prominent role in most compositional interfaces. Many different forms of auditory feedback exist, each of which conveys different information to the composer.

Many composers have famously written at the piano (e.g. Stravinsky [21]), in large part because of its ability to render a full musical texture. In particular, due to its homogeneity of tone-color, the piano offers a powerful tool for auditioning harmony. On the other hand, when composing for diverse instruments, this homogeneity misrepresents

[5]I believe this is an important topic for future research into creative interface design. For instance, a study could be conducted in which users are asked to accomplish an open-ended creative task in a simple interface, with some actions available as permanently-visible buttons (highly discoverable) and others hidden inside of menus (less discoverable). For each participant, the locations of the actions could be randomized, and the resulting patterns of usage could be analyzed.

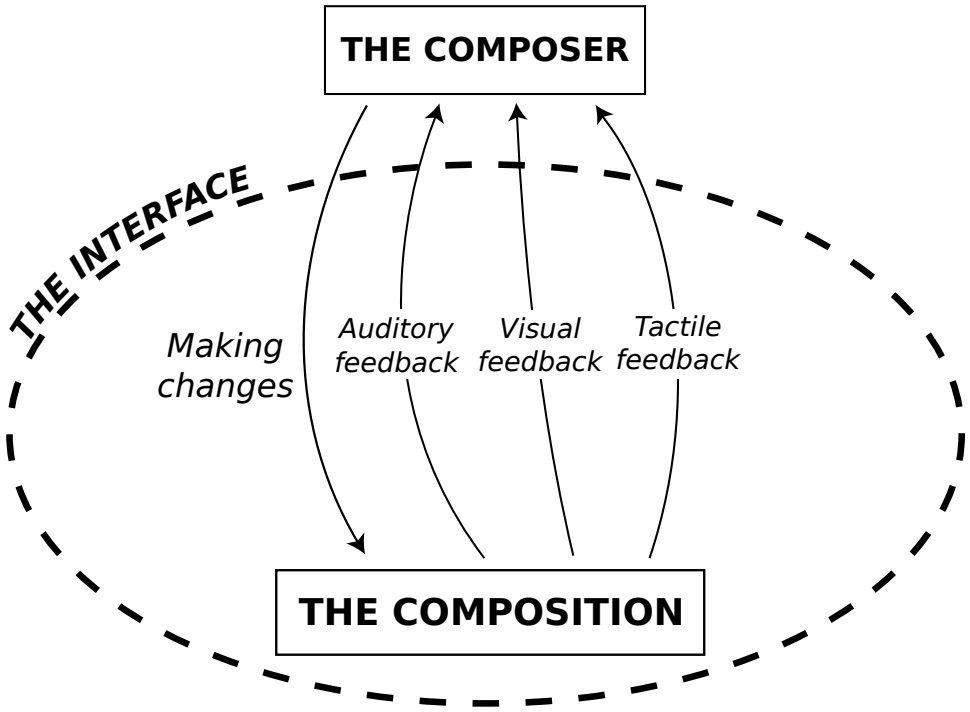

Figure 3.3: Diagram of the compositional interface, with expanded detail on the feedback paths.

the variety in timbre. Similarly, when writing for sustained instruments, the percussive envelope of the piano and lack of expressive control after the attack are misleading.

Even for a very skilled pianist, there are limits to the contrapuntal complexity that can be accurately rendered, especially in real time. By contrast, the playback offered by score-writing software such as MuseScore, Sibelius, and Finale can handle almost limitless complexity, and at any tempo. As such, it can be particular valuable in accurately judging duration and pacing[6]. Depending on the quality of the samples and the playback engine, it can often also give some sense of the timbre combinations within the ensemble.

Where computer playback falters is in its notoriously poor expressive timing and use of dynamics. It also gives highly misleading information about relative balance. Overall,

[6]I have also found such playback to offer different information at different speeds: at very slow tempos, one can pick up inconsistencies in harmony and voice-leading; at very fast tempo (e.g. 4x, 8x, 16x) speed, the notes become a blur, and the gradual changes in contour and register become most salient.

there is a sense of 'flatness' to the sound, since—unlike the piano—the playback does not arise in the context of a live instrumental feedback loop[7].

From the point of view of accurate auditory feedback, then, it would seem that a Digital Audio Workstation offers perfection. What we hear (at least in an accurate listening environment) is the music itself, rather than a version filtered through a particular set of playback limitations. However, there is a flip-side to those limitations: they force the composer to engage their imagination. When writing at the piano, I mentally project tone-color onto the instruments involved; when listening to computer playback in MuseScore, I make an effort to hear beyond its mechanical drabness, grafting imagined nuance onto the performance.

Of course, imagination plays the most central role in interfaces with no external form of auditory feedback, such as pencil an paper. Here, it is the internal ear that provides all of the auditory feedback, and while these sonic images are often fainter and more fickle than their physical counterparts, they are also extremely open-ended. The main limitation here is that, as these mental images are based on recollection, one cannot conjure up a mental image of a sound that one has never heard before[8].

The truth is that no single form of auditory feedback is superior in all circumstances; different kinds of feedback simply offer different kinds of information. There are times when the harmonic feedback and musical timing of the piano are exactly what is needed, and other times (for instance, when working on a work for solo oboe) when its shortcomings render it useless. There are times when the concreteness of playback from a DAW feels over-determined, leaving no room for the imagination, and other times when, working with pencil and paper, the imagination is too *indefinite*, too far from the actual

[7]It is this flatness, I believe, that results in many pieces composed at the computer being given impossibly, and unmusically, fast tempo markings.

[8]Or can one?

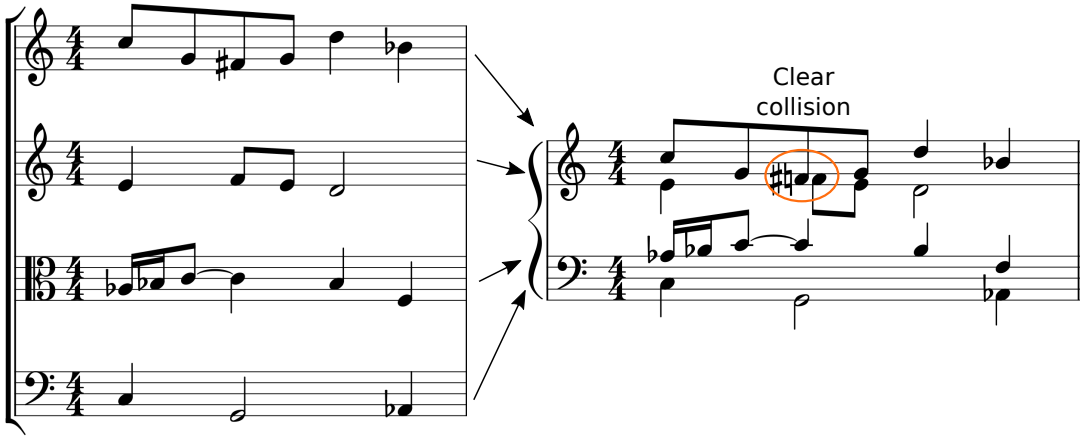

Figure 3.4: The same measure of quartet music, represented with two different kinds of visual feedback. On a grand staff, collisions are easy to spot.

experience of hearing the music[9].

Visual Feedback

As with auditory feedback, different forms of visual feedback offer different information about the work in progress.

For instance, consider the difference between writing a string quartet on four separate staves versus on a single grand staff. When laid out on four separate staves, the melodic motion of each individual part is highlighted, while the resulting harmony is somewhat hidden from view (although score-reading practice helps one to see the harmony in this configuration). By contrast, when writing on a grand staff, the vertical dimension provides a clear, one-to-one mapping with pitch space, allowing harmony, and harmonic spacing, to be quickly parsed. As a teacher, I have noticed that inexperienced composers will frequently create unintended crossings and collisions of voices when working with separate staves, mistakes that they would have quickly caught if the music were represented on a grand staff (see Figure 3.4).

[9]For instance, I am not convinced you can truly experience bass frequencies in your imagination.

Interfaces are often forced to visually partition time in musically arbitrary ways, for instance in the form of system or page breaks. One composer I spoke with noted that his students' phrases are often linked to the size of paper they use [22]. Along a similar vein, inexperienced composers and performers sometimes produce "blocky" music that is overly influenced by the presence of barlines. An interesting response to some of these tendencies is the "Continuous View" in MuseScore (called "Panorama View" in Sibelius and "Scroll View" in Finale) that presents the music as a single, unbroken system.

One valuable property of visual feedback, as opposed to auditory feedback, is that it can allow the composer to perceive a stretch of time all at once. It is possible, for instance, to sketch an hour-long symphony on a single sheet of paper, mapping out the proportions of the different sections. By allowing us to step outside the usual flow of time, visual feedback provides a valuable counterpoint to real-time auditory feedback.

One key outside-of-time function of visual feedback is the organization and categorization of material. One composer I spoke with described spending many hours color-coding his clips in Ableton prior to improvising with them [23]:

> I code them in colors, based, for example, on energy... and then I try to see how certain families of sound of certain kinds of energy speak to each other, and if I want contrast, then there's a name for it, but there's also a color that gives me kind of a sense, and I can in a fast way assemble them and see how they speak to each other.

Tactile Feedback

Tactile feedback is of vital importance to instrumental interfaces, so it is unsurprising that a key role of tactile feedback in compositional interfaces is to connect one to the physical act of playing an instrument. For instance, when composing a fortissimo

orchestral tutti at the piano, the physical act of exertion required to play it mirrors the exertion that will be required of the members of the orchestra.

Tactile feedback can be misleading in this regard, however. Particular care must be taken when the tactile feedback is coming from an instrument different from the one being written for, since the target instrument may have a physical limitation that the feedback instrument does not. For instance, while it may be easy enough to jump down a diminished 12^{th} on the piano, a vocalist will not find the motion so facile. On the other hand, sometimes the instrument providing tactile feedback imposes an artificial limit, such as when writing for orchestra at the piano, where it can be easy to overlook voicings that do not fall comfortably into the hands.

In the realm of electronic music, evidence of the importance of tactile feedback can be found in the recent resurgence of interest in modular synthesizers and other hardware synthesis tools. In fact, even though some companies, such as Mutable Instruments, offer identical software versions of their modules for free [24], people still buy the physical hardware. There is simply no true substitute for the physical experience of patching cables and feeling the knobs at your fingertips.

Information Extraction

An easily overlooked aspect of the compositional feedback loop is the point of contact between the various forms feedback and the composer: the systems of information extraction within the brain. This is illustrated in figure 3.5.

The harmonic feedback of the piano is only helpful insofar as the composer is able to take it in and make sense of it. Similarly, in the process of mixing, only the composer who can recognize that an excess of energy around 200Hz is causing muddiness is able to correct it. Every piece of auditory feedback must first pass through the auditory cortex.

This points to the vital role of ear training for composers. It doesn't matter what

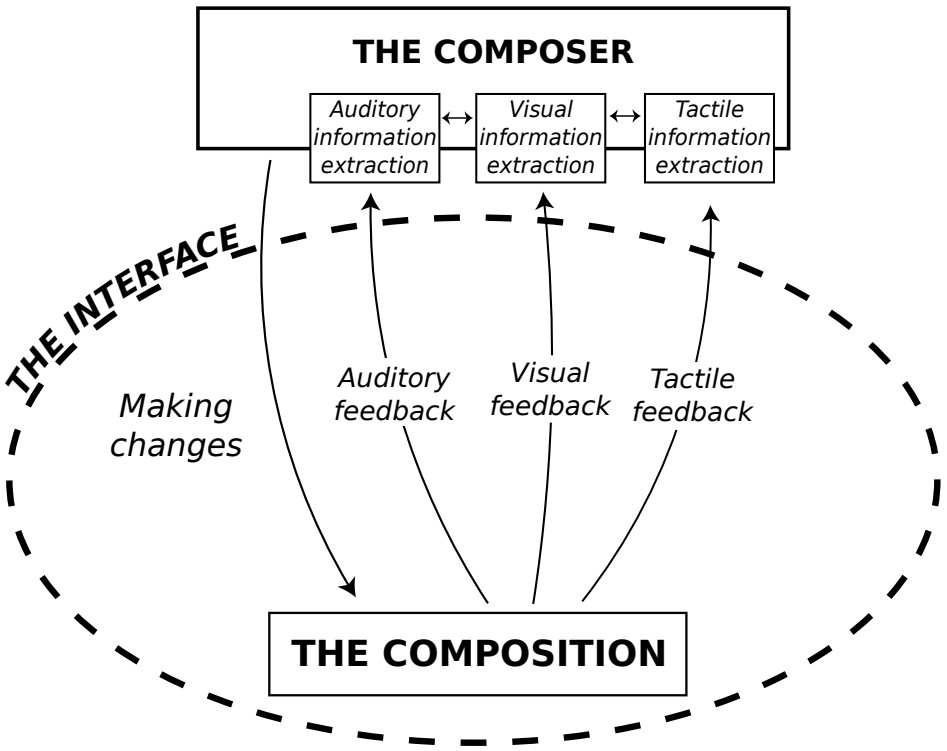

Figure 3.5: Diagram of the compositional interface, incorporating the process of extracting information for the various forms of feedback.

information is available to us if we cannot make sense of it. Moreover, ear training is often feedback- and interface-specific: an ear for muddiness and an ear for harmony are only helpful in their respective contexts. Too often, ear training is treated as a fixed set of generally applicable skills, when in fact it is an endless assortment of context-specific skills.

Beyond ear training, this highlights the importance of training certain forms of visuospatial information extraction as well. Consider again the example (Figure 3.4) of a string quartet being composed on four separate staves, in which there is an inadvertent, dissonant collision between two of the voices. A trained score reader, with experience making the spatial translation from horizontal voices to vertical pitch space, can quickly spot this error at sight, even when viewing the music on separate staves.

Sometimes the same information can be arrived at through different modalities. In the quartet example, a musician with a well-trained ear for interval and harmony might instead notice the voice collision upon hearing the computer playback. On the other hand, a musician with neither the appropriate visual nor the appropriate auditory training might well miss the error entirely, simply wondering why something sounds vaguely amiss.

While a carefully-chosen interface cannot give you a better ear, or better visual processing, it can help direct your attention towards certain kinds of information, lightening the cognitive load resulting from extraneous information. For instance, I was once instructed by a teacher to copy down just the bassline of a work I had written. Without the distraction of the upper voices, it suddenly became clear to me where the music lacked harmonic freshness, and where a change of key was in order.

Finally, although examples come less readily to mind, there is no question that tactile information extraction plays an important role as well, particularly when composing at an instrument. For instance, when recording musical gestures using a custom, joystick-controlled electronic instrument, a composer might come to learn, through feel, how to produce certain sounds.

The lesson here is clear: as a composer comes to settle on the use of particular interfaces, it behooves them to develop their mental capacity to extract information from those forms of auditory, visual and tactile feedback upon which these interfaces rely.

3.3 Interface Design Considerations

Composers often find themselves selecting among, and even constructing their own, interfaces. Indeed, in my own case—as the examples in the following chapter will show—a significant portion of the time spent working on a piece goes into the creation and adjustment of interfaces.

In choosing and designing interfaces, what properties are important to consider? I believe that some of the key considerations are *latency*, *scope*, and *openness/predictability*.

3.3.1 Latency

In a feedback loop of any kind, the term *latency* refers to the accumulated time delay in traversing a full circuit. It is an important property that has been investigated in reference to many interactive systems, including musical ones [25].

In a compositional interface, I use the term latency to refer to the time it takes to make a change to the composition, and then to receive and interpret feedback about the effect of that change. As such, there are two main forms of latency: input and output.

Input latency

Consider the difference between selecting an item from a menu and using a keyboard shortcut: the former likely takes several seconds, while the latter is almost instantaneous. Or consider the difference between transposing 10 measures of highly chromatic music automatically in a piece of notation software, and doing the same by hand on a sheet of music paper. What is done in seconds in software might take hours to do manually.

As these examples show, different actions on the composition can vary wildly in their latency. Moreover, the same nominal action can be orders of magnitude faster or slower in one interface than in another.

Lest the above example about transposition mislead the reader into thinking the computer always offers lower latency, consider the experience of notating a glissando that first overshoots before ultimately arriving at its intended destination (see Figure 3.6). By hand, this is trivial to notate—a curved line is no harder to draw than a straight line. However, on the computer, as no symbol exists for such a glissando, one

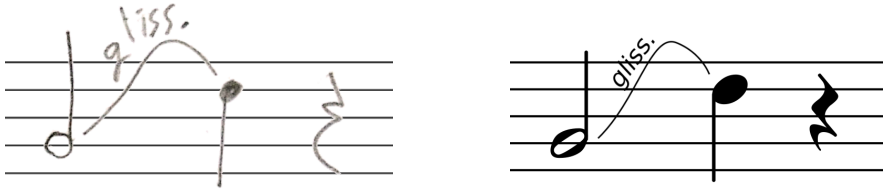

Figure 3.6: A custom glissando shape rendered by hand in 2 seconds (left) and via a custom graphic painstakingly imported into MuseScore in 20 minutes (right). Revealingly, the process of creating the latter made me question whether it was even worth including this figure in the first place.

must be designed in a separate piece of software, and then imported. Such latency could well disrupt the flow of the creative process.

When comparing across different users, rather than different interfaces, an important determinant of input latency is the user's degree of familiarity or expertise at carrying out a particular action. What takes a novice user 30 seconds might take an expert only a couple of seconds.

Feedback Latency

A typical situation in which feedback latency might arise is when a process takes some time to render. In the early days of computer music, feedback latency often represented a large portion of the time spent working on a project. Sharing time on large mainframes, it might take days to find out whether a process was producing interesting results. Nowadays, processes that used to take hours or days to render can often be accomplished in real-time.

However, feedback latency can still crop up in other forms. For instance, my first attempts at generative music programs could only output MusicXML files. This meant that, in order to listen to what I was creating, I had to generate the MusicXML file, leave the program, find the generated file, open it up in Sibelius, and press play. This process

of translation represented a significant feedback latency.

In the same way that different input paths can have differing latencies, different forms of feedback can as well. For instance, visual feedback is often fairly instantaneous, while auditory feedback unavoidably takes time. Moreover, listening to longer stretches results in higher latency. When I'm working on a passage of music, I will often alternate between listening to just the part I'm working on, and starting further back in order to hear the passage in context. The context is always valuable, but it comes at the cost of a larger latency.

The Impact of Latency

The most obvious impact if latency is that, since it takes time, it slows down the compositional process, allowing fewer passes through the feedback loop. This is generally a bad thing, since the purpose of the feedback loop is to improve the composition.

Of course, this assumes that the time spent waiting is unproductive. This was certainly the case for me opening up and playing MusicXML files; it was a nuisance that offered no insight into the work at hand. However, if longer periods of rendering offer time to reflect, this might be a potential benefit. I sometimes wonder if, for early punch-card-era algorithmic composers, there was value in having so much time to think.

Latency is especially problematic when we have limited understanding of what we are doing, or when the interface is unpredictable. As Verplank writes: "The longer the delay between doing and feeling, the more dependent I am on having good knowledge" [5]. Especially when the act of composing is akin to exploring an unknown territory, latency hinders our ability to map out and understand the possibility space.

While feedback latency slows down the process of composition, a more insidious concern arises with input latency, since there are often many actions to choose from, some of which have lower latency than others. In my experience, the lower latency

options are much more tempting, as they more naturally lead to a sense of flow. High latency options—while ostensibly available—will tend in practice to be neglected, in the same manner as the dark patterns described in Section 3.2.2.

One of the places in which this plays out is in the many varieties of Digital Audio Workstation. Although these programs all have roughly the same basic functionality, what is a simple keyboard shortcut in one may be a more complex, time-consuming action in another. When combined with the influence of differences in visual feedback, it becomes clear that the selection of a DAW is a musically significant decision.

This was first driven home to me when working on an fixed-media piece in the program REAPER. Time-stretching a clip in REAPER is as simple as holding down the option key and dragging one end of it. After working on the piece for a while, I took a step back and realized I had used this feature heavily, so much so that it had become an integral part of the fabric of the piece. If time-stretching had instead required me to select an option from a menu, type a new duration into a dialog box, and then press return, I am certain that it would not have played such a large role.

There is not necessarily anything wrong with this—in fact, I'm quite fond of the piece that resulted. However, I am concerned about the degree to which such differences in input latency *unconsciously* shape the music we create. One solution would be to build more configurability into our interfaces, allowing composers to decide—even on a piece-by-piece basis—which paths are to be lower- and higher-latency. This hearkens back the assertion of Runciman and Thimbleby, as early as 1986, that "premature design commitments must be avoided" in user interfaces [26], a sentiment later echoed by Stowell and McLean in their suggestion that interfaces allow people to apply their own conceptual metaphors [27]. For me, this configurability is partly the appeal of programming, as I will discuss in Section 3.4.1.

3.3.2 Scope, Breadth, and Compression

Another important property of an interface is its *scope*: what view of the composition does it represent, and what does it give one the power to alter?

Note that what an interface allows you to change and what it gives you feedback about are not always the same. For instance, many pieces of score-writing software allow the composer to sync playback with an external audio or video file. In a work for acoustic instruments and fixed media, this can give the composer useful auditory feedback about the combined musical texture. However, only the acoustic part of that musical texture is alterable from within the interface. When the acoustic and electronic parts are in dialogue, this can be quite awkward, as I myself have experienced on multiple occasions.

A useful heuristic regarding scope is to consider whether the interface represents a broad or narrow view of the composition. This general property of *breadth* is related to—but distinct from—another property: the degree of *compression* of musical data. For instance, a one-page sketch of a 60-minute orchestral work is both broad and highly compressed. On the other hand, a full, printed score of the same work, while still broad, is totally uncompressed. An example of an interface with a very narrow scope, on the other hand, might be the manual piecing together of grains of sound in a DAW as part of a process of micro-montage.

While questions of breadth and compression often have to do with visual feedback, they are also very much at play in many forms of auditory feedback. For instance, sometimes I find myself narrowing in on a single beat within a measure while at other times, I will play or listen to a draft of an entire work to get a sense of its pacing. These are two different levels of breadth. While temporal *compression* might be a little less common with auditory feedback (since there is generally a single correct tempo), I do sometimes find myself playing back the same music at half-tempo or double-tempo to

gain different insights on it. Finally, auditory feedback can also be broad or narrow in the degree to which it considers the whole or only part of the musical texture.

At different stages in the compositional process, narrower or broader interfaces may be called for. This seems to vary from composer to composer, and from piece to piece: sometimes it may be desirable to start with a broad view and then proceed to fill in the details; other times it may be desirable to start with a narrow view and expand outward from it, growing it like a seed.

3.3.3 Openness and Predictability

Stowell and McLean define the *openness* of an interface as its ability to produce an outcome that it was not explicitly designed to produce [27]. In addition, we might consider the related property of *predictability*: the extent to which we can predict the change in feedback that will result from a given manipulation.

Both of these properties suggest the ability of the interface to produce something novel and unexpected. In the case of openness, this novelty arises from the interface's ability to move beyond the outcomes that were foreseen when it was designed. Unpredictability, by contrast, can be purposefully designed into the interface, for instance via the incorporation of randomization.

A good example of an interface that is open by design is the modular synthesizer. Even when restricted to the simplest of modules—a few oscillators, filters, and envelopes—the multiplicative possibilities of wiring make it easy to produce a sound that the instrument was not designed to produce. As anyone who has worked with modular synthesizers can tell you, they are also unpredictable. Once enough cables have been patched, it becomes almost impossible to predict the result of turning a knob.

Exactly when that threshold will be crossed will depend on the experience of the

player: an experienced player, working with familiar modules, in a familiar configuration, may be able to exert significant control over the sound. A novice will quickly lose control. This suggests a more general relationship between expertise and predictability. As a tool or process becomes familiar, it becomes predictable. Is this a good thing?

For many composers, managing the predictability of their interfaces is an important part of the creative process. Unpredictability tends to be particularly valuable in the early stages of working on a piece, as it allows novel sounds and ideas to arise. One composer I talked to generates unpredictability by working with unfamiliar collaborators [23]. I typically generate unpredictability by designing and exploring new pieces of software.

Later in the process, however, unpredictability can become a hindrance. To take an absurd example, most composers would not want unpredictability in the process of engraving (e.g. random flipping of accidentals, random spacing of notes). Once a work starts to come into clear focus, predictable, precision tools are in order.

Unsurprisingly, Cage offers a unique perspective on predictability. In *Child of Tree*, which involves improvisation on—among other things—an amplified cactus, he intentionally chooses instruments that are unfamiliar to the performer, so that "as you explore...you're not really dealing with your memory or your taste" [28]. Moreover, this unpredictability never gives way to expertise, since the spines loosen and fall off as they are played.

In acoustic composition, an important source of open exploration—as well as unpredictability—can be found in the process of working with performers. Indeed, one could consider the whole feedback loop of presenting an ensemble with, and getting feedback on, successive drafts of a composition to be a kind of open, unpredictable interface. Experienced and thoughtful performers have a delightful ability find new connections and identify musical problems, and history abounds with examples of fruitful collaborations between great composers and virtuoso performers (e.g. Brahms and Joachim).

3.4 Two Case Studies

We now apply some of these principles, as well as the model of the compositional interface in general, to the consideration of two specific interfaces: one high-tech, and one low-tech.

3.4.1 Programming and the Latency/Openness Trade-Off

Programming languages are the ultimate configurable interfaces, at least as far as composing with a computer is concerned. They are also extraordinary open: a low-level, general-purpose programming language can accomplish anything that the computer itself is capable of, at least in theory.

Programming languages also reveal something about the nested nature of interface creation. While they are themselves interfaces—you make changes to a store of information (the code) and receive feedback about it (by running it)—they can also be used to construct new interfaces. Not only that, but programming languages themselves are the product of programming with lower-level languages. Interfaces are being used to create interfaces, which are in turn used to create interfaces, on and on in a chain of greater and greater abstraction.

The problem with the openness that programming languages offer is that it comes at a cost: exceedingly high latency. Depending on the complexity of the task, it may be days or weeks before we can receive meaningful feedback about the functionality of our code. This latency is especially bad with lower-level programming languages (e.g. C); the higher-level the language, the more time-saving, latency-reducing abstractions it provides. Domain-specific languages (of which many exist for musical applications, e.g. SuperCollider, CSound) further reduce latency, by providing abstractions relevant to the particular domain in which one is working.

There is at pattern here: as we reduce latency, allowing ideas to be more quickly expressed in code, we also reduce openness. Higher-level languages lose direct access to hardware and memory management. Domain-specific programming languages operate on a narrower domain of possibilities than do general-purpose languages. Having chosen a language, the use of a particular framework narrows our focus still further.

In fact, the very act of programming itself is a process of reducing both openness and latency, hand in hand. When we write a subroutine or encapsulate functionality into a class, we make it possible to accomplish more with fewer lines of code (reduced latency). However, that code expresses a narrower range of possibilities (reduced openness).

For instance suppose we write a function (in this case in Python) to play a turn around a given note:

```
def play_turn(main_pitch, turn_width, turn_dur, total_dur):
   # assumes we've already defined the function 'play_note'
   # takes a main pitch, the width of the turn (e.g.
   # whole-step, half-step), the turn duration and the
   # total duration of the turn plus the final note
   final_note_dur = total_dur - turn_dur
   turn_note_dur = turn_dur / 4
   play_note(main_pitch, turn_note_dur)
   play_note(main_pitch + turn_width, turn_note_dur)
   play_note(main_pitch, turn_note_dur)
   play_note(main_pitch - turn_width, turn_note_dur)
   play_note(main_pitch, final_note_dur)
```

Calling this function, we gain the ability to accomplish in one line what would otherwise have taken several:

```
# play a half-step turn around midi pitch 60 (middle C)
# (turn lasts 0.2 beats out of a total length 1 beat)
play_turn(60, 1, 0.2, 1)
```

However, all that this line can do is play turns, with the potential to vary only the pitch, width, and lengths involved. Moreover, perhaps inadvertently, we have made it impossible for this function to execute a turn in which the distance to the note above and the distance to the note below differ (as is often the case in diatonic turns). Of course we haven't truly lost the openness that we had before writing this function; we just only have access to that openness in the higher-latency mode in which we do not call it.

When programming an algorithmic piece, my platonic goal is to reach the point where the entire piece (or at least some significant portion of material) is generated with a single line of code, along the lines of:

```
play_the_piece(param1, param2, param3, param4, ...)
```

In this way, the important expressive parameters are easily alterable, in a very low-latency fashion, and can be tweaked until a desired musical result is reached[10]. The process leading up to being able to write this single line of code is a circuitous one, defining subroutine upon subroutine and gradually abstracting away until the identity of the piece—and thus its key expressive parameters—come into focus. At this point, the interface is no longer open at all; it only generates variations of this singular work. However, is that not the goal composition in general: to proceed from generality to carefully considered specificity?

3.4.2 Pencil and Paper: Fluid Switching

Discussions I have had with other composers about their compositional processes have repeatedly reinforced the idea that pencil and paper is a very special kind of interface. In fact, many consider it to be the most important interface that they use [22].

[10]In fact, in an effort to reduce latency to a bare minimum, I will sometimes hook up physical sliders to the parameters so that I can manipulate the key parameters in real time.

What makes pencil and paper so valuable for expressing musical ideas? We have already discussed its expressivity, owing to the fact that it employs almost exclusively continuous control. One composer I spoke to pointed out the effect that this has on memory [23]:

> When I return to my drawing, I almost can see everything I was thinking in that slightly wiggled line there, while if it was represented by the standard wiggled line of some kind of software, I don't have that connection. It doesn't carry the same power of association in memory.

Thus, the continuous nature of the medium allows for a kind of lossless compression of musical information, one in which a complicated mental state can be recovered from a tiny written gesture. Handwriting mirrors thinking, and—like thinking—it has an openness and immediacy (low latency). For many, the fact that auditory feedback is internally generated is also freeing, since one's imagination receives no interference from the outside.

All of these aspects are important. However, in my mind, the truly special thing about pencil and paper is the fact that it is not one interface, but many. Traditional notation, shorthand, expressive curves, shapes, symbols, verbal expression: all of these diverse sub-interfaces coexist within one overarching interface.

Most crucially, it takes little to no effort to switch between these different modes of expression. Such fluidity is rare. Consider, for instance, the challenge of integrating symbolic and graphic notations presented earlier in Figure 3.6. On the computer, the process of interface-switching was a time-consuming one: leaving the score-writing application, opening a vector graphics application, designing and exporting an image of the curved glissando, importing it back into the score-writer, and then finally trying to align it correctly within the measure. With paper and pencil, by contrast, these two very

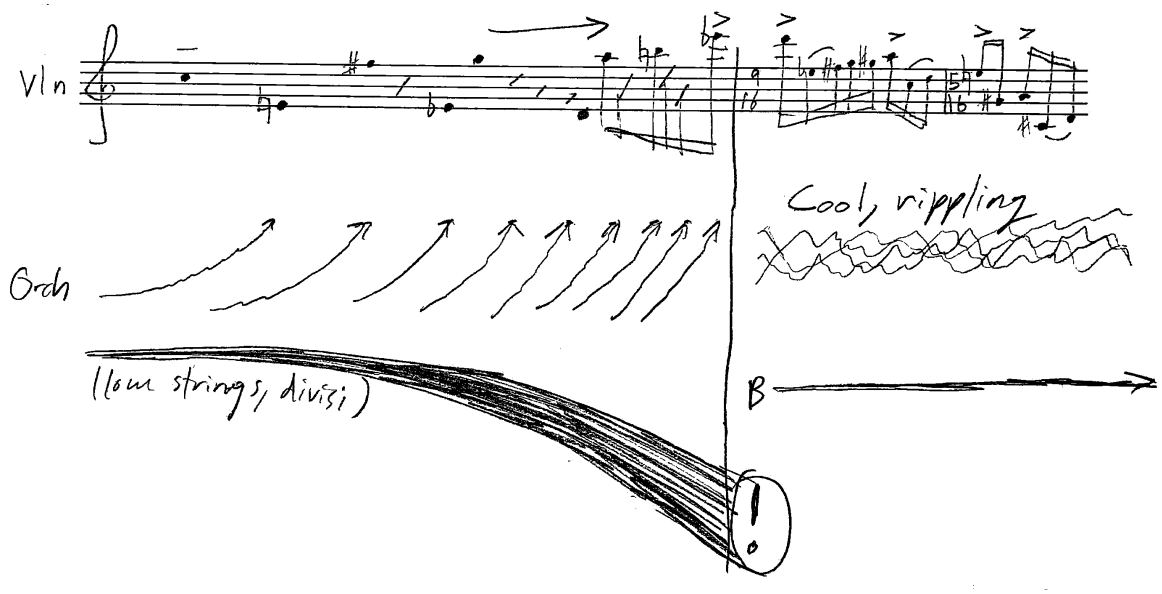

Figure 3.7: Sketch from an imaginary violin concerto, exemplifying the range of expressive tools (sub-interfaces) available with pencil and paper.

different forms of expression were effortlessly and seamlessly combined.

Figure 3.7, a sketch from an imaginary violin concerto, represents a more elaborate example of the same principle. The violin part combines metered and proportional rhythmic notation, as well as definite and indefinite pitch notation. The orchestra part combines gestural notation, notions of thickness and density, suggestive symbols (such as the exclamation point), and verbal descriptions.

One of the valuable things about words, in particular, is that they can simultaneously address multiple musical parameters[11]. For instance, "cool" here suggests some combination of harmony, timbre, and texture. "Rippling" likewise suggests texture, and interacts synergistically with the graphic notation beneath it[12].

Drawing and writing by hand are some of the earliest forms of human artistic expres-

[11]I still remember struggling once to teach a kid how to swing eighth notes, and then finding that "play it jazzy" accomplished it immediately, adding, in addition, some well-placed accents and other appropriate inflections.

[12]Luckily, words are often also available in much less flexible interfaces. For example, I often find myself placing verbal suggestions to myself within scores that I am manipulating on the computer.

sion. We can learn much about good compositional interface design by examining what makes this most basic of interfaces so special.

3.5 Concluding Thoughts

3.5.1 The Value of Interface Switching

We have seen that the ability to fluidly change modes with pencil and paper is incredibly valuable, as it offers up a rich variety of expressive possibilities in a compact manner. However, beyond simply expanding our possibilities, the process of switching itself has value, as is highlights the differences between modes of working.

In reference to the upending of television by the internet, Johnson writes: "Only when another medium rolls into view does the television's influence become perceptible"[1]. It is the same way with different interfaces for composition. Only when we switch from the computer to pencil and paper do we see all of the affordances and limitations of the former. Only when we switch from a graphical programming language like Max/MSP to a textual language like SuperCollider do we see the effect that the spatial, signal-flow approach had on our thinking.

This has important implications for creative blockages. When a particular way of interacting with the material is not yielding the right results, sometimes a change of approach—even a temporary one—will serve to highlight what was not working. For example, while working at the computer on a recent orchestra piece, I hit a wall and could not seem to move forward. When I switched to paper and pencil at the piano, suddenly the sound I was after came into sharper focus. The flatness of the computer playback had made me doubt the texture I was working with, and the workflow imposed by the interface had kept me from slowing down and engaging intimately with the counterpoint[13].

[13]It can go the other way as well: sometimes, when inputting a hand-written score into the computer,

This has important implications for composition pedagogy as well, since young composers are still in the process of learning what approach or combination of approaches work best for them. As Prof. Jaroslaw Kapuscinski expressed to me [23]:

> I think that sometimes students may be helped when they realize that it's something in their environment of work that's in their way rather than their own inabilities... Developing the transitions between [approaches], developing ways of placing things together that I need, it took me many iterations of different pieces to come up with and to be comfortable with.

Helping students navigate the myriad interfaces available to them, and encouraging them to try switching between them, not only furthers their development as composers; it does so in a way that avoids imposing a particular aesthetic.

3.5.2 The Overarching Process

Verplank describes a 'craft' tradition as one that involves "direct engagement with the materials producing immediate results" [5]. By contrast, he describes the 'design' process as one that involves perspective and reflection, and often cycles between "separate phases or modes." For instance, in designing a product, brainstorming might be followed by the testing and building of prototypes, then by evaluation and strategic adjustment, after which we may begin anew with brainstorming.

The compositional interfaces described above by and large represent tools of craft. So what of the reflective, multi-phase process of design? In the compositional context, this might correspond to the process of choosing an interface, working with it, evaluating the results, choosing a new interface based on that evaluation, etc. It, too, is a feedback loop, but a slower moving, more deliberate one, guided by our executive function.

I have found that the deliberateness of manual work was distorting my sense of the pacing of the piece. What seemed like two minutes of music turned out to last less than one.

For instance, in writing an orchestra piece, one might start by drawing graphs and writing words in a sketchbook. At some point, it might then feel appropriate to pivot to writing by hand on music paper, humming the melodic and motivic ideas to oneself. As the melodic content comes into focus, one might turn one's attention to harmony and begin to compose at the piano (or perhaps a digital keyboard with a custom microtonal tuning). Once enough of the material has come together, a full draft could be completed in short score, after which the final process of engraving might take place on the computer.

This, of course, is merely one way of charting a course from the initial creative concept to the final form of a work. Not only will different composers navigate the waters differently, but different pieces will require different combinations of tools and approaches. The decision of which interfaces to use is a highly personal (and aesthetically consequential) one, constantly evolving over years of practice and experimentation.

When I tell people that I compose by building and interacting with computer programs, I inevitably get asked something along the lines of: "Are you scared that eventually artificial intelligence will put you out of a job?" I always respond, with confidence, that I am not. It is true that recent advances in machine learning have led to tremendous new capabilities in pattern recognition, abilities that could be applied creatively to many compositional feedback loops[14]. However, it would be hard to imagine the higher-level process of design—with its complex and deeply human motivations—being carried out by a machine.

[14]Central to any such application would be the determination of a 'cost function,' which essentially is a judgment of how desirable or undesirable a given outcome is. In an artistic context, the decision of which cost function to use lies at the intersection of all sorts of aesthetic considerations.

Chapter 4

The Role of Interfaces in my Own Work

A few years ago it struck me that, for most of the pieces I write, a substantial portion of the time and effort spent composing goes into developing and understanding the process by which I am doing so. Like many composers (though perhaps more than most), I am constantly creating, and juggling, different computer programs, compositional strategies, and forms of musical representation.

The previous chapter offered a general theory of compositional interfaces, and of how the associated feedback loops affect the compositional process. In this chapter, I explicitly relate these ideas to my own practice as a composer.

4.1 Finding a Role for Technology in my Music

For a long time, I kept my interests in music quite separate from my interests in Math, Science and Technology. Although I experimented as a kid with writing simple music programs (like one that randomly combined short musical gestures into diatonic melodies), these were curiosities, not to be confused with my more serious musical endeavors. Even as an undergraduate, I was resistant to combining these interests. Only more recently, during my graduate studies, have I truly felt a meaningful and musically

successful merger of the two evolving.

Reflecting back on it now, I recognize that there was a significant anxiety underlying this reluctance. I sensed that algorithmic processes could be reductive, and that they could lead to a kind of toy music, one that lacked richness and creative spontaneity. In particular, I feared that the dry calculus of mathematical approaches would rob my music of its emotional core.

Indeed, in my early experiments with programming and music, I would invariably find my musical intuition taking a back seat to the more immediate demands of debugging and refactoring code. Sitting at the computer, I would lose touch with my body, my emotions, and my musical ear. It seemed that the deductive, problem-solving activity of computer coding was incompatible with the inductive, open-ended creative work of composition.

In the last years, however, I have come to learn that, while the mental states of coding and composing are difficult to combine simultaneously (at least for me), it is possible to incorporate them both into a broader workflow that produces exciting and musically satisfying results. The key seems to be to actively make space for my musical intuition to operate, both by designing low-latency interfaces that allow for a sense of flow (e.g. constructing a generative process whose parameters I can intuitively tweak with faders), and by frequently taking a step back and asking whether the current approach is serving my broader musical vision[1].

Often this requires a willingness to jettison the purity of strict, all-encompassing systems in favor of a diversity of sometimes conflicting approaches. I am particularly fond of Luciano Berio's [29] statement that:

> Every musical work is a set of partial systems that interact among them-

[1]Among other things, this means quickly getting to the point where I am receiving musically meaningful feedback.

> selves, not merely because they are active at the same time, but because they establish a sort of organic and unstable reciprocity.

Along a similar vein, Curtis Roads describes the process of composition as solving an n-dimensional jigsaw puzzle [30]:

> Multiscale planning [of a musical composition] can be likened to solving an n-dimensional jigsaw puzzle, where each piece in the puzzle is a sound object with a potentially unique morphology. How the pieces will ultimately fit together is not evident at the beginning... This process of solving a compositional puzzle can involve advance planning guided by predetermined design goals, but it can also be intuitive, exploratory, and open-ended. The process of aesthetic puzzle-solving can be quite difficult to describe precisely in words or in computer programs. How do human beings make aesthetic decisions? Our brains are designed to instantaneously tap largely unconscious preferences thousands of times a day in all of life's choices.

In my experience, algorithmic systems are only reductive when they blind us to the possibilities that exist beyond the system, when they keep us stuck in a small subspace of the n-dimensional space of possibilities available to us. In fact, algorithmic systems often suggest to me musical dimensions that I otherwise would not have considered. At its best, the computer is a wild collaborator, leading me to new and unexpected materials, relationships, and forms.

4.2 Randomness and Serendipity

Several compositions of mine, such as *One Wandering Night*, *A Social Network* and *Barlicity* were created via generative processes featuring some degree of randomness. In

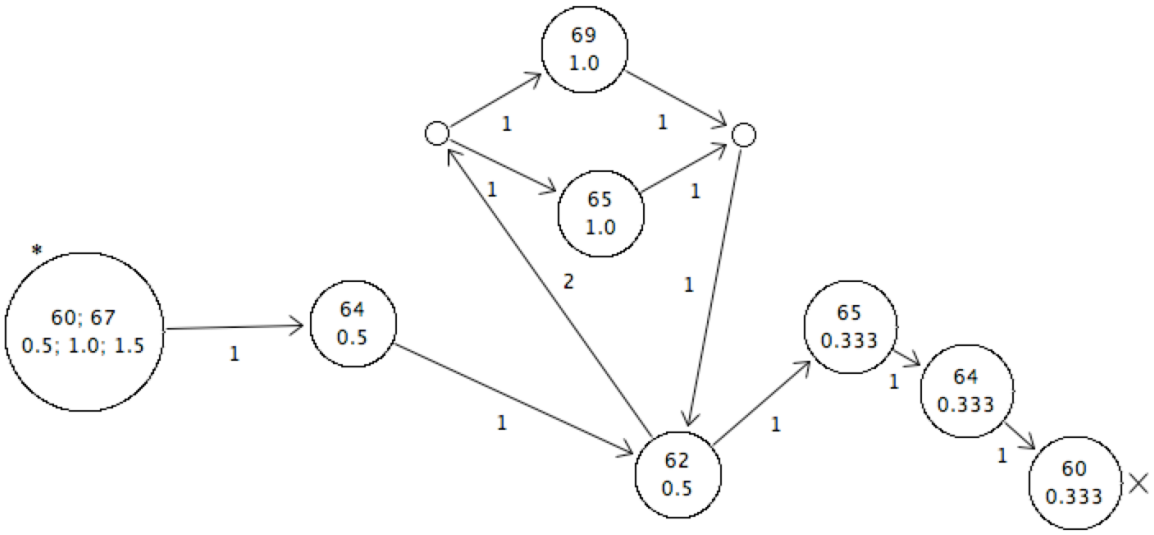

Figure 4.1: Screenshot from *Ramify*, an early program I created for exploring musical Markov chains.

each case, this randomness was shaped by an overarching concept and/or formal design.

One Wandering Night and *A Social Network* were both written with the help of *Ramify*, an early program I wrote for designing and generating music from networks of interconnected pitches and rhythms[2]. Ramify generates music by randomly traversing a network—such as that depicted in Figure 4.1—according to user-defined transition probabilities. Creating one of these networks is like composing a kind of 'meta-score,' which—through random variation—is capable of outputting any number of fixed realizations.

In the case of *One Wandering Night*, I created this meta-score intuitively. *A Social Network*, was instead based on a "found network"—a famous one within the field of Network Theory, in fact, referred to as "Zachary's karate club" [31]. Shown in Figure 4.2, this network represents the social associations within a university karate club, a club that ultimately split in two according to the color-coding in the figure. I scored the piece for two electric guitars, tuned to exploit two different harmonic series, with each guitar

[2]This is the program to which I alluded in the introduction (Chapter 1).

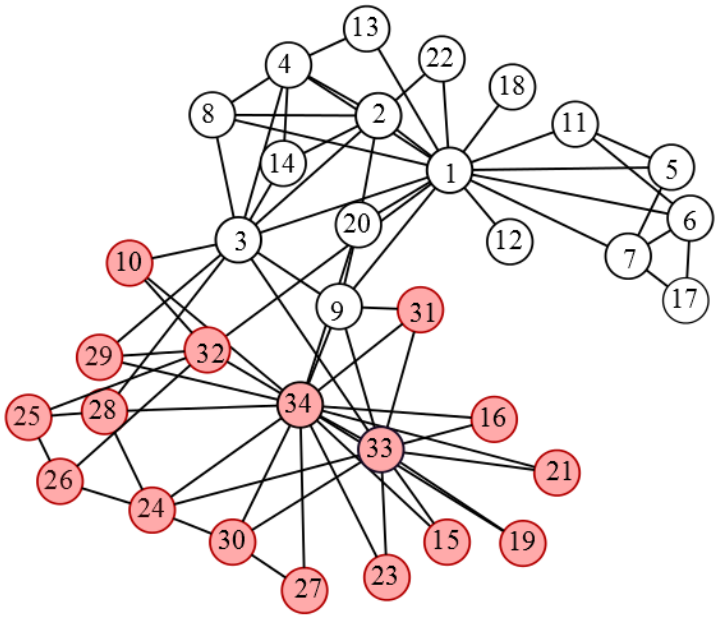

Figure 4.2: Visualization of "Zachary's karate club," the network I sonified to create *A Social Network* (from wikimedia.org).

covering the notes from half of the network.

Barlicity, for piano and fixed media, was written some years later, after I switched over to programming directly in Python. Although I found value in the graphical user interface provided by *Ramify*, I began to realize that it was coming at the cost of flexibility (see Section 3.4.1). For this reason, my focus shifted to creating a set of core libraries for playback and notation that I could reuse in piece after piece. These libraries became SCAMP (Suite for Computer-Assisted Music in Python)[2].

Like *One Wandering Night* and *A Social Network*, *Barlicity* incorporated randomness on a local level, with more deterministic processes shaping the overall form. As the name suggests, the piece owes a huge debt to the ideas of Clarence Barlow, many of whose algorithms I re-implemented in preparation for this piece. The piano part was based on a Bark Scale that was tuned according to Barlow's system of scale rationalization [32]. I then used a process called multidimensional scaling (again inspired by Barlow's example), to create a map of these pitches, such that more harmonically related pitches

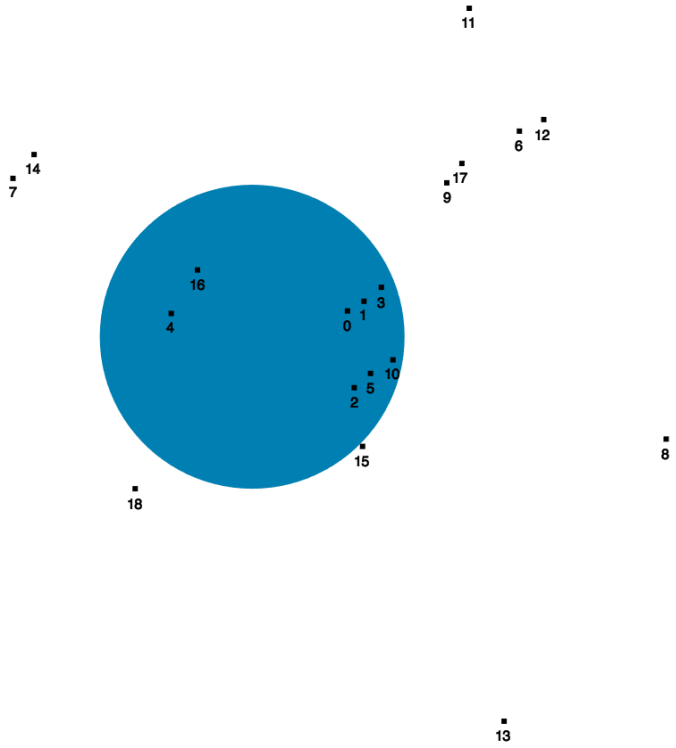

Figure 4.3: Screenshot from the visualization I created of *Barlicity*'s harmonic process.

are closer together. Finally, I allowed the piano to wander around this space, drawing pitches from different harmonic regions over the course of the piece.

Visual feedback (a screenshot of which is shown in Figure 4.3) played an important role in fine-tuning the harmonic process. Each numbered dot in the figure represents a pitch within the just-intoned Bark scale, with the distances between the dots based on harmonic relatedness. At any given point in time, pitches are pulled from within the shaded region, which moves and changes size over the course of the piece. Using this animation, along with my ear, I adjusted the width of the shaded region to an effective size.

For all three of these pieces, the large-scale shaping processes produced a clear, effective form, while the randomness helped to provide local interest and unpredictability (see

Section 3.3.3). Connecting these two levels of form often required some manual intervention. In *Barlicity*, for instance, I altered many of the accidentals towards the middle of the work to add harmonic tension and uncertainty. I also made numerous adjustments throughout in service of idiomatic piano writing.

My experience has been that algorithmic processes seldom, if ever, produce successful results without substantial intuitive adjustment. This can happen either at the level of the algorithm itself, or at the level of the generated material. Most of my work involves both kinds of manipulation. As I develop an algorithm, I use visual and auditory feedback to adjust key parameters in search of an effective result. At some point, I generally abandon the algorithm and start to work directly with the material.

4.3 Intermedia mappings

Although randomization obviously generates unpredictability, the two are not synonymous. In fact, random processes can be perceptually extremely static: take, for instance, a stochastic, asynchronous granular texture [33], or for that matter white noise, which consists entirely of random sample values. Among other things, it seems that the scale at which randomization occurs matters[3].

Moreover, deterministic processes can be just as, if not more, unpredictable than stochastic ones when they explore unfamiliar territory. In particular, I often take inspiration from translations between media. Since such translations are necessarily imperfect, and since in any case our brains process different media differently, unexpected results

[3]Even at larger scales, randomized results can start to feel statistically static. For instance, a piece that randomly changes texture every 1-5 seconds will sound initially chaotic and unpredictable; however, once the process has continued for several minutes, it becomes predictably unpredictable. Thus, the true variable at play seems to be the number of events we experience that are generated by the same random process. When those events are generated quickly (e.g. stochastic granular texture), a sense of stasis is arrived at quickly; when they are generated more slowly, it takes longer to recognize the underlying process.

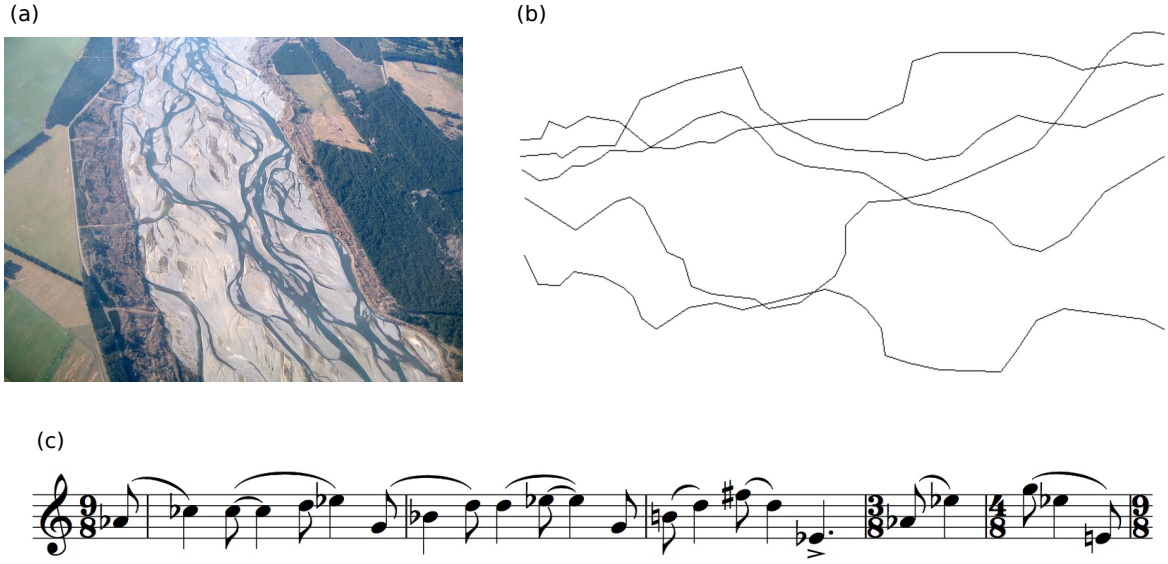

Figure 4.4: (a) Photograph of a braided river that I used in creating *Counterflow*. (b) Lines extracted from that photograph, which I then used to create (c) the opening melody of the vibraphone part.

easily crop up[4].

Counterflow

One of the earliest pieces in which I explored this potential was *Counterflow*, for clarinet, vibraphone, and double bass. *Counterflow* was inspired by the braided rivers of Denali National Park in Alaska. Braided rivers result when the water is very high in sedimentation; as the water flows, the deposit and buildup of sediment forces it to constantly change path. This results in a wide riverbed with many small channels, whose appearance changes constantly (see Figure 4.4a).

The continuously evolving and interacting channels reminded me of the voices of a compound melody, so I set about to sonify them, extracting a few of the main paths from Figure 4.4a to form Figure 4.4b, and then using them to generate the melody

[4]Even within the same medium, translations of scale work the same way: compression and expansion are imperfect, and perception operates differently at different scales. Consider, for instance, the effect of Leif Inge's *9BeetStretch* [34], which dilates Beethoven's 9th Symphony to a duration of 24 hours.

shown in Figure 4.4c. The swung rhythm of the melody was a happy accident, resulting from randomly selecting among durations of 1, 2, and 3 eighth notes. The non-octave-repeating scale used for the melody came from a computer program I wrote to search for scales rich in particular intervals.

Interestingly, this is as far as the process went. Having generated this opening melody, I wrote an upper and lower counterpoint to it by hand, and then treated the rest of the piece like a Bachian three-part invention. I worked in Sibelius, a natural interface for such work, since transposition, inversion, copying and pasting are all low-latency paths in the software. Although it generated no material beyond the opening vibraphone melody, the braided river continued to serve as a useful metaphor, suggesting a constantly shifting flow of melodic lines.

It might seem strange, even comically inefficient, that so much effort would go into a process of sonification that yielded just a few measures of music before being abandoned in favor of a more traditional toolkit. However, despite its small direct contribution, this process became the seed of a piece that I never would have written without it. In my work, algorithmic processes often act in this way: they are doors to new musical worlds, which can then be explored through whatever means feel most appropriate.

Leaf Loops

Another work based on a visual to auditory mapping is *Leaf Loops*, for violin and viola duet. Inspired by the contours of leaves, I wrote a program to trace these contours and translate them to looping sequences of pitches. As with *Barlicity*, the program—a screenshot of which is shown in Figure 4.5—provided both auditory and visual feedback. The blue dot quickly and repeatedly traces the outside of the leaf, while the red dot slowly follows a winding path in the interior (inspired by those of the leaf miner). A pitch is played every time the blue dot hits a sharp corner, and its pitch is determined

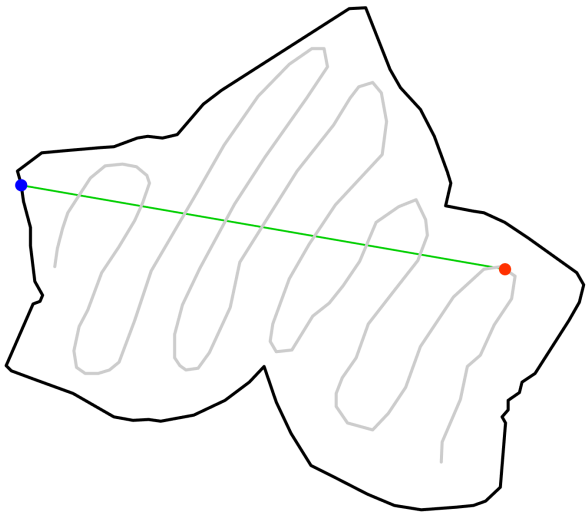

Figure 4.5: Screenshot from the program I used to create the raw material for *Leaf Loops*.

by the length of the line between the dots. Naturally, since the blue dot loops, so does the music, but it also gradually changes, as the red dot crawls along.

The result is a kind of minimalist music, a style which—for me at least—was foreign territory. In this case, unlike in *Counterflow*, I allowed the program to generate material for the entire length of the work. However, having done this, I distributed the pitches between the violin and viola intuitively, using Sibelius. I also frequently used my violin for tactile feedback. By slowing the process down in this way, I was able to ensure more interesting local texture throughout.

I also allowed myself the freedom to cut, to repeat, and to extend, operations that gave me the ability to intuitively modify the pacing and overall form. Undoubtedly, the decision to use these operations was affected by their ease and low latency in Sibelius; however, as with *Counterflow*, this felt in keeping with the nature of the music. The computer playback offered valuable information about pacing, and led me to construct an introduction, as well as to add accidentals at one point, when the harmony started to stagnate.

(a) (b)

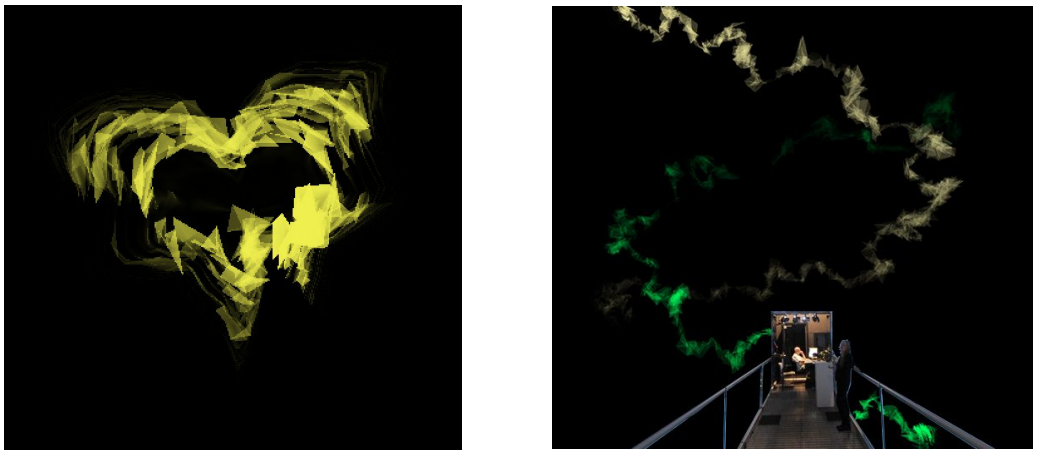

Figure 4.6: Screenshots from the surround, 3D visualization of Leaf Loops rendered in UCSB's Allosphere. (a) A single, rotating leaf shape. (b) Mock-up of two of these shapes revolving around viewers on the bridge of the Allosphere.

A year or two after I completed and recorded Leaf Loops, I used the piece to experiment with the reverse mapping: from auditory back to visual. This took the form of a 3D surround animation presented in UC Santa Barbara's Allosphere. In this presentation, a live spectrogram of the recording for each instrument was looped and stretched so that it outlined the shape of a leaf in three dimensions (Figure 4.6a). These two forms then revolved slowly around viewers in the center, leaving behind traces as they move 4.6b).

The rotating (and revolving motions) in the visualization were timed to coincide with different scales of periodicity in the music: measure, phrase, section[5]. One significant challenge was distilling the program down to a few key control parameters: width of the leaf shapes, distance from the viewer, degree to which they were above or below the horizon, choice of colors, etc. As discussed in Section 3.4.1, exposing these parameters allowed the work to be tweaked in a low-latency manner, something I hope to improve upon in future through the use of presets and physical controllers.

[5]The location of these junctures was marked manually while listening through to the recording.

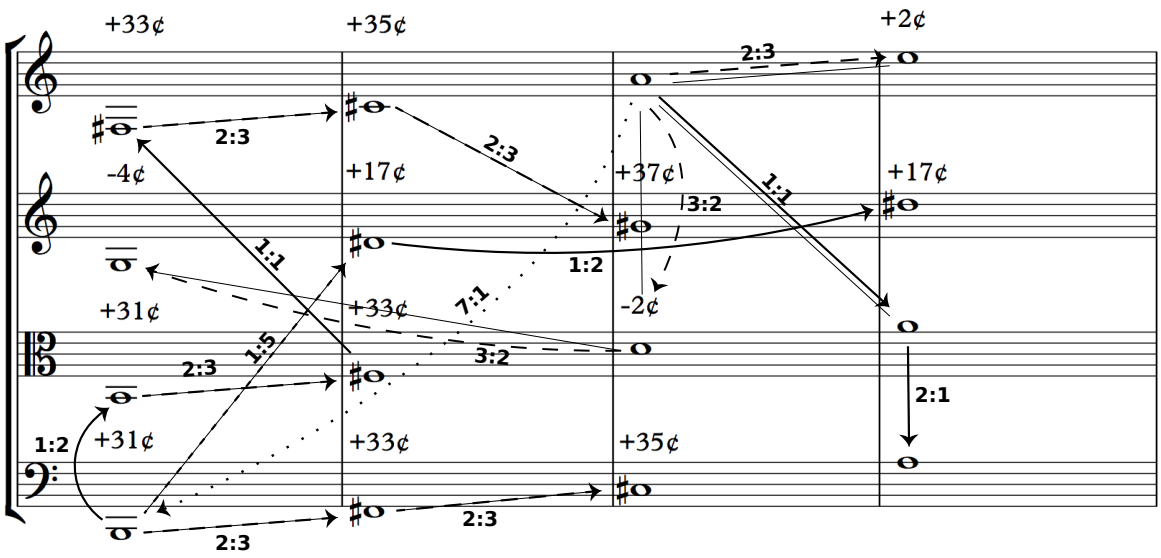

Figure 4.7: Tuning used for *Intimate Expanse*. Starting with the first violin's A, all other open strings are tuned in just intervals.

4.4 Explorable Spaces

One potential role for programming in the compositional process is to create interfaces for exploring unfamiliar musical spaces. These could include both literal spaces and more abstract spaces of musical possibilities (e.g. of harmonies, of timbres, of gestures).

Intimate Expanse

In *Intimate Expanse*, for string quartet, I began by creating a unique scordatura (see Figure 4.7). The purpose of this was to facilitate the playing of just intervals, especially with open strings and natural harmonics. Working within a new microtonal tuning system places the composer in foreign harmonic territory. This foreignness is, of course, the point, but it requires the composer in some way to find their bearings. An exploratory interface is the perfect way to do this.

One possible such interface, for example, would be a microtonally-tuned keyboard, allowing the composer to explore harmonies and harmonic progressions through imme-

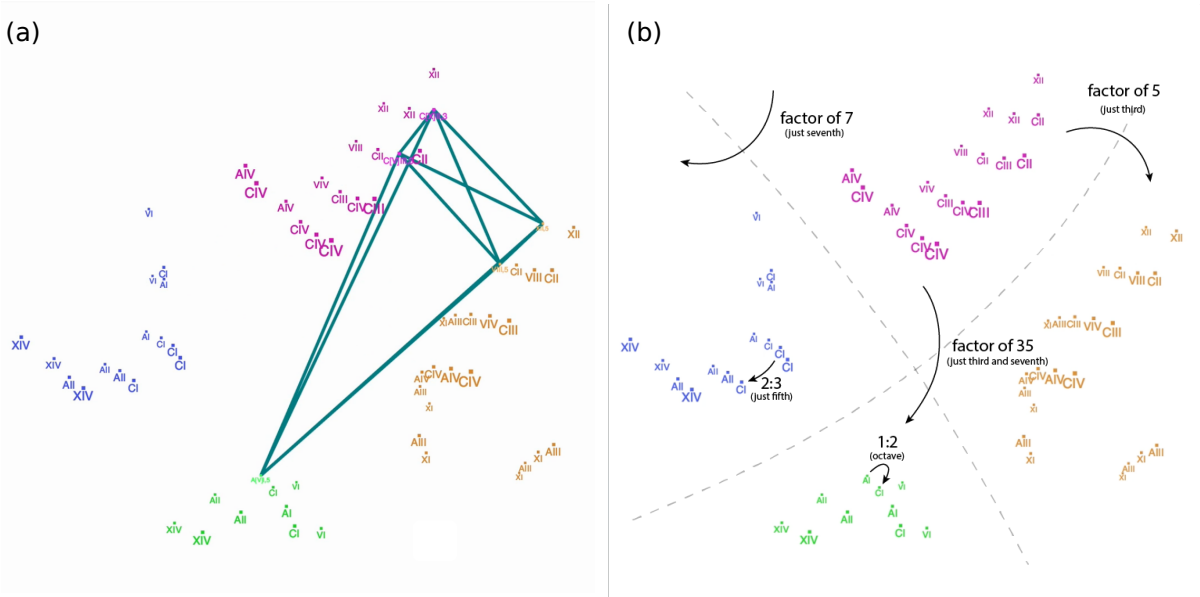

Figure 4.8: (a) Screenshot from the interface I used to explore the microtonal tuning for *Intimate Expanse* (currently playing back a chord). (b) Diagram showing how four main harmonic regions are delineated by factors of 7 and/or 5 in frequency.

diate auditory, visual and tactile feedback. My approach, instead, was to use the same process of multidimensional scaling that I did in *Barlicity*, but this time to make the map interactive (see Figure 4.8a). In contrast to a microtonally-tuned keyboard, distance here gives feedback about harmonic relatedness, rather than pitch spacing. As factors of 5 and 7 represent more distant relationships, they split the space into four quadrants.

Using this interface, I was able to explore the space of possible chords, collect a few particularly evocative ones, and ultimately arrange them into a progression that traversed the space in a circular manner [6]. Once I found this progression, I arranged each chord for string quartet using paper and pencil, guided by my imagination of the resulting texture. I was then fortunate to have a preliminary reading with the LA-based Formalist Quartet, whereby I could hear and refine these textures.

I did the actual assembly of the work in the Digital Audio Workstation REAPER by recording and pitch shifting my violin. This allowed me to accurately judge pacing,

[6] See `http://marcevanstein.com/Compositions/IntimateExpanse/IntimateExpanse.html`

and to stay connected with the actual sound of the harmonies. (I needed this auditory feedback, because these harmonies—and the progressions they formed—were still relatively unfamiliar to me.) The last, and final, step was to notate the music in Sibelius. Due to the extensive use of proportional rhythmic notation, this was a highly awkward process—so much so that this interface could not have been used for any real composing, only for engraving.

Formations

Formations, for solo piano and cued samples, also featured an interface for exploring a space of musical possibilities, but this time an analog one. I began by writing several very short gestures at the piano. Having collected a sufficiently diverse assortment, I proceeded to combine them together in various orders and transpositions to form longer gestures. These longer, compound gestures I then combined to form short phrases. Finally, I combined those phrases into sections, and those sections into an overarching form.

By following this process, I was working with a specific compositional interface, one which explored a very particular set of musical possibilities. Although this might seem simply to be the interface of "composing by hand at the piano," it was actually something far more specific, in which only particular actions of modification were allowed. Furthermore, the 'composition', while in progress, was a collection of unordered fragments; the visual feedback no longer presented a timeline, but rather a space of possibilities[7].

I in no way mean to suggest that this interface is a revolutionary reinvention of what it means to compose at the piano. Quite the contrary: I imagine that there are probably as many interfaces for composing at the piano as there are composers. The key point is that, by altering the process by which I interacted with and represented the material,

[7]In this way, it is very similar to the 'mobile form' of Stockhausen's *Klavierstück XI* [35]. The difference is that this mobility of parts occurred during the process of composing, rather than in the final product.

I was effectively using a distinct interface, one which yielded results unlike other pieces that I have written.

4.5 Musical Contour Spectra

One last technique to discuss, which I have used in several recent works, is the analysis and resynthesis of what I call 'Musical Contour Spectra'. This technique is really no more than the application of spectral (Fourier) analysis to higher-level musical parameters—e.g. pitch, duration, dynamics—rather than to sound waves directly. However, don't be fooled by the references to "Fourier" and "spectra": this has nothing to do with timbre. It is merely that the same mathematical tool-set usually used to analyze timbre (at least in a musical context) is being repurposed for other kinds of musical information.

The technique is explained in detail in Appendix B. However, for the purposes of the following discussion, the important thing to understand is that it is a way of splitting a musical contour into its faster- and slower-moving components. The pitch of a melody, for instance, might have an overall rising and falling shape; within this shape there may be several smaller-scale dips or irregularities; finally, on the smallest scale, there might be ornamentation around particular notes. What the analysis and resynthesis of contour spectra allows us to do is to tease apart these different scales of motion, potentially creating new melodies that feature only some of the elements of the original. For example, we could create a new melody that keeps the overall rising and falling contour of the original, leaves out the middle-scale irregularities, but keeps the most local level of ornamentation.

Effectively, the many options for analysis and resynthesis become a space of musical possibilities. Naturally, this led me to create a program to explore that space, which I verbosely titled the "Spectral Musical Contour Explorer". A screenshot of this program

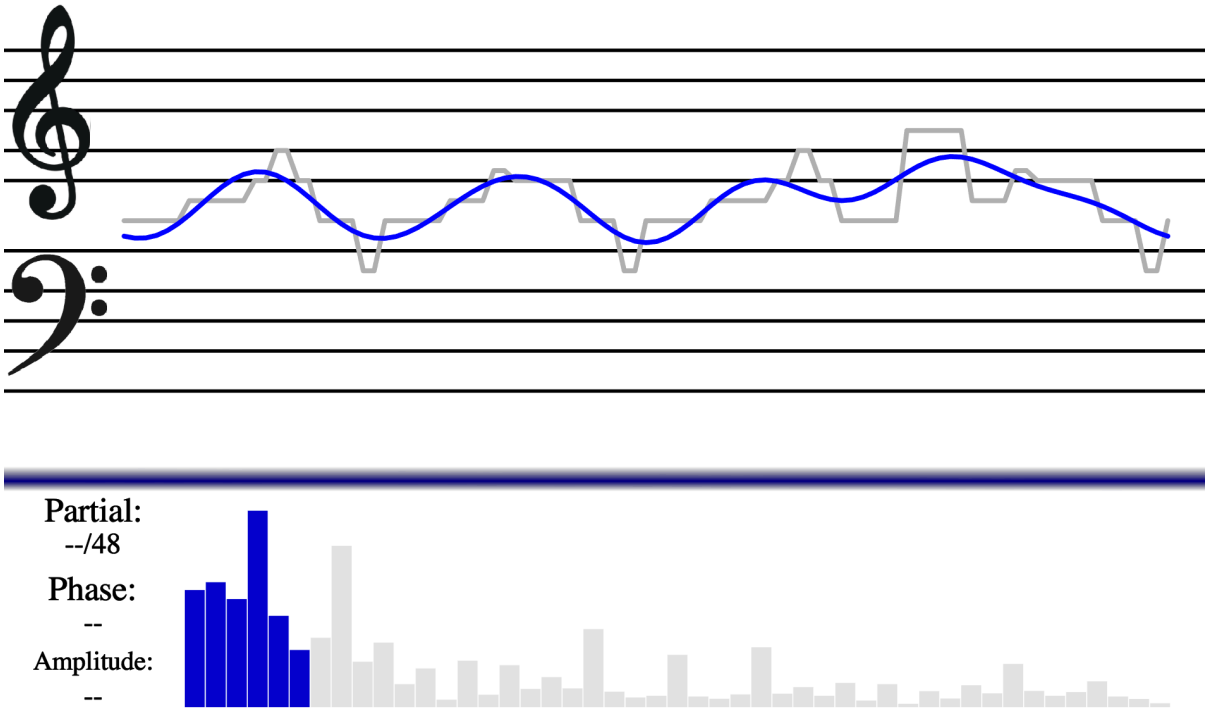

Figure 4.9: Screenshot from my program, "Spectral Musical Contour Explorer." The bottom half of the screen shows which parts of the melody are kept active, with the bars to the left representing smaller-scale motion, and those to the right representing larger-scale-motion. The top part of the screen plots both the original melody (in gray) and the reconstructed melody (in blue).

is shown in Figure 4.9. The pitch-contour of the original melody (in this case, "Pop Goes the Weasel") is plotted in grey against a grand staff. The pitch contour of the current reconstruction is overlaid in blue. At the bottom of the window, we see which speeds of motion are being included in the reconstruction; in this case, only the slower-moving motion (left side) is included, with the faster-moving motion grayed out (right side). As we can see looking at the grand staff above, this results in a "smoothed out" version of the original melody.

This program gives very useful visual feedback, allowing the user to see the kinds of transformations they are imposing on the melody. It also allows the user to play back the reconstructed melody with a sine-tone. However, as is often the case, this program is too

limiting when it comes to creating music; when actually composing with this technique, I have always fallen back to programming in Python, using the graphical program as a guide when I needed better visual feedback.

Adagio Cantabile

The first piece in which I made use of this technique was *Adagio Cantabile*, for oboe and guitar. Using as source material the main theme of the second movement of Beethoven's *Sonata Pathetique*, Op. 13, I created a Python script that treated the pitches and durations of the melody as a series of samples, and then performed Fourier analysis and resynthesis of each of these independently. This is essentially the same process described above, except with rhythm as well as with pitch[8].

As usual, presented with a space of musical possibilities, my next step was to create an interface to explore these possibilities. I set up the melody to play back in a loop, while I used faders from a midi controller to adjust the degree to which faster and slower components of the melody were included in the reconstruction. When I found something that I liked, I recorded the snippet and had the program generate notation for it. (A happy accident occurred in this process: since I had set up my Python tools to treat pitch as continuous, rather than discrete, I ended up with beautiful microtonal melodies. As a result, microtonal inflections on the oboe became a crucial part of the piece's identity.)

Having generated this collection of snippets, I was faced with the process of assembling them into larger phrases, and ultimately a form. I decided to do this with pencil and paper, humming the music to myself.[9]. In retrospect, the reason for this was that it

[8]The results turned out to be quite interesting with rhythm: When no frequencies (except DC) of oscillation were present, the rhythm was static, with all notes the same length. When slow oscillations were included, the rhythm started to accelerate and decelerate at the faster and slower parts of the melody. As I included faster and faster oscillations, these accelerandi and decelerandi became more and more local, until all of the detail of the original rhythm was recreated.

[9]...sitting on a folding chair, overlooking the ocean.

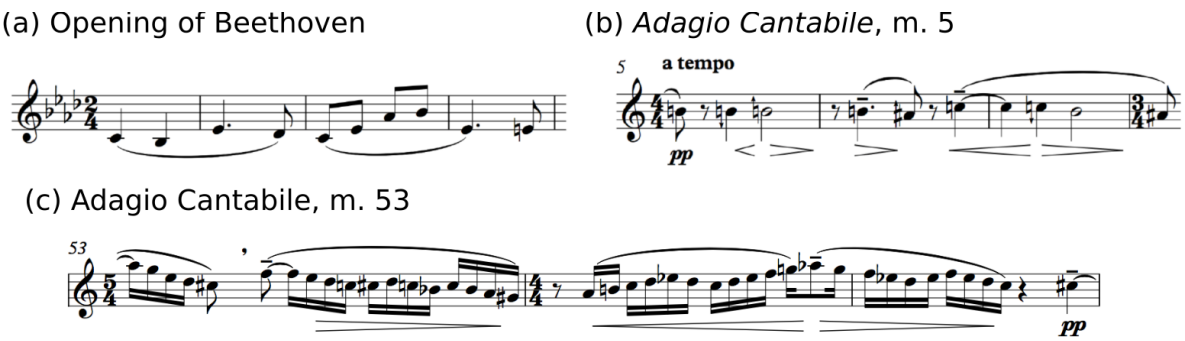

Figure 4.10: Comparison of (a) the opening melody from the second movement of Beethoven's *Pathétique* and (b) a corresponding passage in *Adagio Cantabile*. (c) is an excerpt from towards the end of the work.

provided the right kind of feedback; as oboe is a wind instrument, the process of humming the melody (or even of imagining doing so) kept me connected to the breath, and to the effect of different tessiture. The continuous, flexible input provided by the pencil, as discussed in Chapter 3, was also valuable.

Figure 4.10 compares the opening melody of the Beethoven (a) with an excerpt from *Adagio Cantabile* featuring a partial reconstruction of the melody, in which all of the larger contour of the melody has been removed (b). Figure 4.10c, from near the end of the work, shows a reconstruction of the melody with all of the local ornamentation removed, such that it sweeps gradually up and down. In this latter case, I allowed myself considerable flexibility in choice of accidentals, letting my ear guide such decisions intuitively. The same was true for the guitar part, which applied the same process to a reconstruction of the lower three parts of the opening chorale texture of the Beethoven.

Unraveled

The second piece in which I used this technique was *Unraveled* for Percussion Quartet and Impossible Electronic Orchestra. The title suggests exactly the kind deconstruction that this process of analysis and resynthesis entails, while also giving a hint as to the

source material: the famous melody from Ravel's *Bolero*. This melody is played, in various degrees of reassemblage, by the "Impossible Orchestra," which consists of pitch-bent samples of real orchestral instruments.

Thus, as with *Adagio Cantabile*, the contour in question is a melodic pitch contour. An added wrinkle in this case, however, is that rolls in the percussion parts are used to emphasize and complement the various oscillations within the melody[10]. At the same time, the percussionists are engaged in a parallel process (of a different sort) that gradually reconstructs the (in)famous Bolero rhythm.

The process of composing *Unraveled* was the most challenging I have experienced to date in terms of the juggling of interfaces. I began with a Python script, which I used to generate both the notated music for the percussionists and the music of the electronic orchestra. This was already complicated and awkward: I had made recordings of every kind of percussion stroke and roll that I intended to use, and was triggering them from the Python script; at the same time, the script was in communication (via OSC) with SuperCollider, which was pitch-bending the orchestral samples.

This, however, was only the beginning: The algorithmic process resulted in far more material than was realistically playable by a percussion ensemble, and thus the material needed to be paired down by hand. Moreover, the notation it generated was not true percussion notation; pitched notes stood in for the symbols that would ultimately be used. Finally the electronic material was too simplistic, too flat, and likewise in need of direct manipulation.

What I needed at this point was an interface in which I could edit both the notated percussion part and the electronic part, as well as get reasonable feedback about how each might sound both on their own and together. As it turned out, it was hard enough

[10]If you are familiar with Fourier analysis: each roll would emphasize a particular, prominent frequency. Since the rolls are expressed with volume, rather than pitch, the process is additive, allowing the effect of different frequencies to accumulate.

to get Sibelius to play back custom recordings of percussion instruments, let alone to map those recordings to the correct notations. Ultimately, I had to create a Sibelius document with three different staves for each instrument (one each for sounds produced with the hands, brush and drumsticks). I used that document to refine the music for the percussionists, after which I rendered its audio and transferred the notation to simpler document that could no longer playback sound. Finally, I imported the rendered audio into REAPER and manipulated its electronic accompaniment to my liking.

I have certainly left out and/or forgotten many details in the above description. The whole process took years, frequently stalled by one or another technical hurdle. This piece, more than any other, convinced me of the creative frustration that can result from interfaces that are poorly matched to one's creative goals.

Anamnesis

A third work that I composed using spectral musical contour analysis is *Anamnesis* for Chamber Orchestra. *Anamnesis* differs from *Adagio Cantabile* and *Unraveled* in that the musical contour being analyzed was a dynamic contour, rather than a pitch contour. Here again, I used a famous work as the source material for analysis: the *Alegretto* from Beethoven's *Symphony No. 7*.

Figure 4.11a shows a screenshot (from the program Audacity) of the recording by Carlos Kleiber and the Vienna Philharmonic Orchestra. Figure 4.11b shows this same recording, as loaded into my "Spectral Musical Contour Explorer": a dynamic contour has been created by calculating RMS values for every half-second of audio, effectively resulting in a unipolar version of what we see in Audacity. Figures 4.11c, d, and e show three examples of strong periodicities found in the dynamic contour through analysis. Notice, for instance, how Figure 4.11c highlights the two main peaks of intensity within the movement.

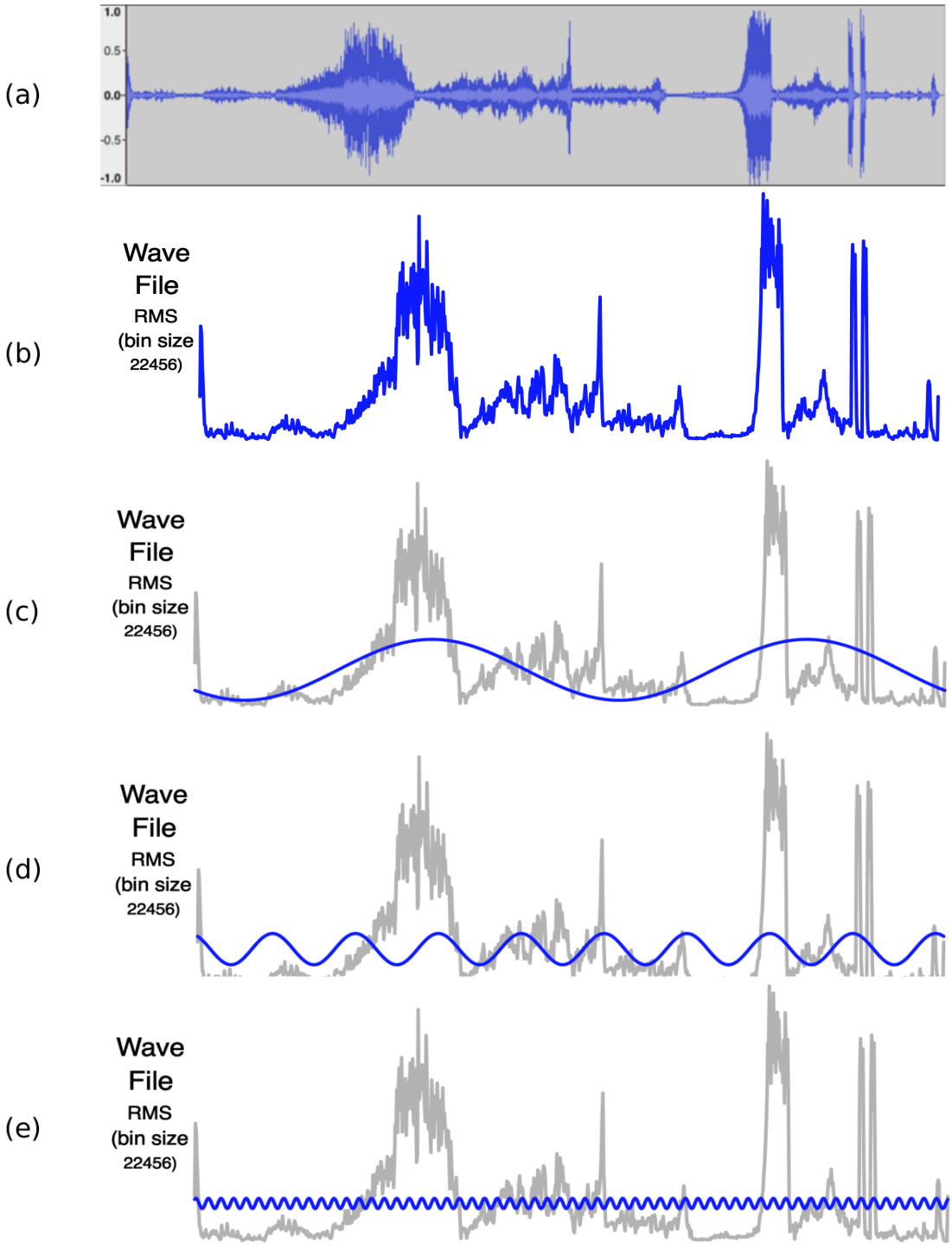

Figure 4.11: Analysis of the dynamic contour of the *Allegretto* from Beethoven's *Symphony No. 7*. (a) The dynamic shape of the movement as shown in Audacity. (b) The same contour, as represented in "Spectral Musical Contour Explorer." (c), (d), and (e) Three prominent periodicities found at different scales.

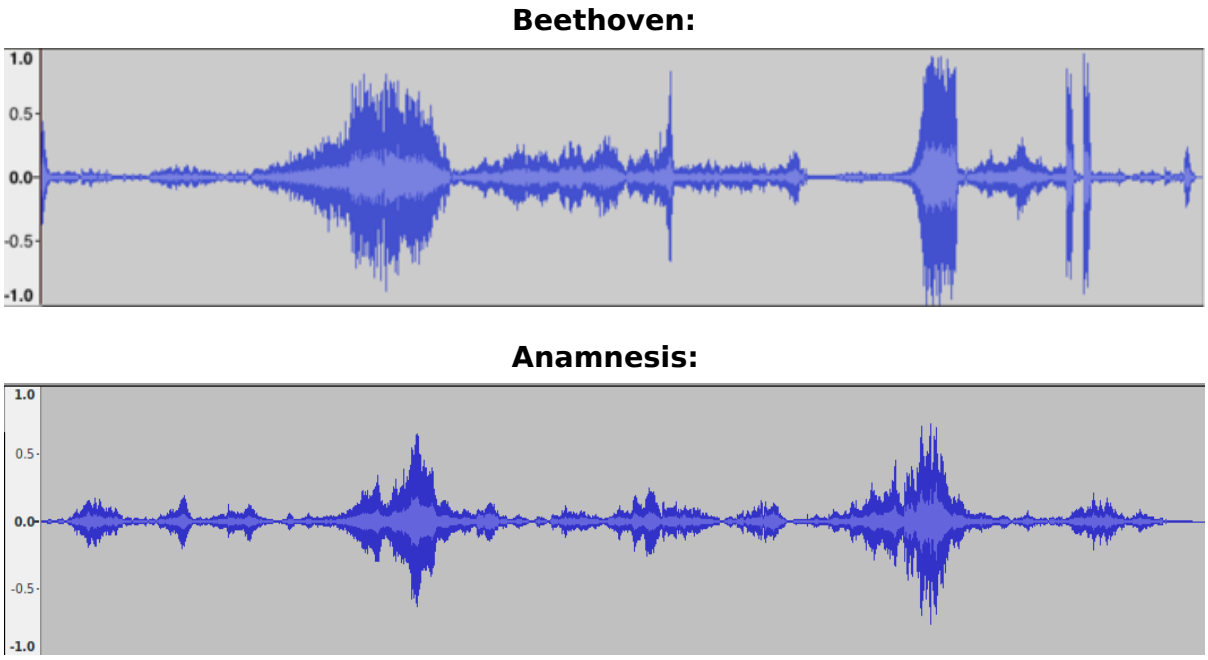

Figure 4.12: Direct comparison of a recording of the original Beethoven with a recording of *Anamnesis*.

In Python, I created a script to orchestrate these different layers of motion, layering them on top of one another. Some instruments played the larger swells, others played the mid-level swells, and still others played the fastest-moving swells. The exact notes were shaped by another, parallel process, which analyzed and, in some sense, inverted the harmonic language of the Beethoven, turning points of harmonic attraction into points of repulsion.

Part of the rationale for this process was that this would result in a new work that borrowed to some extent from the effective dynamic contour and harmonic scheme of the original Beethoven. Indeed, as Figure 4.12 shows, the dynamic shape comes through fairly clearly in the final work[11]. At the same time, as discussed in Section 4.3, the imperfections in this process of translation were part of the point, and led to interesting

[11]The reader will certainly notice some differences, however. In particular, since swells from different sections of the orchestra cannot destructively interfere, it is impossible to get the sudden jumps in dynamics found in the Beethoven. This smearing became a distinctive feature the work.

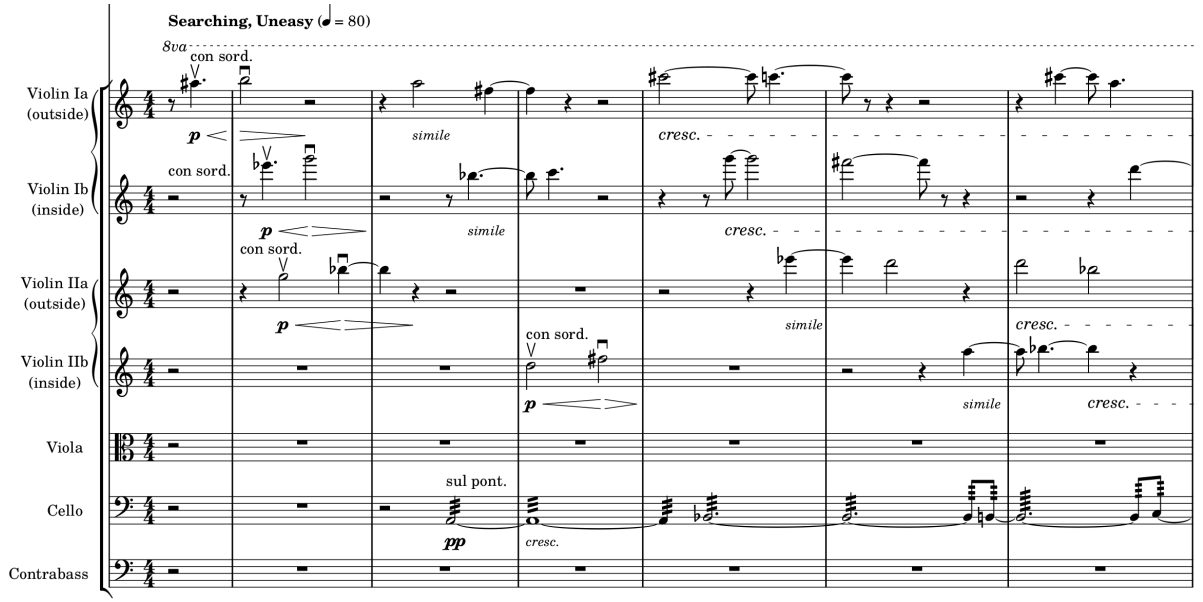

Figure 4.13: String excerpt from the opening of *Anamnesis*.

and unexpected results.

One characteristic texture that resulted from the process of analysis and resynthesis was the overlapping of many short swells, such as that shown in Figure 4.12e. This texture, present throughout but most prominent in the violins at the beginning and end of the work, is shown in Figure 4.13. Below, one can see a larger swell beginning in the cello section.

The processes described above created the basic material for the entire work: a block of granite out of which to carve the final product. As usual, it was by no means the first version of the algorithm; I tried many different mappings and orchestrations before committing to a particular version and generating the notation. From there, I began to work in Sibelius, shaping the material by ear.

This worked well to an extent, but at some point I hit a wall. The visual feedback of the large orchestral score was unwieldy: the information was completely uncompressed, overwhelming in its size and scope. On a horizontal monitor, it was hard to even see

an entire page at once without rendering the details illegible. It was also hard to get an accurate sense of the voicing and balance of chords, something which the playback was likewise misrepresenting. Meanwhile, the workflow on the computer tended to encourage quick, shallow actions, like copying and pasting, and to discourage slower, more deliberate engagement with the material.

As a result, I began to write out some of the especially dense passages by hand in short score. This was a marked improvement, but I ran into a different problem: often the texture was so complicated, with so many independently moving voices, that I could not get an accurate sense of it by playing it on the piano. Ultimately, the interface that worked best for me was to simultaneously write the score out by hand while entering it into a DAW to be played by virtual instruments. The playback in the DAW was more customizable than in Sibelius, and the process of writing by hand allowed me to make more conscious, deliberate decisions.

It is interesting to consider why I was working in Sibelius in the first place: it was simply the quickest way to start manipulating the notation I had generated from my Python script. However, quick and easy though it was, I might have been better off copying down the notation by hand from the beginning. This may be an important consideration for other composers who, like me, write computer programs that generate notation. The generative process may serve one's compositional aims, but the tools it then pushes one towards may not.

4.6 Conclusions

The creation and selection of interfaces has becoming an increasingly important part of my creative process. Most new works I create incorporate a newly designed interface somewhere in the process, and all new works I create in some way modify the order and

function with which I use the interfaces I already have. At this point in my career, the interface(s) I use to create a work have become one of the core elements of that work's identity.

Although the question of which tools to use may seem like a superficial, technical matter, I find that it touches on the deepest questions of my ultimate goals as a composer[12]. The purpose I wish my music to achieve is tied unavoidably the form(s) in which it is represented, and the means of modification and feedback attached to that representation affect my subjective experience of the work I am creating. The more effective I make my tools, the more clearly I see the relationship between my actions my ultimate goals[13].

Increasingly, I find that creative blockages turn out to be interface problems. In the past, when a particular approach stopped working, I would feel helpless. Now, instead, I examine my interface, questioning whether each element of it is serving my musical goals. Often this leads to the realization those goals themselves are in need of clarification.

My ultimate aim is not tool-building; it is only through the music I create that my interfaces prove their value. However, given that interfaces are unavoidable, I strive to see them as a gateway to the central questions of what I hope to achieve as a composer.

[12]It is the same with the technical aspects of playing the piano: I practice the choreography of my arm and the transfer of weight into my fingertips because it helps me control the tone, voicing, and dynamic shape of musical gestures, which in turn lets me clearly convey the intricacies of counterpoint and phrasing in the works I admire. If, instead, I cared only about giving the impression of speed and virtuosity, I would simply focus on remaining relaxed and turning up the metronome.

[13]It is also true that the more clearly I understand my musical goals, the effectively I can construct my tools to serve those goals.

Appendix A

Portfolio of Compositions

A Social Network (2014) for two electric guitars	Score	Recording
Romance? (2014) for Viola and Piano	Score	Recording
Like as the Waves (2016) for Tenor and Piano	Score	Recording
Counterflow (2015) for Vibraphone, Clarinet and Bass	Score	Recording
Inroads (2014) for Stereo Fixed Media	—	Recording
Adagio Cantabile (2016) for Oboe and Guitar	Score	Recording
Leaf Loops (2016) for Violin and Viola	Score	Recording
Barlicity (2017) for Piano and Fixed Media	Score	Recording
Intimate Expanse (2018) for String Quartet	Score	Recording
Unraveled (2018) for Percussion Quartet and Fixed Media	Score	Recording
Anamnesis (2019) for Chamber Orchestra	Score	Recording
Marc of Santa Barbara (2019) for Flute and Live Electronics	Score	Recording

Appendix B

Analysis and Resynthesis of Musical Contour Spectra

This appendix describes in some detail the technique of analyzing and resynthesizing musical contour spectra using Fourier analysis. We begin with a description of Fourier analysis in the context of its most common musical application: the breaking down of a complex timbre into a spectrum of frequencies though analysis of the sound waveform. There is nothing new in this description, and those readers familiar with the mathematics of the Fourier Transform can probably skip it.

With this background in mind, we then take advantage of the fact that Fourier analysis is an abstract mathematical tool, and apply it to a very different kind of musical contour: the pitch curve of a melody. By applying Fourier analysis to this more abstract musical contour, we can uncover interesting information about what makes a melody tick, and how it evolves at different time scales. Branching out from pitch contours, we then consider other possible musical contours, such as variations in tempo or note duration, or variation in volume over the course of a movement.

Finally, we look at some of the possible creative applications of this process, such as decomposing the contour of a well-known melody, and then altering and reassembling it in novel ways.

B.1 Fourier Analysis

Fourier analysis is what we use when we talk about the spectrum of a complex sound; for instance, when we say that a clarinet tone has only odd harmonics, or that the first harmonic of a trumpet is stronger than its fundamental, we are referring to the results of Fourier analysis. The basic idea is that a complicated motion—in this case, the motion of an air particle under the influence of a trumpet or clarinet—can be decomposed into a superimposition of very simple motions at different speeds.

Figure B.1a shows the waveform (i.e. graph of the fluctuation in air pressure) of several periods of a trumpet tone. The unique shape of the waveform creates the trumpet's sonic signature. Note that these fluctuations happen very quickly; the shape repeats three times over the course of 5ms, which translates to 600 oscillations per second (Hz), or roughly a concert D5.

What Fourier synthesis does is break down this complex signature into a sum of simple motions—sine waves, in fact—at integer multiples of the 600Hz frequency of the complex pattern. Thus, for a 600Hz trumpet tone, we have components at 600Hz, 1200Hz, 1800Hz, 2400Hz, etc., which we would term the 1^{st}, 2^{nd}, 3^{rd}, and 4^{th} 'harmonics' or 'partials'. Each partial has its own weighting ('amplitude') and alignment ('phase').

We can see what this looks like in Figures B.1b-e, which show the 1^{st}, 2^{nd}, 4^{th}, and 5^{th} partials of the trumpet waveform respectively (the original waveform is shown in gray for reference). The first partial completes one cycle for every cycle of the complex trumpet tone; thus it, like the trumpet tone, is oscillating at 600Hz. The second partial completes two cycles for every cycle of the trumpet tone; thus it is oscillating at 1200Hz. Of the four partials shown, notice that the second partial is the strongest, and that each of the sine waves is positioned so that its peaks and valleys align as well as possible with the peaks and valleys of the complex waveform.

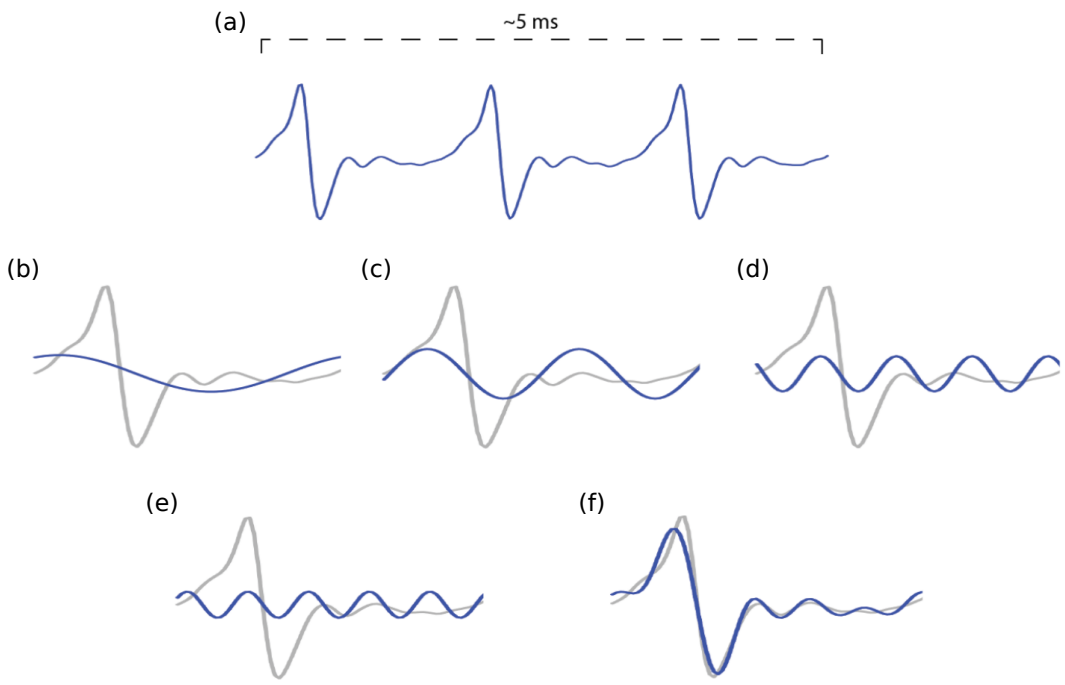

Figure B.1: (a) Short excerpt from a trumpet waveform showing a repeated fluctuation in air pressure. (b)-(e) One period of that fluctuation, with the 1st (fundamental), 2nd, 4th, and 5th harmonics isolated, respectively. (f) The recombination of those harmonics (blue) as compared with the original waveform (light gray).

Figure B.1f shows the sum of these sine waves, which (as if by magic) very nearly reproduces the original trumpet waveform. One of the remarkable properties of waves is that they simply add together in the medium in which they propagate. For instance, if an air particle is under the influence of two waves, one of which is pushing it to the right with a force of 5 and the other to the left with a force of 3, it responds by accelerating to the right under a net force of 2. For this reason, there is no physical or acoustical difference between the simultaneous sounding of the simple sine waves in B.1b-e and the sound of their sum in B.1f. If we want to reproduce the original trumpet wave with perfect fidelity, we simply need to include the remaining relatively weak higher harmonics.

It turns out that there is only one way that we can add together sine waves in this way to produce any given complex wave shape, and we call this unique combination of

harmonics with different amplitudes and phases its 'spectrum'. This is why we can say that the sound of a trumpet has a strong second harmonic: the effect of the complex pressure wave produced by a trumpet is identical to the effect of a multitude of carefully tuned sine waves added together, and this unique spectrum has a strong second harmonic.

Sonically, this spectrum translates to a certain timbre[1]. However, this is only because we were analyzing a fluctuation in air pressure. In general, 'spectrum' refers to the way in which a complex motion can be broken down into many simple motions moving at different rates, and it is this definition that comes into play in the following discussion.

B.2 Fourier Analysis of Melodic Pitch Contour

Because Fourier analysis is a mathematical abstraction, it can be just as easily used to analyze the variation in any other musical parameter, at any time scale[2].

Figure B.2a shows the melody for "Pop Goes the Weasel" in traditional music notation. By replacing the noteheads with a line, we can see in Figure B.2b that the pitch contour is a wave like any other; the only difference from the trumpet waveform above is that this is a wave representing the motion of an abstract musical parameter, rather than of air pressure directly, and that the variation is on the scale of seconds rather than milliseconds.

This means that there is no reason we cannot apply Fourier analysis to the pitch contour, just like we did with the trumpet waveform. As with the trumpet, these oscillations operate at 1x, 2x, 3x, etc. the frequency of the melody itself[3], and each of these 'partials' has its own amplitude and phase, with some partials being especially influential.

[1]Timbre is actually a very complex topic, also incorporating aspects of sound that are hard to analyze spectrally.

[2]Or, for that matter, non-musical parameters, such as economic time series.

[3]For those more familiar with Fourier analysis, it will be apparent that I am using a window size equal to the whole length of the melody.

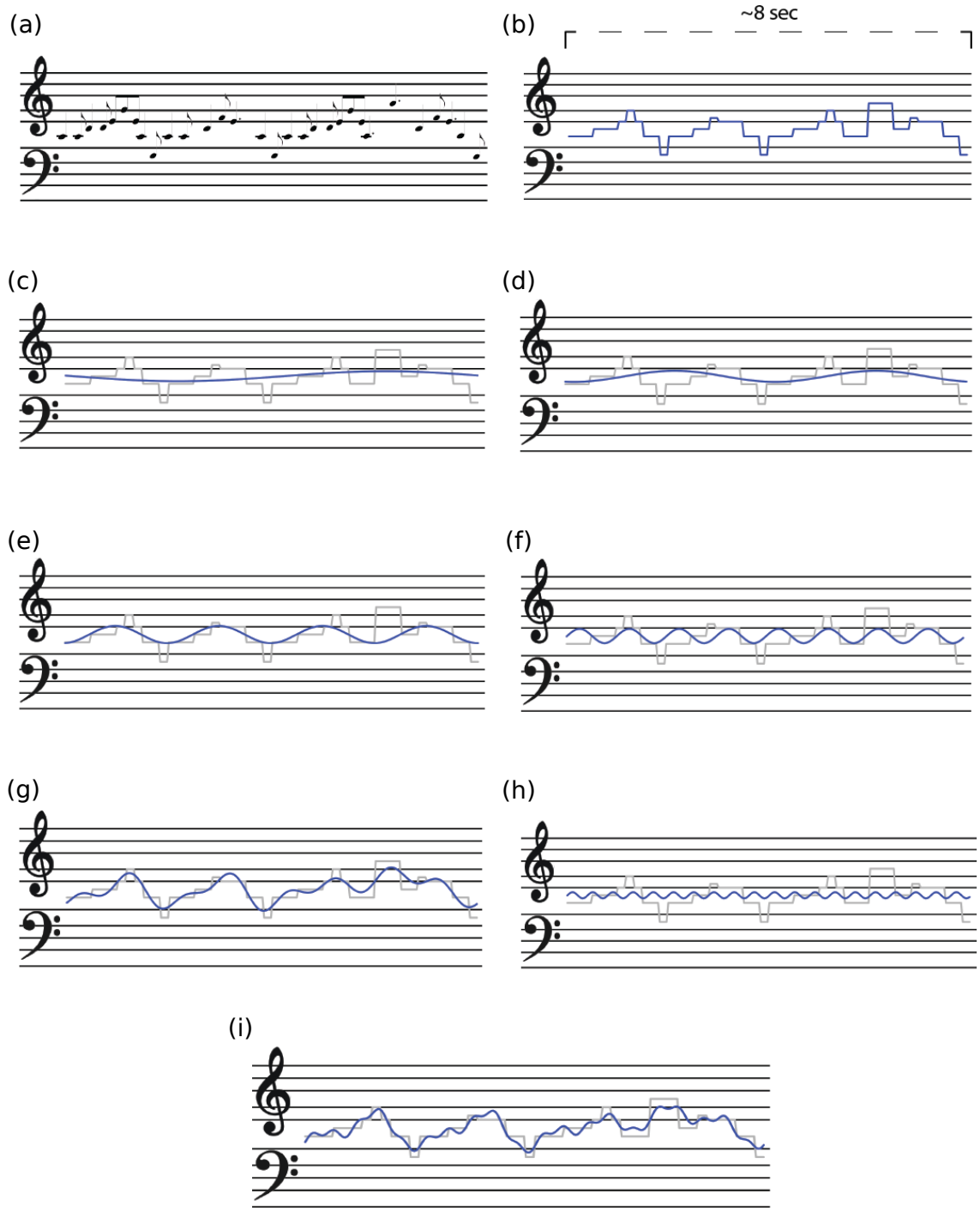

Figure B.2: Analysis of the pitch contour of "Pop Goes the Weasel." (a) The melody in (compressed) traditional notation against a grand staff. (b) The same as (a), except represented as a line graph. (c)-(f) Several of the lower 'partials' in the spectrum, which add together to form (g). (h) A faster-moving upper partial. (i) The contour in (g) combined with the higher partial in (h).

Figures B.2c-f show some of the lower partials. The first partial (Figure B.2c) is not particularly strong, but the phase is nevertheless aligned so that the peak coincides with the highest note (A4) of the melody. The same can be said of the second partial (Figure B.2d). The strongest component is the fourth partial, which completes four full cycles over the course of the melody (Figure B.2e). Why is this?

The reason is that the melody itself is in four parts! Each two-bar phrase (if notated in 6/8) starts low in the first bar and peaks at the beginning of the second bar, with one exception: the final phrase begins with a peak, the highest note of the whole melody. In order to compensate, the first and second partials are aligned so as to peak at this exact moment, as is the eighth partial shown in Figure B.2f. The eighth partial also helps to create the more local peaks at G4 in the first and third phrases.

Adding together several of these slower-moving components, we arrive at the contour shown in Figure B.2g, which tracks the motion of the melody fairly faithfully, albeit a little too smoothly[4]. In order to achieve the flat pitch plateaus that our western ears have come to expect, we need more rapid fluctuations like the 20^{th} partial (Figure B.2h) to help flatten out the peaks of the slower sine waves (Figure B.2i). As with the trumpet waveform, by including enough partials we can reproduce the original melodic contour with perfect fidelity.

B.3 Fourier Analysis of Other Musical Parameters

As described in Section 4.5, I have explored the use of this technique on parameters other than pitch. Dynamic contours, for instance, can be analyzed at various time scales. In composing *Anamnesis*, I performed a Fourier analysis of the variation in RMS amplitude in a performance of the *Allegretto* from Beethoven's 7^{th} Symphony. Smaller

[4]Pop Goes the Weasel is seldom performed as a continuous glissando.

time-scales might also be interesting; for example, the dynamic contour of speech could be analyzed, remapping the individual partials to swells in various instruments within an ensemble.

Other possibilities might involve the analysis of rhythm or tempo. In *Adagio Cantabile*, in addition to analyzing the pitch contour of a melody, I also performed Fourier analysis on its rhythm, treating each note duration as a sample. More generally, any tempo curve could be analyzed, with rhythm perhaps treated as an extremely fine-grained tempo fluctuation.

The truth is that any continuously varying parameter could be subjected to this process—the key is to find a way of precisely quantifying it. For example, given a program that returns a numerical measure of sensory dissonance given a short segment of audio, one could map out the rise and fall of sensory dissonance over the course of a recording, and then perform Fourier analysis on that contour.

B.4 A Mathematical Schenkerian Analysis?

Those familiar with the theories of Heinrich Schenker may note a certain kinship between Schenkerian analysis and the ideas presented above. After all, both reflect the multiscale nature of musical structure. When we use Fourier analysis to investigate the pitch contour of a melody, the lower partials that result are responsible for the overall shape of melody, while the higher partials are responsible the more surface-level ornamentation. These are analogous to the background and foreground levels of a Schenkerian analysis, respectively.

This may be a valuable analytical tool, one whose objectivity is complementary to the more subjective process of Schenkerian analysis. It can also be used on parameters other than pitch, which makes it potentially applicable to a wide range of musical styles.

B.5 Creative Applications

The use of this technique for analysis, in and of itself, has some potentially interesting creative applications. By its very nature, Fourier analysis is a process of fission: one complex motion is being split into many simpler motions. This suggests a transformation from monophony to polyphony. I explore this possibility in *Unraveled*, where a melody is accompanied by swells based on a Fourier analysis of its pitch contour. Likewise, in *Anamnesis*, a single loudness contour gives rise to a highly polyphonic texture of overlapping crescendi and diminuendi.

Other creative possibilities arise from spectral modification and subsequent resynthesis. A simple example might be to remove the upper or lower partials before resynthesis, resulting in a high- or low-pass-filtered version of the original contour (something that I employ in both *Adagio Cantabile* and *Unraveled*). However, there are many other options: what if, for example, we were to remove only the even numbered partials?

One of the most interesting possibilities, which I have yet to explore in any of my work, is to modify only the phase of the spectrum before resynthesis. In this way, the same basic frequencies would be present, in the same proportions, but now peaking at different times. Compared with amplitude, I suspect that phase is equally, if not more, important to the character of a musical gesture.

There is a kind of alchemy involved in the creation of a good melody. Why is it that in some contexts a high point fills the listener with exultation, whereas in other contexts it leaves us relatively cold? I think that a partial answer has to do with the richness of the "melodic contour spectrum": when several interlocking cycles, peaking at different times, finally rise up together at the same moment, there is something in us that responds.

Bibliography

[1] S. Johnson, *Interface Culture: How New Technology Transforms the Way We Create and Communicate*. Basic Books, Inc., New York, NY, USA, 1999.

[2] M. Evanstein, *SCAMP: a Suite for Computer-Assisted Music in Python*. GitHub, 2018. `https://github.com/MarcTheSpark/scamp`.

[3] R. Benavidez, "Piñatas of Earthly Delights." `http://robertobenavidez.com/bosch`.

[4] M. Behrmann, E. Ec, D. Authors, and M. Kinas Jerome, *Assistive Technology for Students with Mild Disabilities: Update 2002*, .

[5] B. Verplank, *Interaction Design Sketch-book*, 2009. `http://www.billverplank.com/IxDSketchBook.pdf`.

[6] G. Lakoff and M. Johnson, *Metaphors we live by*. University of Chicago Press, Chicago, 2003.

[7] A. P. McPherson and Y. E. Kim, *Piano Technique as a Case Study in Expressive Gestural Interaction*, in *Music and Human-Computer Interaction* (S. Holland, K. Wilkie, P. Mulholland, and A. Seago, eds.), Springer Series on Cultural Computing, pp. 123–138. Springer London, London, 2013.

[8] C. Krahé, U. Hahn, and K. Whitney, *Is seeing (musical) believing? The eye versus the ear in emotional responses to music*, *Psychology of Music* **43** (Jan., 2015) 140–148.

[9] A. Hunt, *Radical User Interfaces for Real-time musical control*. PhD thesis, University of York, UK, 1999.

[10] S. Jordà, *FMOL: Toward User-Friendly, Sophisticated New Musical Instruments*, *Computer Music Journal* **26** (Sept., 2002) 23–39.

[11] M. Kimura, *Performance Practice in Computer Music*, *Computer Music Journal* **19** (1995), no. 1 64.

[12] B. Pennycook, *Computer-Music Interfaces: A Survey*, *ACM Comput. Surv.* **17** (June, 1985) 267–289.

[13] J. Savage, *New Models for Creative Practice with Music Technologies*, 2011. `www.jsavage.org.uk/jsorg/wp-content/uploads/2011/03/new_models.pdf`.

[14] J. Endersby, *Lumpers and Splitters: Darwin, Hooker, and the Search for Order*, *Science* **326** (Dec., 2009) 1496–1499.

[15] G. E. Lewis, *Too Many Notes: Computers, Complexity and Culture in Voyager*, *Leonardo Music Journal* **10** (Dec., 2000) 33–39.

[16] G. Nottebohm, *Two Beethoven sketchbooks : a description with musical extracts*. Gollancz, London, 1979.

[17] B. R. Levy, *Metamorphosis in Music: The Compositions of György Ligeti in the 1950s and 1960s*. Oxford University Press, 2017.

[18] J. P. Zagal, S. Björk, and C. Lewis, *Dark Patterns in the Design of Games*, .

[19] C. M. Gray, Y. Kou, B. Battles, J. Hoggatt, and A. L. Toombs, *The Dark (Patterns) Side of UX Design*, in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*, (Montreal QC, Canada), pp. 1–14, ACM Press, 2018.

[20] B. Scott and T. Neil, *Designing Web Interfaces: Principles and Patterns for Rich Interactions*. "O'Reilly Media, Inc.", Jan., 2009.

[21] C. M. Joseph, *Stravinsky Inside Out*. Yale University Press, Oct., 2008.

[22] K. Ueno, *E-mail Interview*, Nov., 2018.

[23] J. Kapuscinski, *Personal Interview*, Jan., 2019.

[24] "VCV - Audible Instruments." `https://vcvrack.com/AudibleInstruments.html`.

[25] C. Bartlette, D. Headlam, M. Bocko, and G. Velikic, *Effect of Network Latency on Interactive Musical Performance*, *Music Perception: An Interdisciplinary Journal* **24** (2006), no. 1 49–62.

[26] C. Runciman and H. Thimbleby, *Equal opportunity interactive systems*, *International Journal of Man-machine Studies* (1986).

[27] D. Stowell and A. McLean, *Live Music-Making: A Rich Open Task Requires a Rich Open Interface*, in *Music and Human-Computer Interaction* (S. Holland, K. Wilkie, P. Mulholland, and A. Seago, eds.), Springer Series on Cultural Computing, pp. 139–152. Springer London, London, 2013.

[28] G. Solis and B. Nettl, *Musical Improvisation: Art, Education, and Society*. University of Illinois Press, 2009.

[29] L. Berio, *Remembering the Future (based on lectures delivered in 1993 and 1994)*. Harvard University Press., Cambridge, MA, 2006.

[30] C. Roads, *Composing Electronic Music: A New Aesthetic*. Oxford University Press, June, 2015.

[31] W. W. Zachary, *An Information Flow Model for Conflict and Fission in Small Groups*, *Journal of Anthropological Research* **33** (Dec., 1977) 452–473.

[32] C. Barlow, *On Musiquantics: Von der Musiquantenlehre translated*. Royal Conservatory The Hague, 2012.

[33] C. Roads, *Microsound*. MIT Press, 2004.

[34] L. Inge, "9 Beet Stretch." `http://www.9beetstretch.com`.

[35] P. Griffiths, *Modern Music and After*. Oxford University Press, New York, NY, USA, Feb., 2011.