Journal of Music, Technology & Education
Volume 9 Number 1
 doi: 10.1386/jmte.9.1.17_1

DAVID OGBORN
McMaster University

Live coding together: Three potentials of collective live coding

ABSTRACT

Informed by the author's experience directing the live coding Cybernetic Orchestra at McMaster University, this article discusses three areas in which collective live coding can make strong contributions to music education: the reconnection of studio labour to public musical performance, the exploration of new forms of musical co-presence, and the comprehension of music as positioned within a wider digital media context.

KEYWORDS

live coding
laptop orchestras
acousmatic music
musical co-presence
projectional editing
remediation

INTRODUCTION: MUSIC, CODE AND CO-PERFORMANCE

The sociologist Alfred Schütz, writing at the beginning of the 1950s, assigned music a special relationship to the social. For Schütz, listening to music – and even more so making music together – constituted the 'mutual tuning-in relationship, the experience of the "We"', which is at the foundation of all possible communication' (1951: 173). This fundamental role for music was primarily due to music's demand to be experienced as a step by step unfolding in time, 'polythetically'. It is not possible to *understand* music without paying attention, in some way, to *when* its constituent elements happen.

Software, like music, also involves a process that is unfolded in time: the algorithm. An algorithm is notated, in code, and then performed by the computer. The notated code of software is often concealed in some way from the users of software, who we might think of as software's audience, just as

the fixed musical score or driving structures of an improvisation are often not directly available to the audience at a musical event. Recent history provides us with two significant exceptions to this concealment: the free and open-source software movement has, since roughly the 1980s, insisted on the public sharing of source code (Kelty 2008); and the live coding movement has championed the public performance of the act of writing and modifying code, principally but not exclusively to produce artistic results, and often but not always in front of a live audience (Magnusson 2014; Collins et al. 2003).

Music and software thus both offer a situation where we negotiate the unfolding of a web of relationships in time. This goes a long way in explaining why so much live coding is directed, precisely, at music and sound (as opposed to say, visual or sculptural art). In both the performance of music, and the computer's performance of the algorithm, there is a critical surplus or difference between the specification and the execution. To understand the music, we *must* play it and hear it. To understand the algorithm, we *must* run it. In both cases, the translation from plan to temporal experience is far from trivial (Chun 2011).

Schütz insisted that, if all music drives at a fundamental mutual tuning-in relationship, it is the co-performance of music that exemplifies this the most, by involving so many forms of this tuning-in. As he wrote,

> The coperformers … have to execute activities gearing into the outer world and thus occurring in a spatialized outer time. Consequently, each co-performer's action is oriented not only by the composer's thought and his relationship to the audience but also reciprocally by the experiences in inner and outer time of his fellow performer.
>
> (Schütz 1951: 175)

Until recently, such co-performance has been the exception rather than the rule in electronic music cultures, or has been erased or concealed or downplayed when it did exist. Instruments like MIDI synthesizers and electric guitars have typically and famously been used in ensembles, but these are instruments that are largely extensions of 'traditional' instruments, and are most often employed as direct replacements for them. The recent proliferation of laptop orchestras (and similar ensembles) represents a departure from the rather solitary traditions of electronic or computer music made in studios with instruments that are no longer primarily localized vibrating objects. Some of these ensembles, such as the Cybernetic Orchestra directed by the present author, have a strong relationship to live coding in particular, while all of them have a strong relationship to sound and music programming in general.

Collective live coding, such as that which is possible in a laptop orchestra, offers a number of significant, potential contributions to music education. Following a brief review of laptop orchestra as a field and the evolution of the Cybernetic Orchestra, this article discusses three such potentials: the reconnection of studio labour to public musical performance, the exploration of new forms of musical intersubjectivity, and the comprehension of music as positioned within a wider digital media context.

THE CYBERNETIC ORCHESTRA

In the past decade, ensembles of laptop musicians have become increasingly common. These groups are as diverse in their intent and results as they are numerous. For our present purposes, laptop ensemble will be understood

as a general category for all ensembles that place more than one laptop and laptop musician together in a performance setting. Some of these ensembles choose to call themselves 'laptop orchestras' while others do not. One sometimes encounters the idea that groups calling themselves laptop orchestras are necessarily restricted to a model of performers following instructions from a composer, but this is contradicted by numerous examples, including the Cybernetic Orchestra, as well as a number of early ensembles who used the term orchestra while emphasizing collaborative improvisation (Knotts and Collins 2014). In any case, whether the group is small or large, and whether it calls itself an ensemble, an orchestra or something else, many of the same potentials and challenges for music education will be found.

Within these broad parameters, there are groups with a fixed number of members in performance, such as the Princeton Laptop Orchestra (Trueman et al. 2006), alongside groups with fluctuating numbers of members, and groups where the membership is not only fixed in number but individual identity also (like Queen, who could not be Queen without Freddie Mercury). There are groups whose identity is tied closely to a particular technical platform or programming language, such as the Linux Laptop Orchestra (Bukvic et al. 2010) or Powerbooks Unplugged (Rohrhuber et al. 2007), whose pioneering experiments have been closely tied to SuperCollider and JITLib. Conversely, there are groups whose identity eschews particularity. The Concordia Laptop Orchestra (CLOrk) is conceived in terms of transformative education and action research, with its direction continuously subject to group input and modification (Tsabary 2014).

There are numerous and active links between the live coding and laptop orchestra fields. The influence of collaborative music-making is evident even in settings where virtuosic solo live coding comes to the forefront. If one attends closely to the development of live coding systems, one will most often find duos, trios or at least co-developed systems, not too far away in the network that produces the 'solo' artist. And while not all laptop ensembles/orchestras do live coding, many have at least some experience with the medium, and most, if not all, have coding and development activities as key features of their environment. Rehearsals of the Princeton Laptop Orchestra include direct programming instruction (Wang et al. 2008), for example, and the necessity of incorporating troubleshooting (which involves a kind of computational thinking even when it is not expressed as code) into rehearsals is a direct consequence of the fungible, reconfigurable nature of the laptop as a device (Booth and Gurevich 2012). Play-based learning is an effective way of scaffolding this computational learning, especially for those coming without substantial programming experience, and live coding can introduce structures of play without any specific game-like structures (Hancock 2014).

The present reflections have been most directly nourished by the experience of founding and directing, since early 2010, a larger ensemble that very directly marries collective electronic music-making to live coding: the Cybernetic Orchestra at McMaster University. The Cybernetic Orchestra was formed under the influence of Henry Jenkins' ideas about participatory culture, as a setting in which learning takes place in an informal, peer-to-peer way, closely related to the growth of Internet-mediated sharing (Jenkins et al. 2006). The Cybernetic Orchestra has never been tied to a specific University course or program, but was rather formed as a research project supported by Canada's Social Sciences and Humanities Research Council as well as the University's Arts Research Board. Members have

included undergraduate and graduate students from diverse programs, alumni, staff and faculty. Live coding was not a central feature of the orchestra's conception, but quickly and organically, it became the orchestra's defining preoccupation. This often involved improvisation, but has sometimes also led to 'live coding compositions' that have proven especially helpful in the orientation of new members (Ogborn 2012a). As members join and depart the orchestra in continuous waves, the preservation of experience from cohort to cohort is a key challenge, and one shared with traditional music education contexts (Reid and Duke 2015). At the moment, the ensemble consists of ten to twelve people, and excepting brief periods where the number of members has risen to the mid-20s, ten to twelve people has certainly been a very typical size for the ensemble over time.

A key pedagogical strength of the laptop ensemble is the fluidity of the roles its participants take on from moment to moment, especially when it is a case of algorithmic performance (as in live coding). The member of a collective algorithmic ensemble might be a composer, an instrument designer, a listener or some other role in quick succession, with other members of the ensemble 'covering' complementary roles (Brown and Dillon 2007). This fluidity of roles is augmented by a second fluidity – the evolution of an ensemble's performance practice over time. Without a strong, traditional model for how they are supposed to work, laptop orchestras are free to continuously redefine themselves. In its earlier years (2010 to early 2013), the Cybernetic Orchestra's live coding used the ChucK language (Wang and Cook 2003) exclusively, in tandem with an application that was developed for fine-grained time synchronization in laptop orchestras, EspGrid (Ogborn 2012b). A number of performance practices developed on this basis, including the direct and repeated execution of small pieces of code, sometimes as brief generative events (rather than completed generative architectures), the idea of function calls as instruments (and as easy starting points for newcomers), and games involving sound or responsibility passing around the group in circles (Ogborn 2014). The ensemble then transitioned to using SuperCollider (McCartney 1998), and then transitioned again to using both SuperCollider and Tidal (McLean 2014) in alternation, becoming a bilingual orchestra (with *Bilingual* becoming the title of the group's third and most recent album release).

When the Cybernetic Orchestra was launched, part of the research agenda was to develop answers to the classic question from the field of New Interfaces for Musical Expression (NIME): how can music technologies have a low entry fee without imposing any ceiling on virtuosity (Wessel and Wright 2002)? In effect, the answer that has emerged through the activity of the orchestra has been to encourage and allow the differentiation of the group's activities, and, in particular, the encouragement of the literal differentiation or individuation of members from the group. If on the one end of a spectrum there were simple live coding compositions were members would execute similar function calls according to a composer's intention, on the other end of the spectrum there came to be spaces for solo live coding in a potpourri of languages (somewhat broader than the languages used in the orchestra as a whole). For example, in late summer 2013, members of the orchestra provided the entire roster of solo and small group artists for the first Canadian algoraves (Collins and McLean 2014) – dance nights fuelled by algorithmic music – in two back-to-back nights in Hamilton and then in Toronto. Alongside the development of soloistic roles, the recent years of the orchestra's activity have also seen

practices where the entire group works on a shared piece of code, a practice that I will come back to later in this article.

To summarize the Cybernetic Orchestra's evolution, at the beginning we used only one language and were frequently doing similar things with independent data and machines. As things have progressed, our range of options has widened, to encompass everyone working together on the same data and machine, and to not working together at all, but rather allowing the individual to develop their virtuosity, with the orchestra functioning as their audience, foil, support network or sparring partner.

THE ACOUSMATIC PERFORMER

The 'inherent sociality' of live coding has important implications for music education, particularly with respect to the role of electronic means. By and large, music education paradigms of the early twenty-first century have scarcely any incorporation of the electronic music developments of the middle of the twentieth century. Take the experiences and insights around Pierre Schaeffer. Schaeffer, on the basis of his early experiments with *musique concrète*, proposed that acousmatic listening situations would become of ever increasing importance. Half a century later, acousmatic practices are indeed more prevalent than ever and in numerous forms, ranging from the ubiquity of private listening devices to the easy embrace, by multiple music traditions (popular and 'unpopular'), of sound transformations and samples as a fundamental feature of the medium. And yet, Schaeffer is relegated to the margins of music education, if taught at all, with exceptions typically connected to strong electroacoustic studies or music technology programs.

It might be tempting to blame this lacuna on a presumed 'conservatism' on the part of educators. However, I believe the fault line in this discourse is more fundamental and specific than that, and that it has to do with the way that electronic music traditions tangle or fold the role of the performer. Electronic music traditions for a good part of the twentieth century have often been bound to solitary studio practices, which, at first glance, might even appear to diminish or eliminate the role of the performer completely. Nonetheless, the various acousmatic milieus of the past century all contain social performance elements. In acousmatic music, pieces were often diffused, a situation involving a kind of performer (although many from that tradition insisted, symptomatically enough of my argument here, that this live diffusion did not constitute performance) lots of collective work and camaraderie around moving speakers, and periodic festivals involving tight-knit communities of artists and enthusiasts (Harrison 1999). The genres of house, techno, hip hop, Jamaican sound system, all produce the figure of the DJ as a performer, who brings the recorded music 'to life' with their performance interventions over it, whether these interventions are subtle or drastic, and with the tangible presence of large loudspeaker systems. The culture of home studios, emerging in the 1980s, had its magazines, gear swaps and other meeting places. Closer to our own time, a thriving participatory culture produces pieces of music, created on individual laptops in random places, that are then shared with a broad, international public via a range of music-sharing platforms.

Alfred Schütz insists that 'the social relationship between performer and listener is founded upon the common experience of living simultaneously in several dimensions of time', referring to their common experience both of the sound events relative to clock time and to the various layers of

relationships in time behind those outer events. Schütz' description of the performer applies quite handily to the diffuser and the DJ, who like the classical performer cultivate a close knowledge of the 'text' of the music that they 'gear into the outer world' (1951: 175). At the same time as these quasi-acousmatic situations produce a 'performer' role, they tend to conceal the role of the studio or computer work in producing the sonic result. We watch the turntablist scratch, but not the labour that produced the underlying record. We can choose to see the acousmatic artist project their work spatially by skilful manipulation of the faders, but we generally have no access to the protracted process of transforming and assembling sounds that produced the composition. Consistent with this concealment, the principal electroacoustic analytical traditions are explicitly based on the perspective of the listener rather than the perspective of production in the studio (Moore et al. 2013). Audience members without intimate knowledge of the concealed labour that preceded these performance situations could easily be misled into thinking that more of the sonic result is due to decisions being made then and there than is actually the case.

These considerations give a special significance to the live coding practice of projecting one's screen for the benefit of the audience. While not all live coding performances involve screen sharing, most do. The part of the TOPLAP manifesto that sustains the most discussion is probably the manifesto's demand to 'show us your screens' (Ward et al. 2004), which can readily be related to fundamental features of post-Linux early twenty-first-century code culture, with its strong tendency to share source code, make development processes publicly visible, and its recognition that the roles of participants change – yesterday's casual user becomes tomorrow's project advocate, becomes next year's committed developer (Coleman 2012). The code that is projected is not only a surfacing of work on code but also, and simultaneously, a surfacing of work on sound and audio that in other circumstances might have been performed in the isolation of the studio.

We thus arrive at a first key potential of live coding for music education; by encouraging the exposure of labour on sound materials, it will become easier to relate the various acousmatic traditions to traditions of music education centred on the visible and tangible labour of the performer. Musical learning in contemporary popular and dance traditions is overwhelmingly a matter of peer learning across a range of informal settings (Lebler 2007), and the laptop ensemble offers a setting in which this learning can be scaffolded in the form of cognitive apprenticeships (Thompson 2012). At the same time, musical skills gained in the laptop ensemble, such as improvisation, can transfer back to more traditional music performance settings (Albert 2012). Electronic music programs and initiatives that adopt collective live coding as part of their activities stand to gain new common ground upon which to have a conversation with their 'more traditional' counterparts.

Live coding's exposure of the work of the studio also reverses another historical erasure – the obscuring of the close relationship between programming and electronic music. Schaeffer's earliest experiments on *musique concrète* were accompanied by computational fantasies. In his April 1948 journals, he imagines a '*most general musical instrument possible*' (Schaeffer 2012: 7, original emphasis) while several months later, he laments having to wait for a 'huge cybernetic-like machine that can achieve millions of combinations' (Schaeffer 2012: 18). Modular analogue synthesis, with its patch cables and racks of modules, replicates the physical structure with which some early

computers were programmed, and this structure recurs as the principal metaphor of various graphical patching environments, such as Max and Pure Data. Consider that the construction of a chain of plug-ins in a digital audio workstation (DAW) is in fact a kind of programming, analogous to creating a chain of dependent cells in a spreadsheet, the most common form of End-User Software Engineering (Burnett et al. 2004). In the electronic music of the early twenty-first century, programming is always just around the corner. Even when an artist or ensemble simply relies on unmodified presets in standard software, these presets and software themselves are the result of the ongoing programming activities of others.

PROJECTIONAL INTERSUBJECTIVITY

The projected code of live coding not only exposes labour on code and sound – it also establishes a situation of intersubjectivity in which people become keenly aware of what others are doing. Musical intersubjectivity is generally fragile, and is even more fragile when it is built upon electronic and computer instruments with less traditional educational and cultural scaffolding (Lagerlöf et al. 2014). A second potential of live coding for music education addresses this precisely. By allowing the actions of performers to be sensed in multiple ways, live coding encourages the development of new forms of musical co-presence (along with the development of a new perspective on traditional forms of musical co-presence) and thus new forms of musical intersubjectivity.

In a 'traditional' musical ensemble, the visual and tangible presence of the moving bodies and instruments, and the relationship between the faces and limbs of the players especially, helps create a robust co-presence. With laptop music, the face staring at the screen is often not enough to establish this co-presence. The rudimentary projection of the code, in live coding, goes a long way towards facilitating co-presence between a solo performer and an audience. However, when it is a matter of an electronic ensemble, as in the laptop orchestra, the rudimentary projection of the code becomes a less simple matter (Wilson et al. 2014). One could provide each of the members of an ensemble with their own projector and projection surface, but this runs up against three limits. First, projectors can be expensive. Second, projection surfaces are typically an even more finite resource as they involve space and architecture. Finally and perhaps most fundamentally, the projectors and projection surfaces are only part of a communication system that includes the attention and perception of situated, embodied humans, and as the number of ensemble members and performance actions grow, simple projection begins to present both the audience and the performers with challenges of attention and focus.

A basic and universal way in which laptop ensembles address the increased challenges of co-presence and intersubjectivity is through an enormous amount of dialogue and discussion. Laptop ensemble rehearsals include far more talking than traditional musical rehearsals do, and they need to include this talking. At London's Kingston University Digital Arts Collective, the presence of not one but several faculty members in the collective was found to be especially helpful in facilitating this discursive, reflective aspect (Ben-Tal and Salazar 2014).

A live coding-specific response to the challenge of co-presence can be found in the practice of *roulette*. In a roulette, any number of performers

take turns adding and modifying code as it runs on a single machine that can be connected to a single projector (Guzdial 2013). The single projector and surface provides a single point of focus for both the audience and all the performers, and creates a strong sense of co-presence. The roulette can also be understood as an example of the highest, 'synergistic' level of the taxonomy of collaborative e-learning, at which 'contributions … are fully meshed into [a] collective final product' (Salmons 2008: 218). The above-mentioned fluidity of roles in laptop ensembles is systematized in the roulette, as members move quickly between different types of listening, forming compositional intentions, programming, correcting errors, etc.

The practice of roulette highlights the way in which projected code is both the expression of an intention and a kind of trace or echo of that intention. When a live coding performer comes to their turn in the roulette, they are called upon to intervene in the music as it sounds, and with the code and the state of the machine as they are, but they are also called upon to interpret and react to the presence of the group (who will inherit the results of their action) and to the intention of the group as manifested in their memory (both of the sound and the code). What follows is one concrete and common example of how this dynamic can play out. A performer adds some code to the roulette whereby the musical intention is clear but they make a mistake in the syntax before passing the turn on to the next performer. The next performer can now decide whether and how to correct the mistake, and as they make this decision the code is present as a kind of clue or reminder about what is happening, and about what has happened.

Roulettes can quickly arrive at points where it is not only challenging to keep track of the musical evolution of the piece, but also difficult to keep track of what is going on 'in the code'. Indeed, in many live coding environments the code is fundamentally ambiguous in any given moment, as it can be unclear whether it has yet to be executed, will ever be executed, when it was executed or if it was executed multiply in the past. It is possible to adopt a style of coding that sidesteps this ambiguity, by making the displayed code a complete diary of the operations performed through code on the machine. In this situation, the code plus audio combination can be contrasted with the traditional musical co-performance described by Schütz and the fixed acousmatic composition. In diaristic live coding, the sound events in outer time are indexed to some or other extent by a visual experience that tends to function as a gradually vanishing trace of the artist's intention. When the mind compares present events with past events, the projection is there as a kind of amanuensis. This may be likened to reading a score while listening, except that unlike the score, the code functions at multiple levels of abstraction simultaneously (low level DSP code, patterns, other higher level algorithms). I would contend that such diaristic code comes closer than the traditional score to externalizing the multiple simultaneous dimensions of time to which Schütz refers.

The Cybernetic Orchestra has recently been using the *extramuros* software to live code (in either Tidal or SuperCollider) into shared text buffers that are accessed via a web browser. Like the roulette, such 'shared buffers' provide a unified visual focus for both the ensemble and the audience – everyone can see everyone else' code, on the same screen – but unlike the traditional roulette there is no need to take turns. *extramuros* has recently been adapted to so that any event anywhere in a networked ensemble (potentially distributed around the globe) becomes a call to a JavaScript stub function in each web browser. This supports the rapid 'proposal' and iteration of visualizations of the activity

of an ensemble, and even the possibility of live coding the visualization of live coding (Ogborn et al. 2015).

Projectional editing is an area of live coding research that has enormous application to the problem of co-presence in electronic music. While straightforward text editors have been the dominant environments for programming, and thus also very visible in live coding, it is possible to express and modify algorithms through custom interfaces that are tailored to some combination of the structure of the target domain, the structure of the programming language/elements, and the nature of the cognitive demands placed on the programmer/performer. Projectional editors project an abstract representation of the structure of the algorithm into specific, visual and editable projections. A number of recent live coding projects have featured such projectional editing. The drone-music focused Threnoscope includes a graphical projection of the way that synthesized drones move around a spatial array of loudspeakers, alongside a conventional text-based editing window for SuperCollider code connected to the drones (Magnusson 2014). Texture is a graphical editor for the Tidal language, visualizing both the specification of the program as a set of elements connected by proximity in space and the result of the program, in the form of events flowing from one element to the next (McLean and Wiggins 2011).

In the context of electronic ensemble performance, such projectional editing could be handily combined with the capture, analysis and visualization of other aspects of the performance setting, including but not limited to the physical motion of live coding performers. The goal of such a combination would be to support the awareness of the members of an ensemble of everything that is happening in that ensemble. In a broader sense, projectional editing in live coding ensembles suggests the possibility of reconceiving performance events as multidimensional sets of data that can be captured and then navigated through multiple projections, both in the moment of creation and as an archive. What kinds of musical awareness will become possible when electronic and traditional musicians can work together in settings where they not only have access to rich data about each other's actions and dispositions, but where they can return to that data in subsequent moments for discussion and reflection?

MUSIC, METAPHORS AND REMEDIATION

As the director of the Cybernetic Orchestra, I have seen this trajectory numerous time; at first when someone joins a live coding laptop orchestra, their focus is on trying to achieve a certain musical result. With time their concern becomes wider, to include concern for the elegance, or 'economy' of the code, for discovering newer ways of doing things, for playing with the way in which the code is visually projected and so on. This may be understood as a process of substitution in which a participant's previous metaphor for the computer as a music-making device is replaced or augmented by a new metaphor, with such moments of metaphor formation as fundamental to musical learning (Brown 1997).

Audio programming provides a rich, experiential setting for exploration and play with new metaphors. Indeed, software itself can be seen as a metaphor for metaphor (Chun 2011). While the trope that 'anything is possible with programming' is without a doubt an exaggeration, it is true that programming interfaces represent especially pliable surfaces on which materials

can be refashioned and reconceived in quite drastic ways. It can happen that a project develops a certain amount of code on the basis of a specific metaphorical intention, such as 'to create my own step sequencer', only for some local feature of that effort, such as a set of dynamic synthesis textures created for that step sequencer, to then become an emergent focus of future work and interest, leading to the abandonment of the original intention and metaphor. This is particularly the case when the process is conceived and executed as 'bricolage programming'; tinkering with readily available materials and making manoeuvres in the software that are guided by their momentary intuition rather than some preconceived architectural result (McLean and Wiggins 2010).

This third potential of live coding for music education is closely related to the way in which bricolage programming encourages the development of new metaphors. Live coding provides a context in which music is apprehended not as a self-sufficient monad but rather as an element of a larger, fluid cultural context. Such a contextual, cultural understanding of the limits of any particular metaphor for how music 'should' work should be fundamental to any twenty-first-century conception of music pedagogy, given the many diverse ways in twenty-first-century musical activity proliferates and circulates (Brown 2012). This is not a matter of simply accepting in principle that there are multiple understandings of music, but rather a matter of actively cultivating the ability to go to those understandings, develop them and form new connections between them, aided by the extreme pliability of software.

Ultimately, live coding can be conceived not simply as a way to make music, but rather as a paradigmatic form of new media that remediates musical traditions (Bolter and Grusin 1998). If live coding were simply another one of many ways to make music, then the question 'how can this musical result be produced through live coding?' might not be so important. If, on the other hand, music is a remediated element of a new, live coding form, then the more pertinent question will be how musical expectations and experiences are changed, and how they have a different meaning, on the basis of this new context? This wider digital media context can also be thought of as the provision of lines of flight, points of entry into other worlds. The traditionally educated musician that finds themselves in a live coding laptop orchestra will find paths that lead to forms of activity that might not have been expected. Even if, for example, programming is an expected and clearly sign-posted activity of the orchestra in question, this activity may lead into some kind of engagement with hacking and free and open-source software. And even if a hypothetical ensemble is clearly oriented towards making acousmatic music or sound art, a new participant may not expect this to lead towards a potential engagement with the wider media arts, generative art, net art, etc. And yet, frequently it does. An ensemble may also be a gateway into involvement with online sound sharing communities, themselves instances of and potential gateways to a wider universe of online art platforms. At the most general level, these lines of flight become generic but important 'transferrable skills', which can be heightened by explicit journaling and reflection activities (Mudd 2012).

At the same time as live coding connects musical activity to other contexts, it does this without simply making music a pliant servant of 'non-musical' aims – this is more than just program music (the unfortunate pun notwithstanding). Live coding offers a setting in which musical considerations frequently operate in a tension with another set of considerations that are

not simply functional demands. The same is also true in the other direction; music in performance offers a programming situation where 'requirements' are fluid or perhaps not even usefully conceived of as requirements at all. Research has begun to demonstrate the beneficial effect of live coding (in the sense of live demonstrations of programming activities) on computer science pedagogy (Rubin 2013; Paxton 2002), while other research has demonstrated that exclusive behaviours and attitudinal barriers represent significant obstacles to broad, deep education about computing (Margolis and Fisher 2008). The ambiguity of the collective live coding ensemble (Is it about music? Is it about programming? Is it about social activity?) can be a distinct advantage in recruitment as it potentially offers multiple points of engagement. A 'barrier' represented by one of the ensemble's aspects may not be as severe as it would be if that were the only feature of the activity.

CHALLENGES AND OPPORTUNITIES

Collective live coding – aka the live coding laptop orchestra – can be an ambassador – or perhaps a diplomatic corps would be a better metaphor – between alienated areas of human activity. It enables a type of interdisciplinary conversation that was more difficult to have previously, when electronic music was mostly made by individuals working in a mostly invisible studio environment, where the traces of their process were, as a rule, ephemeral and quickly lost to public examination. It establishes a situation of fluid co-presence, where people are aware of their own and other's actions in heightened ways, with the possibility that the evolution of the field will expand this potential for musical intersubjectivity further in the years to come. And collective live coding brings together activities and traditions that have hitherto been separate, supporting the formation of musicians who are not 'only' musicians but rather are keenly aware of music's participation in broader cultural contexts.

The discussion above has highlighted some of the exciting educational potentials of collective live coding, clearly recognizable as a constructionist learning situation, wherein all concerned are 'consciously engaged in constructing a public entity, whether it's a sand castle on the beach or a theory of the universe' (Papert and Harel 1991). There are numerous challenges and opportunities that can be the fruitful subject of the future, 'ramified research program' that Papert and Harel call for in that seminal work. Foremost among these is the objective evaluation of the learning outcomes within these ensembles – as most work on this so far has unavoidably tended towards the anecdotal and descriptive – and more rigorous evaluation work could go hand in hand with the provisional identification of best practices. A growing body of work identifies and evaluates the best practices for teaching and learning programming to musicians, such as pair programming and reflective journalling (Moore 2014), but this tends to sideline specifically artistic learning outcomes, however those might be defined.

As a demarcated environment filled with independently creative people whose work is closely integrated with algorithmic, rule-following machines, the collective live coding environment could also represent a model case for the development of computational social creativity, which studies creativity by modelling it in virtual populations (Saunders and Bown 2015). Alongside the development of actual live coding ensembles, we can model how such ensembles might develop were we to make specific sets of assumptions about how everything in the situation behaves. Common styles of live coding performance could even give selected results of this emerging field

a concrete presence; in the dark, watching a text editor projected onto the wall, human performers and successful virtual performers from computation models can intermingle, and the audience can be productively unaware of what is being done by humans versus what is the result of software agencies. Live coding already involves a theatrical play with the boundaries of what is generated by algorithms (i.e., consequences of prior thought) versus what is generated by whim, intuition and personality in the now. It is thus a primary site for the education of musical intelligences adapted to a twenty-first-century reality increasingly characterized by human-algorithm hybrids.

REFERENCES

Albert, J. (2012), 'Improvisation as tool and intention: Organizational practices in laptop orchestras and their effect on personal musical approaches', *Critical Studies in Improvisation*, 8: 1. http://www.criticalimprov.com/article/view/1558. Accessed 10 September 2015.

Ben-Tal, O. and Salazar, D. (2014), 'Rethinking the musical ensemble: A model for collaborative learning in higher education music technology', *Journal Of Music, Technology & Education*, 7: 3, pp. 279–94.

Bolter, J. D. and Grusin, R. (1998), *Remediation: Understanding New Media*, Cambridge: MIT Press.

Booth, G. and Gurevich, M. (2012), 'Collaborative composition and socially constructed instruments: Ensemble laptop performance through the lens of ethnography', in *Proceedings Of The International Conference On New Interfaces For Musical Expression 2012*, Ann Arbor, Michigan, 21–23 May.

Brown, A. R. (1997), 'Changing technologies, changing minds: Taking account of music technologies in the curriculum', in E. F. Gifford, A. R. Brown and A. Thomas (eds), *ASME XI National Conference Proceedings: New Sounds For A New Century*, Brisbane, Australia: Australian Society for Music Education pp. 31–35.

—— (2012), 'Musicianship in a globalized world', in A. R. Brown (ed.), *Sound Musicianship: Understanding The Crafts Of Music*, Newcastle upon Tyne: Cambridge Scholars Publishing, pp. 42–66.

Brown, A. R. and Dillon, S. C. (2007), 'Networked improvisational musical environments: Learning through on-line collaborative music making', in J. Finney and P. Burnard (eds), *Music Education With Digital Technology*, London: Continuum, pp. 96–106.

Bukvic, I., Martin, T., Standley, E. and Matthews, M. (2010), 'Introducing l2Ork: Linux laptop orchestra', in *Proceedings Of The International Conference On New Interfaces For Musical Expression 2010, 15–18 June*, Sydney, Australia, pp. 170–73.

Burnett, M., Cook, C. and Rothermel, G. (2004), 'End-user software engineering', *Communications of the ACM*, 47: 9, pp. 53–58.

Chun, W. H. K. (2011), *Programmed Visions: Software and Memory*, Cambridge: MIT Press.

Coleman, E. G. (2012), *Coding Freedom: The Ethics and Aesthetics of Hacking*, Princeton: Princeton University Press.

Collins, N. and McLean, A. (2014), 'Algorave: Live performance of algorithmic electronic dance music', in *Proceedings Of The International Conference On New Interfaces For Musical Expression 2014*, London, UK, 30 June–4 July.

Collins, N., McLean, A., Rohrhuber, J. and Ward, A. (2003), 'Live coding in laptop performance', *Organised Sound*, 8: 3, pp. 321–30.

Guzdial, M. (2013), 'Live coding, computer science, and education', in A. Blackwell, A. Mclean, J. Noble, J. Rohrhuber and J. A. Otto (eds), *Report From Dagstuhl Seminar 13382: Collaboration And Learning Through Live Coding*, Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, pp. 162–65.

Hancock, O. (2014), 'Play-based, constructionist learning of pure data: A case study', *Journal of Music, Technology & Education*, 7: 1, pp. 93–112.

Harrison, J. (1999), 'Diffusion: Theories and practices, with particular reference to the BEAST system', *EContact!*, 2: 4, http://econtact.ca/2_4/Beast.htm. Accessed 15 September 2015.

Jenkins, H., Clinton, K., Purushotma, R., Robinson, A. J. and Weigel, M. (2006), *Confronting the Challenges of Participatory Culture: Media Education for the 21st Century*, Cambridge: MIT Press.

Kelty, C. M. (2008), *Two Bits: The Cultural Significance of Free Software*, Durham: Duke University Press.

Knotts, S. and Collins, N. (2014), 'The politics of laptop ensembles: A survey of 160 laptop ensembles and their organisational structures', in *Proceedings Of The International Conference On New Interfaces For Musical Expression 2014, 30 June–4 July*, London, UK: Goldsmiths, University of London pp. 191–94.

Lagerlöf, P., Wallerstedt, C. and Pramling, N. (2014), 'Playing, new music technology and the struggle with achieving intersubjectivity', *Journal of Music, Technology & Education*, 7: 2, pp. 119–216.

Lebler, D. (2007), 'Student-as-master? Reflections on a learning innovation in popular music pedagogy', *International Journal of Music Education*, 25: 3, pp. 205–21.

Magnusson, T. (2014), 'Scoring with code: Composing with algorithmic notation', *Organised Sound*, 19: 3, pp. 268–75.

Margolis, J. and Fisher, A. (2008), *Unlocking the Clubhouse: Women in Computing*, Cambridge: MIT Press.

McCartney, J. (1998), 'Continued evolution of the SuperCollider real-time synthesis environment', in *Proceedings Of The International Computer Music Conference 1998, 1–6 October 1998,* Ann Arbor, Michigan: ICMA.

McLean, A. (2014), 'Making programming languages to dance to: Live coding with Tidal', in *Proceedings Of The 2nd ACM SIGPLAN International Workshop On Functional Art, Music, Modelling And Design, 6 September,* Gothenburg, Sweden: ACM.

McLean, A. and Wiggins, G. (2010), 'Bricolage programming in the creative arts', in *Proceedings Of 22nd Psychology Of Programming Interest Group, 21–22 February 2008,* Madrid, Spain: PPIG.

—— (2011), 'Texture: Visual notation for live coding of pattern', in *Proceedings Of The International Computer Music Conference, 31 July–5 August*, Huddersfield, UK: ICMA, pp. 621–28.

Moore, A., Moore, D., Pearse, S. and Stansbie, A. (2013), 'Tracking production strategies: Identifying compositional methods in electroacoustic music', *Journal of Music, Technology & Education*, 6: 3, pp. 323–36.

Moore, D. (2014), 'Supporting students in music technology higher education to learn computer programming', *Journal Of Music, Technology & Education*, 7: 1, pp. 75–92.

Mudd, T. (2012), 'Developing transferable skills through engagement with higher education laptop ensembles', *Journal Of Music, Technology & Education*, 5: 1, pp. 29–41.

Ogborn, D. (2012a), 'Composing for a networked, pulse-based, laptop orchestra', *Organised Sound*, 17: 1, pp. 56–61.

—— (2012b), 'EspGrid: A protocol for participatory electronic ensemble performance', in *Audio Engineering Society Convention 133,* Audio Engineering Society, San Francisco, California, 26–29 October.

—— (2014), 'Live coding in a scalable, participatory laptop orchestra', *Computer Music Journal*, 38: 1, pp. 17–30.

Ogborn, D., Tsabary, E., Jarvis, I., Cárdenas, A. and McLean, A. (2015), 'Extramuros: Making music in a browser-based, language-neutral collaborative live coding environment', in A. McLean, T. Magnusson, K. Ng, S. Knotts and J. Armitage (eds), *Proceedings Of The First International Conference On Live Coding, 13–15 July*, Leeds, UK: ICSRiM, School of Music, University of Leeds, pp. 163–69.

Papert, S. and Harel, I. (1991), *Constructionism: Research Reports and Essays 1985–1990*, Norwood, NJ: Ablex.

Paxton, J. (2002), 'Live programming as a lecture technique', *Journal of Computer Sciences in Colleges*, 18: 2, pp. 51–56.

Reid, A. and Duke, M. (2015), 'Student for student: Peer learning in music higher education', *International Journal of Music Education*, 33: 2, pp. 222–32.

Rohrhuber, J., Campo, A. de, Wieser, R., Kampen, J.-K. van, Ho, E. and Hannes, H. (2007), 'Purloined letters and distributed persons', in *Music In A Global Village: Symposium, 6–8 September,* Budapest, December.

Rubin, M. (2013), 'The effectiveness of live-coding to teach introductory programming', in *Proceedings Of ACM Special Interest Group On Computer Science Education (SIGSCE'13)*, ACM, Denver, Colorado, 6–9 March.

Salmons, J. (2008), 'Expect originality! Using taxonomies to structure assignments that support original work', in T. S. Roberts (ed.), *Student Plagiarism In An Online World: Problems And Solutions*, Hershey, Pennsylvania: IGI Global USA, Information Science Reference, pp. 208–27.

Saunders, R. and Bown, O. (2015), 'Computational social creativity', *Artificial Life*, 21: 3, pp. 366–78.

Schaeffer, P. (2012), *In search of a concrete music* (trans. J. Dack and C. North), Oakland: University of California Press.

Schütz, A. (1951), 'Making music together: A study in social relationship', *Social Research*, 18: 1, pp. 76–97.

Thompson, P. (2012), 'An empirical study into the learning practices and enculturation of DJs, turntablists, hip hop and dance music producers', *Journal of Music, Technology & Education*, 5: 1, pp. 43–58.

Trueman, D., Cook, P., Smallwood, S. and Wang, G. (2006), 'PLOrk: The Princeton Laptop Orchestra, year 1', in *Proceedings Of The International Computer Music Conference, 6–11 November*, New Orleans, Louisiana: ICMA, pp. 443–50.

Tsabary, E. (2014), 'Music education through innovation: The Concordia Laptop Orchestra as a model for transformational education', in *8th International Technology, Education And Development Conference (INTED), 10–12 March*, Valencia, Spain: IATED, pp. 657–64.

Wang, G. and Cook, P. R. (2003), 'ChucK: A concurrent, on-the-fly, audio programming language', in *Proceedings Of The International Computer Music Conference, 29 September–4 October*, Singapore: ICMA, pp. 219–26.

Wang, G., Trueman, D., Smallwood, S. and Cook, P. R. (2008), 'The laptop orchestra as classroom', *Computer Music Journal*, 32: 1, pp. 26–37.

Ward, A., Rohrhuber, J., Olofsson, F., McLean, A., Griffiths, D., Collins, N. and Alexander, A. (2004), 'Live algorithm programming and a temporary organisation for its promotion', in O. Goriunova and A. Shulgin (eds), *Read_Me – Software Art And Cultures*, Aarhus: Aarhus University Press. pp. 243–61.

Wessel, D. and Wright, M. (2002), 'Problems and prospects for intimate musical control of computers', *Computer Music Journal*, 26: 3, pp. 11–22.

Wilson, S., Lorway, N., Coull, R., Vasilakos, K. and Moyers, T. (2014), 'Free as in BEER: Some explorations into structured improvisation using networked live-coding systems', *Computer Music Journal*, 38: 1, pp. 54–64.

SUGGESTED CITATION

Ogborn, D. (2016), 'Live coding together: Three potentials of collective live coding', *Journal of Music, Technology & Education,* 9: 1, pp. 17–31, doi: 10.1386/jmte.9.1.17_1

CONTRIBUTOR DETAILS

A media producer and researcher whose pursuits span electronic music, artistic programming, games and networked collaboration, Dr David Ogborn is especially known for his activities as a live coding performer, creating music and visuals by programming in front of an audience, frequently in collaborative settings. At McMaster University in Hamilton, Canada, he teaches digital audio, code and game design in the undergraduate Multimedia and graduate New Media and Communication programs, and supervises theses and major research projects addressing these and broader digital humanities concerns. He is a member of the duo 'very long cat' (with Montréal tabla player Shawn Mativetsky) and the director of the live coding laptop orchestra, the Cybernetic Orchestra.

Contact: McMaster University, 1280 Main Street West, Hamilton, Canada, Ontario, L8S 4L8.
E-mail: ogbornd@mcmaster.ca

Choreographic Practices

ISSN 2040-5669 | Online ISSN 2042-5677
1 issues per volume | Volume 4, 2013

Aims and Scope

Choreographic Practices operates from the principle that dance embodies ideas and can be productively enlivened when considered as a mode of critical and creative discourse. The journal provides a platform for sharing choreographic practices, inquiry and debate.

Call for Papers

Choreographic Practices is an international, peer-reviewed, bi-annual journal. Contributions are invited that articulate choreography from a diverse range of perspectives. We are especially interested in receiving articles that address research-led movement practices that are interdisciplinary and experimental in nature. Selected issues will also be thematically arranged. *Choreographic Practices* publishes both conventional and alternative modes of writing, including performative and visual essays.

Editors

Vida L. Midgelow
Middlesex University
choreographicpractices@live.co.uk

Jane M. Bacon
University of Chichester
choreographicpractices@live.co.uk

Intellect is an independent academic publisher of books and journals, to view our catalogue or order our titles visit www.intellectbooks.com or E-mail: journals@intellectbooks.com. Intellect, The Mill, Parnall Road, Fishponds, Bristol, UK, BS16 3JG.