CHOR-RESPONDENT: A DYNAMIC SYSTEM IN THE MUSICAL DOMAIN

Oliver Hancock
Leeds College of Music
Quarry Hill, Leeds, LS2 7PD,
U.K.

ABSTRACT

This paper introduces an example of a live algorithm interacting with a human improviser. Improvisation is considered from the point of view of communication, and the requirement for comprehensibility is discussed with reference to acquired understandings, informal learning and the cognitive event cycle. Dynamic systems for music are examined from this perspective, and systems in the musical domain are proposed as embodiments of non-expert bottom-up musical understandings. That is: dynamic systems having primitives which are representations of sonic entities; and with rules for the interaction of those entities that deal with sonically perceptible relationships. The live algorithm chor-respondent is reported as an example of such a system. Its consonance-based functioning is explained and its performance evaluated both for idiosyncratic and innovative emergent behaviour, and for musical comprehensibility by programmers and performers.

1. INTRODUCTION

This paper is primarily focussed on human instrumental improvisation with a live algorithm. My particular interest is in dynamic systems and their ability to generate patterns and forms similar to those found in Nature as opposed to human constructs. This paper examines the improvisation process, introduces the concept of dynamic systems conceived and acting in the musical domain, and evaluates one example: chor-respondent. The benefits of this approach are considered in the specific context of live improvisation.

2. IMPROVISATION

2.1. Group improvisation as communication

Berliner [1] notes that one common metaphor ‘likens group improvisation to a conversation that players carry on among themselves’. And Stroppa [13] finds that human-computer improvisation possesses vitality ‘when some kind of communication is going on between the performer and the surrounding electronic material, between both of them and the audience’. Blackwell and Young [2] unpick this communication and identify ‘strong interactivity, autonomy, innovation, idiosyncrasy and comprehensibility’ as desirable features for such a situation.

Dynamic systems are certainly capable of generating idiosyncratic and innovative emergent behaviours; comprehensibility for human improvisers and audiences is perhaps more problematic, and certainly not always achieved. Stroppa [13], writing about the computer music community in 1999, suggests that ‘in the last period too many efforts were devoted to purely technological concerns, and that fundamental musical research was somewhat forgotten’.

How are the components of comprehension learned by humans, and how can they be embedded in a dynamic system? The conventional rules of any musical language are top-down in nature and best implemented as expert systems. Cope’s work demonstrates this for a variety of musical styles [4]. But the emergent behaviour sought from dynamic systems is the result of bottom-up interactions between (often) simple elements of a complex system. There is an untaught form of learning which does not concern itself with explicit rules: acquired knowledge.

2.2. Acquired knowledge

Hall [6] considers that improvisation is embedded in culture. He considers that cultural understanding is primarily acquired, and that this process is different to the formal study of musical theory. The process of acquisition is presented as the automatic, untaught absorption of environmental influences.

Acquired understandings are described as ‘matters that people… can’t talk about’, and Berliner [1] reports that ‘musicians discussing the background and knowledge they bring to performance comment often on how much more complex jazz is than it is possible to verbalize in an interview.’ In free improvisation explicit rules of musical language are de-emphasized, and matters of instinct and intuition, taste and sensibility assume even greater importance. Peters [10] writes of ‘coming from nothing’ to indicate a minimum of prior knowledge. Acquired information is specifically personal to each individual. It is ‘equated with the self, and its patterns are automatic and totally out of awareness’[6].

Hall links acquisition to informal learning, which is contrasted with formal and technical learning. Formal learning concerns those aspects of the world which are taken for granted; technical learning involves symbolic systems which aim to make precise statements; ‘informal learning is acquired, non-linear, cooperative and not controlled by anyone except the group and its shared internalized patterns’.
This shared ‘regard for the unwritten rules of the medium as well as the subject’ make improvisation a high context (HC) activity, where the participants understand more than just the explicitly coded content of the messages. He suggests that ‘with any HC system, the link to the audience is more binding (since there is more shared information)’, although this presumably relies on the audience sharing some of the acquired, informal cultural learning as the performers.

2.3. Acquired knowledge in communication

Sarath [12] considers communication within cognitive event cycles which draw on the improviser’s knowledge (learned and acquired). An improviser is considered to initially look inward to an internal reservoir to create a musical idea. Once this idea is actualized as a performed musical utterance the improviser shifts to considering possible implications of the utterance, and assessing which are more probable responses within the prevailing context. Ultimately a specific response is chosen which neutralizes the other possibilities, allowing the cycle to recommence with a new actualization. Sarath writes that ”“probable” … refers to instances where conception is driven by strongly conditioned patterns.’ And he also considers the audience’s participation in the communication, speculating that ‘listeners familiar with the musical language at hand likely share, at least to a degree, in implication-realization perceptions similar to those of the performers’.

3. LIVE ALGORITHMS

It may not be desirable that a computer algorithm should behave exactly like an intensively communicating human improviser. Blackwell and Young [3] insist that ‘the iterative, generative, idiosyncratic world of algorithmic organisation must be accessed’. However, Lewis [8] hints at the richness available from computer algorithms. He describes his Voyager system as having its own “sound” (in the jazz sense of a personal style). And he states that ‘the [improviser’s] emotional state may be mirrored in the computer partner.’ He even aspires to embedding cultural elements citing the ‘notion of a non-Eurocentric computer music’ which ‘might embody the assumptions and cultural markers embedded in a non-European point of view’.

4. DYNAMIC SYSTEMS IN THE MUSICAL DOMAIN

In dynamic systems research there have been many musical endeavours which utilized systems from the general literature including cellular automata, iterated equations, iterated function systems, genetic processes, and agent-based systems. But Fischman [5] asserts that ‘most mathematical constructs are neutral to musical logic’, and Miranda [9] asks: ‘Would musically biased generative algorithms as opposed to adapted non-musical processes compose good music?’

In my own practice I seek to utilize the emergent, unexpected, non-human behaviours of dynamic systems, yet also produce music which is comprehensible to human improvisers, audiences and programmers. To facilitate this I have conceived and designed dynamic systems in the musical domain. That is: dynamic systems having primitives which are representations of sonic entities (notes, grains, drone pitches, durations etc.); and with rules for the interaction of those entities that deal with sonically perceptible relationships (such as consonance, synchronicity, dynamics, stepwise/leap melodic motion). It is hoped that by embedding musical concerns at a fundamental level, the dynamic systems themselves may exhibit a comprehensible, emergent identity.

Simple, perceptible relationships of pitch and rhythm are suitable foundations for bottom-up dynamic systems rules. The octave is a musical universal, as is the concept of temporal synchronicity and separation. Such things may be incorporated into dynamic systems by constructing systems from representations of sonic entities interacting in perceptually valid ways.

As well as an increased likelihood of perceptible sonic discourse, there are other questions arising from the use of dynamic systems in the musical domain. Do they provide more intuitive or elegant control of musical outcomes? Can control parameters be reduced or linked in meaningful ways? How and where in the process of programming and exploring algorithms can a composer’s personal tastes be expressed, given the use of essentially autonomous systems? How can an improviser meaningfully interact without disrupting the characteristic system behaviours?

5. CHOR-RESPONDENT

5.1. The algorithm

This work is a real-time interactive musical algorithm which interacts with a live improvising human musician solely through audio. The algorithm listens to the performer, processes perceptually significant information, runs an algorithmic process, and synthesizes sound.

In keeping with the tripartite division proposed by Blackwell and Young [3], the patch is considered in terms of three modules: listening and analysis (P), dynamic system algorithm (Q), and sound synthesis (F). chor-respondent is implemented in Pure Data as a single patch which integrates all three components. Since live performances are usually run on laptop computers, the patch is deliberately simple to promote efficiency.
Figure 1. Architecture of chor-respondent.

5.1.1. Listening & Analysis
Listening is achieved by the fiddle object, common to Pure Data and Max MSP. This derives pitch material from the improviser’s audio which is converted to an integer MIDI note number. The patch stores the most recent pitches reported by the fiddle object in a 5-note rolling buffer. This buffer is analysed to give the range of recent pitches and their consonance. The consonance measurement in this analysis phase uses the same rules as the dynamic system.

5.1.2. Dynamic System Algorithm
The algorithm has a 10-note buffer which stores pitches that have been randomly generated and then subjected to the consonance test.
A newly generated pitch is tested against each existing pitch in both buffers using a lookup table. Intervals are measured in MIDI notes (semitones), and rectified to remove octave displacements (compound intervals). The scores for reciprocal intervals (inversions) are symmetrical. The scores were chosen intuitively to provide a clear preferencing of consonant intervals. Greater consonance produces lower scores, with unisons (rectified octaves) scoring a perfect 0. 

Table 1 shows intervals in MIDI notes on the top row; the same intervals in conventional music abbreviations in the second row, and the associated consonance scores in the bottom row.

<table>
<thead>
<tr>
<th>Interval</th>
<th>m2</th>
<th>M2</th>
<th>m3</th>
<th>M3</th>
<th>P4</th>
<th>A4</th>
<th>P5</th>
<th>m6</th>
<th>M6</th>
<th>m7</th>
<th>M7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10</td>
<td>7</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 1. Consonance scores.

A newly generated pitch whose total consonance score against all buffer pitches falls below a threshold value is added to the buffer and played. Each pitch enters with its associated consonance score. Over time each pitch’s score is degraded and eventually even the most consonant pitches are eliminated. Control of the overall degree of consonance is managed by using the performer’s consonance value as the threshold value for passing the consonance test.

5.1.3. Sound Synthesis
The synthesis engine is rudimentary, using a sine source and a linear attack-decay envelope. Volume levels are determined by consonance scores, so that the more consonant pitches are louder and remain prominent for longer than the less consonant pitches. This also reflects the time they take to degrade within the algorithm’s buffer, which is mirrors the time during which they participate in the consonance assessment.

5.2. Emergent Behaviour
The algorithm exhibits emergent behaviours: some that were intended in its design; and others that were not initially obvious.
In the absence of human input the algorithm can build up consonant chords autonomously. These can gradually shift their harmonic focus as old pitches fade out of the chord and new pitches assume a focal role. Interacting with a human improviser produces chords which harmonize with the performer’s pitch material. Also, the level of consonance is informed by the improviser’s pitches: improvisation using a complex set of pitches gives the algorithm leeway to produce a response which is dissonant; in contrast, highly consonant improvisation constrains the algorithm to output which is similarly consonant.
Less expected is the fact that the algorithm plays faster when it is more dissonant. As the consonance level rises, the threshold of the consonance test becomes more demanding. Randomly generated pitches are less likely to pass this stringent consonance test than a more relaxed (dissonant) one, so as the consonance level rises the algorithm finds it ever more difficult to satisfy the test, and new notes enter the chord less and less often.
More subtle is the consideration that the pitch contents of the chord impose a control separate to the threshold value. If consonance is high the algorithm tends to reinforce the principal harmonic focus of its
chord, creating an ever more consonant chord, and with it slower and slower playing. Eventually it may subside into silence. Actually there is a tipping point between highly dissonant, fast and forthcoming playing; and highly consonant, ever slowing behaviour. This running down to a consonance followed by silence is quasi-cadential in character. After such a closure the algorithm’s own pitch buffer is empty and its responses to live input are slow to accumulate. The net result is to produce conventionally shaped phrases or sections, rising and falling both in terms of density and harmonic tension.

6. CONCLUSIONS

This dynamic system in the musical domain is comprehensible to humans with musical understanding (learned or acquired). For the programmer it is possible to interpret its behaviour by listening to the musical output, and to manipulate control parameters with an intuitive understanding of their musical effects.

For the human improviser, the responses of the algorithm also seem to be musically comprehensible. Flautist Finn Peters found that some responses, such as range following, were simple and fairly predictable; others (such as the development of harmonic fields) were judged appropriate, but not at all what might be expected from a human accompanist [11].

The system’s emergent behaviour produces serendipitous harmonic effects (quasi-cadential phrase endings). But it also generates non-harmonic temporal effects (faster playing under conditions of high dissonance, and phrase shaping) without explicit programming of temporal rules.

7. FUTURE WORK

It is desirable that the algorithm’s interactions with the performer be as strong as possible. Currently this algorithm is limited to a pitch-driven response. Implementing sampling of the live performer might allow timbral interaction, although there will be issues related to clear pitch extraction from sounds such as multiphonics and unpitched timbres.

In more general terms, dynamic systems in the musical domain offer further possibilities for the application of fundamental compositional concerns to system design, whether in interactive or autonomous presentations. Rhythmic and dynamics-based rules have been used by the author in other pieces [7]; melodic and timbral systems have yet to be explored.

8. REFERENCES


[10] Peters, F. Re: LAM 2009 [email], Message to: o.hancock@lcm.ac.uk, 2011.

