NN MUSIC: IMPROVISING WITH A 'LIVING' COMPUTER

Dr Michael Young

Music Department,
Goldsmiths, University of
London

ABSTRACT

This paper proposes attributes of a living computer music, the product of a *live algorithm*. It illustrates how these attributes can inform creative design with reference to a real-time system for solo performer-machine collaboration, *Neural Network Music*, and the PQ*f* framework proposed for live algorithms. Improvisation is treated as a classification problem at a high level of musical behaviour which can be measured statistically and train a multilayer perceptron neural network. Network outputs shape a stochastic-based synthesis engine. Mappings are covertly assigned, revisited by both player and machine as a performance develops. As the timing and choice of mapping is unknown, both participants are invited to learn and adapt to a responsive sonic environment which is created afresh on each performance. This offers a novel real-time application of feed-forward neural networks and a challenging, creative technological platform for freely improvised music.

1. INTRODUCTION

A *live algorithm* (LA)[1] is the function of an ideal autonomous system able to engage in performance with abilities analogous (if not identical) to a human musician, and produce a living computer music [3]. A true LA would employ methods distinct from established AI techniques to generate music from a rule-base, whether invented or derived from existing genres. Rather, a LA is relevant to creative, improvised performance where structure and character – in so far that they are evident – are emergent properties, and products of interaction within a heterarchical group. The emergence of mutually cooperative behaviours within improvising groups, and music events and structures which result, has been studied in Sawyer [13]. This phenomenon provides aspiration for living computer music.

An autonomous system, by definition, does not rely on human agency, so differs from the established practice of 'live electronics' by rejecting the notion of the computer as musical instrument. Any explicit *a priori* knowledge, agreement, or stated compositional design (with or without notation) would cast doubt on true autonomy. Any inference, whether in design or during live performance that a system is reliant on human input suggests only reactive, or only weakly interactive behaviour, which cannot be compared readily to the strong interactions evident within successful human group music-making.

1.1. Properties of a living computer music

The following proposed properties are integral to the idea of autonomous machine improvisation: adaptability, empowerment, immersion, opacity, and the unimagined. [17].

Adaptability is the ability to acclimatise to a shared audio environment, demonstrable in changes of musical behaviour. Lewis's term *emotional transduction*, defined as a "bi-directional transfer of intentionality through sound" [9], establishes the essential criterion that musical interaction should occur principally through the medium itself, rather than via control information. An alternative analogy, proposed by Blackwell and the author [4], is *stigmergy*, the process by which self-organising, structured behaviours of insect populations result from the interaction by individuals with their environment, not directly with each other. Musical performance, between players or players and machine can be regarded as similarly self-organising. Collaboration within a human/social environment involves a continuous assuming and casting of roles, and the development of a mutual history during music-making [7]. Such contextual contingency can be modelled with adaptable parameter mapping, optimization and machine learning.

Empowerment entails control over decisions that impact upon future experience. Decisions (or at least non-arbitrary changes in state) can be instigated by chaotic or complex systems: cellular automata, particle swarms [4] or neural networks [16]. These processes are not design-led, and evidence self-organising properties that can modify the audio environment and necessitate response from both human and machine participants.

Immersion occurs if there is an intimate, binding understanding shared by performers through informed listening and observation. Emulation of this should attend to nuance and broader musical states, realised as sound, rather than production techniques (lip pressure, gestural information), however intimate the control. A truly immersive, intimate relationship – as experienced by the performer – suggests *optimal flow*; a goal-orientated, mental state that explores the limits of experience and expectation, obtaining pleasure in meeting these challenges with appropriate skills [5].

[1] Live Algorithms for Music Research Network, Established in 2004 by Dr T. Blackwell and the author. UK EPSRC Grant no. GR/T21479/01.

Opacity is a prerequisite for this flow; an avoidance of naïve processes of cause and effect, via either direct control or a shared audio environment. The system ought to offer an ambiguous and shifting balance between the interactive and proactive, and across the threshold of the apparently chaotic and the readily predictable.

A fifth attribute of living computer music is suggested; an *unimagined* music, the unresolved 'work in movement' offered by collaboration of machine and human musicians on a would-be equal footing. If computers might extend, not copy, human behaviour through autonomous and prosthetic capabilities, machine music need not model established styles or practices. All contributions may have equal significance, but may not necessarily be equivalent; as a "musique informalle", such music is unfettered by external or inflexible influence, "free of anything irreducibly alien to itself or superimposed on it" [1], seeking its own means of emergent coherence.

2. A PERFORMANCE SYSTEM

These goals are addressed in the design of a system Neural Network Music (NN Music) in Max/MSP. It has been deployed with a number of instrumental combinations under the titles *au(or)a*, *piano_prosthesis* and *cello_prosthesis*.[2] All musical instances of NN Music bring together a solo player with computer, who mutually interact by proxy in the sonic environment. In this system, audio analysis and synthesis are mapped via a feed-forward neural network at the heart of the system. The network adapts to attributes of the performance, and outputs synthesis parameters accordingly.

Analysis focuses on underlying harmonic characteristics of the improvisation (rather than its step-by-step note progression) and extends it logically to provide a related and wider source of musical material. This approach is stylistically neutral and seems best suited to "non-idiomatic" free improvisation, as described by Bailey [2]. There is further analysis to identify other attributes of the performance together as a single musical 'behaviour' (for instance, a tendency to play loudly, intermittently, and in a high register) and then be learned by the network. Synthesis comprises the iteration of sound events based on a stochastic method; each iteration has its own parameter profile, and depending on the rate of iteration, the sound world can develop a "laminal" (textural) character or be more definitively note-based or "atomized" [11]. Stochastic techniques are well established in notated music and audio synthesis; for NN Music it offers the possibility of complex, mutable musical behaviours that only exist in an "interpretative state" [15]. This is well suited to the contingencies of the neural network outputs.

In the modular ***PQf*** architecture proposed for live algorithms [4], ***P*** is an analysis function, ***Q*** is a synthesis function and ***f*** is a hidden algorithm; any complex generative system, mapping function or AI learning mechanism (as in this case). ***P*** and ***Q*** interpret, and interface with the sonic environment, relaying parameters to and from the algorithm ***f*** respectively; this is analogous to the processes of listening, playing and creative thinking practiced by a human performer.

2.1. Analysis and Classification

P comprises two independent analysis functions that provide representations of the player's improvisation. The first function, $\boldsymbol{P_1}$, parameterises pitch characteristics, and the second, $\boldsymbol{P_2}$, offers a statistical representation of musical behaviour.

$\boldsymbol{P_1}$ produces a dynamic state $\mathbf{S_{chord}}$ which comprises a list $\{x_1, x_2, \ldots, x_n\}$ of the last pitches identified from the performer and approximated to the nearest quartertone sharp/flat. In current versions, $n = 6$. This analysis is refined by a statistical filter that determines the attentiveness of the system; the probability that an identified pitch will be allowed to update $\mathbf{S_{chord}}$. In its most attentive mode, all pitches are admitted, and when least attentive, there is only a small probability that one will be successful. The filter is deployed dynamically, mapped from the mean onset density detected over an adjustable time Δt, so relative inactivity on the performer's part fosters more attentive machine 'listening'.

As pitches are admitted, a generative function recalculates ten transposition tables by cross-multiplying each pitch within the primary set of six (see Figure 1).

$$f: S_{chord} \rightarrow S_{chord\_set}$$

This method emulates the post-serial technique of chord multiplication, devised by Boulez as, for example, identified in the "L'artisanat Furieux" cycle of *Le Marteau sans Maitre*. [8]. The obvious difference is that this function continuously updates S_{chord_set} in real time, as new pitches are admitted. S_{chord_set} is a dynamic pitch corpus, deployed as a resource for q_{pitch} , explained below.

The second function, $\boldsymbol{P_2}$, creates a dynamic performance state $\mathbf{S_{audio,}}$. This is a statistical representation, measured over time, Δt, of a number of audio descriptors $\{p_1, p_2, \ldots, p_n\}$ measured in 50ms windows. Familiar descriptors are used: pitch, loudness, onset density, sustained-ness (ratio of sound to silence). Other descriptors are included when relevant, depending on the solo instrument used: brightness (spectral centroid) audio periodicity and roughness. In all versions, the performance state $\mathbf{S_{audio}}$ comprises the normalised mean and normalised standard deviation of the parameters involved, measured over Δt and updated continuously, where $5s < \Delta t > 30s$.

.

[2] Audio recordings are available at the author's website, www.myoungmusic.com

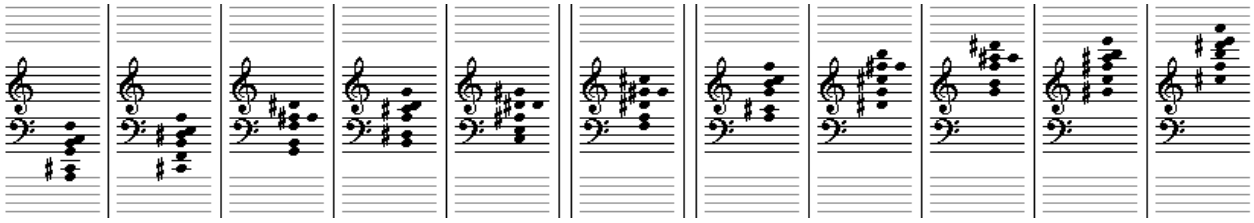

Figure 1. An example chord set. The primary set is the central chord.

These performance states offer a classification problem, which is well suited to the multilayer perceptron neural network. This is trained using a back-propagation error algorithm that minimises the error between required and actual outputs by gradient descent [16], given a set of pre-defined input and output conditions.

As noted by Toivianen [16], this network type benefits from its capacity for generalisation and tolerance to apparently unpredictable or contradictory data; consequently it is well suited to audio analysis of improvised musical material. This implementation involves two connected neural networks, **A** and **B**, in which **A** learns new input conditions from the performance and maps these to **B**. Two networks offer greater transparency for classification in the modular process of analysis-synthesis explained below. In parameter mapping [12] the networks implement convergent and divergent strategies, respectively, and the number of input/output nodes varies. Both have three hidden node layers. Implementation is with op.fann.mlp[3] for Max/MSP.

Classification problems match one input training pattern to a single output neuron. This is applied by taking $\mathbf{S}_{\mathbf{audio}}$ states as inputs to neural network **A**. This network retrains using the back-propagation method each time a new input state is received. The output of the network is an expanding truth table that represents each learned state. This could be described as a special form of adaptive convergent mapping, in which several inputs are mapped to a simple output in a perpetually expanding data space. The exact number of input and output nodes has varied in different instances of the system – 8-20 input nodes are typical – and the number of output nodes increases during performance.

It is not possible nor desirable for retraining to occur at the analysis rate: the aim of the process is to identify musical 'behaviours' that are well-defined and contrasting, so the network can be trained to respond effectively to a broad range of subsequent activity. To achieve this, the dynamic state $\mathbf{S}_{\mathbf{audio}}$ is considered for relearning only if fit: the fitness function f_{fit} is a measure of the similarity of the current state to all those previously learned. The function used, found through experimentation, is the sum of the mean and standard deviation of the absolute difference between the new state under consideration and each of the previously admitted states. This produces a list of values, $\{a_1, a_2, \ldots, a_s\}$, where s is the number of already admitted states. If any value of a is greater than a predetermined threshold z, the new state is allowed to update the network, which is retrained on the fly; otherwise it is discarded. In the current implementation the threshold is set by the user, as to be most useful it must adjust to characteristic behaviours of the instrument and performer.

$$f_{\text{fit}} : \mathrm{S}_{\text{audio}} \rightarrow \{a_1, a_2, \ldots, a_{s+1}\}$$

Once the musician begins, the network is trained with several new states – usually within the first few seconds, depending on the threshold value z. There is then a tendency for the time interval between retraining to increase, depending upon the character and structure of the improvisation and the consequent variance of $\mathbf{S}_{\mathbf{audio}}$ over time. As the performance develops, new analysis states will approximate one or, more often, several of those previously obtained. The network is continually queried to evaluate how far the current state $\mathbf{S}_{\mathbf{audio}}$ approximates any of those previously learned. This evaluation is mapped to the network **B** for synthesis, and is the first step of generative/synthesis ***Q***.

2.2. Sound Generation

A function f_{map} relays outputs of network **A** to network **B**, randomly re-sorting the indices of the data. This jumbling up of output and input nodes provides genuine opacity; it is covert, challenging the player to adapt to the system as its behaviour widens.

Network **B** creates new input nodes as s increases; in the versions of the system 60–100 output nodes are typical. They constitute a probability distribution for a stochastic synthesis function. Each synthesis state $\mathbf{S}_{\mathbf{synth}}$ comprises a probability distribution of a large number of output parameters q $\{q_{\text{duration}}, q_{\text{pitch}}, q_{\text{amplitude}}, q_{\text{density}}, q_{\text{sample}}, q_{\text{filter}}, q_{\text{stereo}} \ldots\}$. A higher-level parameter determines the probability of parameter-choice looping. Synthesis comprises the iteration of sound events; each iteration generates its own parameter profile according to the probability distribution. Events are MIDI-based in *aur(or)a* for solo instrument and disklavier. An instrument-specific sound corpus is used in *piano_* and *cello_prosthesis* comprising recordings of each individual note across the entire tessitura, with a number of playing techniques. The sound materials are also processed with filtering and ring modulation, according to the relevant parameters q. The advantage of this approach is its generality; new materials,

[3] An implementation of FANN, Fast Artificial Neural Network by Olivier Pasquet.

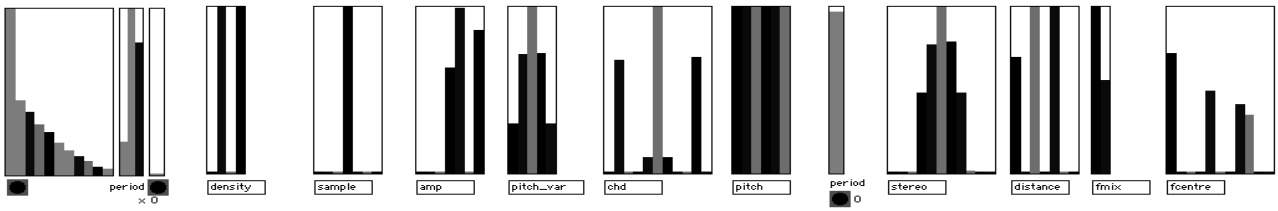

Figure 2. An examples of stochastic distributions for synthesis.

instrumental sounds and playing techniques can be added with ease to further expand the timbral vocabulary of the system. A library of stochastic distributions constitutes the knowledge base of the system; pre-composed material, albeit of highly mutable kind. Each version of the system has its own library of distributions. Figure 2 shows an example. In practice, the behaviour of network **B** is entirely dependent on the classifications made by network **A**. If a player suggests two previously learned behaviours (e.g., in reference to the earlier example, improvising quietly, intermittently, but in the low register) this will be reflected in a fusion of two output synthesis states. The choice of actual pitches is related to both the player's recent activity and the internal chord multiplication process.

3. CONCLUSION

The system NN Music developed for *aur(or)a* and *piano_* and *cello_prosthesis* functions on two levels; it contributes to the sonic environment in which the player is immersed and must adapt, and it is a sonic or gestural prosthetic. This is more evident in *_prosthesis* by use of sound materials related to the piano and 'cello. The reaction of the entire system, computer and resultant sonic environment, is prosthetic in effect; it aims to provide a technological augmentation of the performance capabilities of the musician and his/her musical expression. Even brief, 'low-key' interventions from the player can have substantial effects on the system and the resulting sound. Such a correspondence between performer and machine illustrates a shift in the user-computer relationship proposed by Stojanov and Stojanoski, in which the 'conversation' paradigm has been supplanted by the metaphor of prosthesis [14]. It is hoped the attributes proposed for living computer music, evidenced to some extent by NN Music, offer avenues for further research.

4. REFERENCES

[1] Adorno, T. "Vers une musique informelle". In *Quasi Una Fantasia: Essays on Modern Music*. Verso, London, 1961.

[2] Bailey, D. *Improvisation : its nature and practice in music.* DaCapo, London, 1992.

[3] Blackwell, T. and Young, M. "Live Algorithms". *Artificial Intelligence and Simulation of Behaviour Quarterly*. Vol. 122 pp. 7-9, 2005.

[4] Blackwell, T. and Young, M. "Self-Organised Music". *Organised Sound*. Vol. 9:2 pp.123-136, Cambridge University Press, 2004.

[5] Csikszentmihalyi, M. *Flow: The Psychology of Optimal Experience.* Harper Collins, 1991.

[7] Katovitch, M. "Temporary stages of situated activity and identity activation". In Couch, C., (Ed.) *Studies in Symbolic Interaction: The Iowa School.* Greenwich, CT: JAI Press, 1986.

[8] Koblyakov, L. *Pierre Boulez: A World of Harmony.* Harwood Academic, 1990.

[9] Lewis, G. E. "Too Many Notes: Computers, Complexity and Culture in Voyager". *Leonardo Music Journal.* Vol. 10, pp.33–39, 2000.

[10] Louzoun Y., and Atlan H. "The emergence of goals in a self-organizing network: A non-mentalist model of intentional actions". *Neural Networks.* Vol. 20 pp.156–171, 2007.

[11] Prevost, E. *No Sound Is Innocent: AMM and the Practice of Self-invention*. Copula, 1995.

[12] Rovan, J. et al. "Instrumental gestural mapping strategies as expressivity determinants in computer music performance." In A. Camurri (Ed.). *Proc. of the AIMI Int. Workshop,* pp.68–73. Genoa, 2007.

[13] Sawyer, R. K. *Group creativity: Music, Theater, collaboration*. Lawrence Erlbaum Associates, 2003.

[14] Stojanov, G. and Stojanoski, K. "Computer Interfaces: From Communication to Mind-Prosthesis Metaphor". *Proc. of the 4th International Conference on Cognitive Technology: Instruments of Mind.* Springer Berlin, 2001.

[15] Stroppa, M. "Live electronics or live music? Towards a critique of interaction". In, Battier, M. |(Ed.) *The Aesthetics of Live Electronic Music*. Contemporary Music Review Vol. 18:3. Routledge ,1999.

[16] Toivianen, P. "Symbolic AI versus Connectionism in Music Research". In Miranda, E. (Ed.) *Readings in Music and Artificial Intelligence.* Harwood Academic, 2000.

[17] Young, M. *Aur(or)a: Attributes of a Live Algorithm*. Electroacoustic Music Studies Conference, Leicester, U.K., 2007.