Beyond the Cybernetic Jam Fantasy: The Continuator

François Pachet
Sony Computer Science Laboratories Paris

The Continuator is a useable musical instrument combining techniques from interactive and automatic learning systems. It learns and interactively plays with a user in the user's style.

Music-generation systems have traditionally belonged to one of two categories: interactive systems in which players trigger musical phrases, events, or effects, such as the Karma musical workstation,[1] and systems such as Risset's interactive piano,[2] which allow for user input such as keystrokes or chords, but can't learn and use preprogrammed musical styles. Most of these systems propose *musical effects libraries* (a term used in the Karma workstation meaning a generation of music material based on user input). Although some of these effects are musically impressive, these systems can't be considered cybernetic musicians or even musical companions, because they use preprogrammed reactions and have no memory or facility for evolving.

On the other hand, many musical learning systems reproduce music in the style of a given composer, such as those Cope describes.[3] Here also, some results are impressive musically (music composed by Cope's system has been successfully performed and recorded). These systems are intrinsically noninteractive, however, and usually require additional human input—indeed, it is difficult to jam with an automata.

The Continuator project (http://www.csl.sony.fr/~pachet/Continuator) aims to fuse the two worlds—that is, to design interactive and useable musical instruments that can learn. As such, it can be seen as a realization of Kurzweil's prophecy,[4] which predicts that in the future, "Human musicians will routinely jam with cybernetic musicians." The principle underlying the Continuator is also applicable to nonmusical interactive learning systems.

Continuator

An interactive musical learning system must be able to

- learn musical styles in real time without a priori musical knowledge,
- account for real-time user input during the generation process to bias the generation and allow true musical dialogs (as opposed to musical triggers), and
- follow interaction protocols that give users total control over the music generated while enhancing users' musical expressiveness.

Design

To address these issues, I've developed a number of technical improvements to prior art in musical interaction[5] (see the "Related Work in Automated Musical Learning System" sidebar).

The system's *robust and efficient representation* of musical phrases accounts for polyphony, noise, and arbitrary rhythmic structures.

Its *extended multilayer Markov model* efficiently learns from arbitrary corpora of musical phrases in various styles. This model can generalize patterns found in musical phrases that are only somewhat similar, thus drastically speeding up the learning phase. Consequently, the system can immediately respond to musical phrases in unknown styles.

Finally, the *biasing mechanism* forces the Markov generation to specific harmonic regions. The mechanism lets users control system generation in real time, and thus avoids the mechanical-sounding effect of traditional music-generation systems. It does this by introducing a probabilistic scheme in the Markov-generation process that makes the process elastic. The scheme combines Markovian probability (the most expected continuation) with a fitness function (the most appropriate continuation with regard to external input).

Architecture

The Continuator's architecture consists of an *analysis* module and a *generator* module. The analysis module takes as input real-time MIDI sequences. This module has three main parts: a phrase end detector, pattern analyzer, and global property analyzer.

The phrase end detector uses an adaptive temporal

 Published by the IEEE Computer Society

Related Work in Automated Musical Learning System

Researchers in the artificial intelligence and information theory communities have long addressed the technical issue of learning automatically, and, in an agnostic manner, musical style.

In his seminal 1948 paper, Shannon introduced the concept of information based on the probability of message occurrence.[1,2] Other researchers, such as Brooks et al.,[3] used this notion to model musical styles. These experiments showed that by simply computing and exploiting note transition probabilities, the systems could generate pieces of music that sounded like given styles. More precisely, by analyzing a given corpus of musical material (typically musical scores or MIDI files), a system could compute transition probabilities between successive notes. It could then produce new music by generating notes using the inferred probability distributions.

Cope presents one of the most spectacular applications of Markov chains to music,[4] although his system isn't entirely automatically. Triviño-Rodriguez et al.[5] survey current Markov-based techniques for music, including variable-length Markov models, which capture stylistic information more finely.

Continuator is yet another musical Markov system but offers novel features.

References

1. C.E. Shannon, "A Mathematical Theory of Communication," part I, *Bell Systems Technical J.*, vol. 27, July 1948, pp. 379-423.
2. C.E. Shannon, "A Mathematical Theory of Communication," part II, *Bell Systems Technical J.*, vol. 27, October 1948, pp. 623-656.
3. H. Brooks, "An Experiment in Musical Composition." *IRE Trans. Electronic Computers*, vol. 6, no.1, 1957.
4. D. Cope, *Experiments in Musical Intelligence,* A-R Editions, 1996.
5. J.L. Triviño-Rodriguez, and R. Morales-Bueno, "Using Multiattribute Prediction Suffix Graphs to Predict and Generate Music," *Computer Music J.*, vol. 25, no.3, 2001, pp. 62-79.

threshold mechanism to detect when a musical phrase ends. This detector analyzes the time intervals between input sequence onsets to produce the threshold. Thus, if the input sequence is slow (that is, contains few notes per second), the detector increases the threshold; otherwise, it decreases the threshold. This simple mechanism ensures a temporally seamless continuation.

The pattern analyzer receives these input sequences and builds a Markovian model of them. (I describe the complete algorithm elsewhere.[5]) This analyzer parses a sequence left to right to build a tree of all possible continuations for all possible sequence prefixes. To speed up the learning process, the system also learns the sequence's transpositions.

The global property analyzer analyzes density (number of notes per second), tempo and meter (location of strong or weak beats), overall dynamics (loud or soft), and other input sequence properties.

The generator uses these analysis module properties to produce a continuation that is musically seamless with the input. The production of this continuation exploits the Markovian graph created by the analysis module, as I describe elsewhere.[5]

The generation process essentially entails producing the continuation note-by-note. The generator produces each note using the Markovian probabilities inferred during the analysis stage. Technically, it uses a variable-order Markov generation that optimizes the relevance of each single note continuation by looking for the longest possible subsequence in the graph.

I've been careful to perform meaningful segmentations of the input phrases in the learning phase. Indeed, real-world input phrases never consist of perfectly successive notes or chords. A segmentation process detects note or chord transitions in the input phrases and cuts the phrases into chunks, possibly across unfinished notes, which it feeds to the learning system. To retain the naturalness of the material's original style, the analysis module saves the possible residual discrepancy, restoring it at generation phase.

This continuation sequence is crude and unstructured in that it doesn't necessarily have the input sequence's global musical properties. I've therefore applied a mapping mechanism to transform the crude continuation into a musical phrase that will be played just in time to produce seamlessness. Currently, I analyze and map tempo, metrical position, and dynamics, but can easily use the mechanism to integrate other global properties from the input phrase.

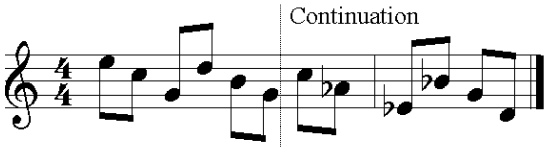
Musical interaction protocols

The interaction protocols define the rules of the game—that is, how the user interacts with the system in real time. I've defined two primary interaction protocols. In *Continuation or question and answer* mode, the user freely plays musical phrases. The Continuator detects phrase endings using a dynamic threshold and produces a continuation to the input phrase in the style learned so far. If the user plays a new phrase before a continuation ends, the system stops. These rules ensure that the user-generated music never overlaps the system-generated music, no matter what the user is doing.

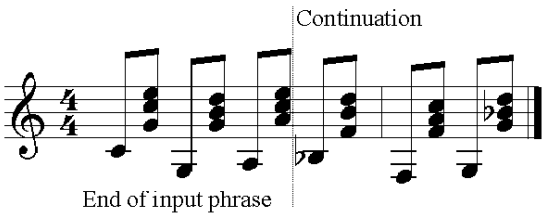
In *Collaboration* mode, the Continuator plays an infinite stream of music in some learned style (jazzy chord sequences, for example). The user plays a melody and the Continuator tries to adapt its generation to the user's input in real time.

Both protocols can lead to many variations. For example, a user could simultaneously launch several Continuators with the same input but different outputs, possibly with different sounds. This configuration creates polyphonic outputs whose individual parts share stylistic consistency but vary in real time.

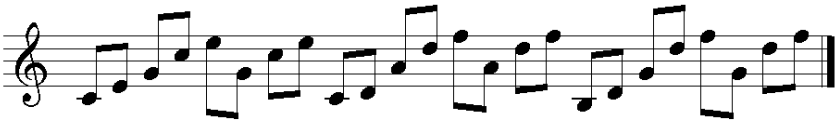
I've also defined other protocols in which the system decides a continuation's triggering based on a rule-based analysis of the input sequences. For instance, the system might only trigger continuations after given pitches or it might play chords as soon as the user plays



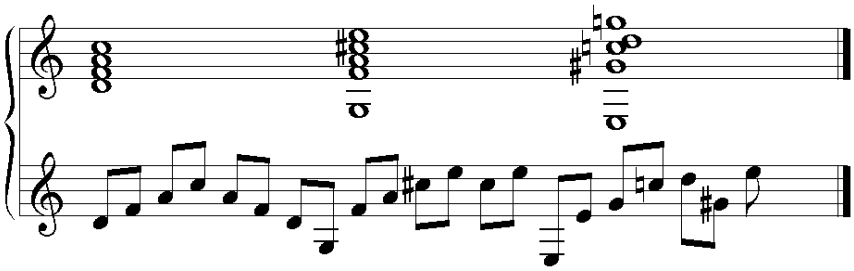
1 Simple continuation of two descending arpeggios. The system has no prior musical knowledge and continues the input phrase with more descending arpeggios. The continuation is temporally seamless and repeats at the next perceptually plausible temporal location (here, a quarter note after the last note).



2 Simple bass/chord sequence, continued by the Continuator without a priori knowledge.



3 Bach arpeggiator example. In a first phase, a user plays a Bach prelude in C, and the Continuator learns it (in all tonalities).



4 In the collaboration mode's second step, the user plays chords (top line), and the system reacts to them by playing Bach-like arpeggiations of the chords (second line).

a note. I don't report on these protocols here, but mention them to stress the core learning mechanism's total independence of the interaction protocol.

Continuation mode examples

In continuation mode, the generator produces continuations when it detects a phrase ending. Figure 1 shows a simple example in which a user plays a few arpeggios, and the system produces a continuation in the same style. The analysis and generation processes occur between the last note of the input phrase (a G in Figure 1), and the first note of the continuation (a C in the figure).

More complex scenarios are possible, particularly with polyphony. Figure 2 shows a simple example with a bass/chord sequence, continued in the same fashion.

More complex examples of continuations are available at http://www.csl.sony.fr/~pachet/Continuator.

Collaboration mode examples

Collaboration mode involves two steps. In the first, the user explicitly teaches the Continuator musical material. This is especially effective with a metrical style, in which the Continuator learns the musical sequences in relation to an imposed beat or tempo. The example in Figure 3 is a Bach prelude in C played by the user (or from a MIDI file) and learned by the system.

In the second step, the Continuator produces an infinite stream of sequences in the same style (in the following example, the sequences are ascending arpeggios using thirds of diatonic chords) while trying to adapt its production to a user-generated melody or other musical material. The mechanism for producing this compromise (described in more detail elsewhere[5]) entails substituting the generator's Markovian probability function with a function allowing for the fitness between the continuation and the user's melody.

Figure 4 shows a somewhat simplified example of music produced when the Continuator (bottom line) reacts in real time to the chords played by the user (top line), taking into account the style it has learned from the Bach prelude.

Although this example is a musical caricature, it shows the collaborative mode's basic principle. In some sense, the systems let users (musicians) literally play along with themselves. In the first stage, a user teaches the system his or her patterns, tricks, preferred chords, and so on. The user then plays a melody, and the Continuator accompanies the user using the learned material.

Experiments and effects

I conducted many experiments with the Continuator and professional jazz improvisers, including several performances at jazz festivals. In addition to generating technical improvements and ideas, these experiments showed that the system strongly affects its users. The most striking effect, systematically affecting all musicians experimenting with the system, was the *aha effect,* which occurred when users suddenly realized that the system was starting to play in their style.[6]

The musicians (György Kurtag Jr., Bernard Lubat, and Alan Silva) who played with the system expressed strong subjective impressions, which, though hard to define precisely, are illustrated by the following quotations:

- *Handles basic tasks* (Kurtag). "The system is like an amplifying mirror"; "It manages the past for me"; "It relieves me of my core, repetitive tasks, and lets me

perform high-level musical tasks, such as organizing superstructure in my musical discourse."

- *Triggers new ideas* (Lubat). "The system shows me ideas I could have developed, but that would have taken me years to actually develop. It is years ahead of me, yet everything it plays is unquestionably me."
- *Creates new forms of musical improvisation* (Lubat). "Because the system plays things at the border of the humanly possible, especially with the long but catchy melodic phrases played with an incredible tempo, the very notion of virtuosity is challenged. Virtuosity is becoming a musical object that can be created and manipulated in simple ways."
- *Relates to one's own learning* (Silva). "The system is doing what took me years to learn, in particular through Schillinger's book—that you can do much more with simple musical material (for example, a few notes) than what the scale-based approach tells you. It is a kind of materialization of Schilinger and Sloniminsky's vision."

Interestingly, the Continuator's use onstage also creates new modes of musical performance. Figure 5a shows Lubat during a concert at the Institut de Recherche et Coordination Acoustique/Musiqe (Ircam) in October 2002. The musician is deep in concentration, listening to the Continuator continuing his musical phrase for a few minutes. Later in the same concert, as Figure 5b shows, Lubat plays repeated ostinato patterns, then raises his hands. While the Continuator produces ostinatos in the same style, Lubat moves his hands across the keyboard as if he were actually playing. Audience reactions were amazement, astonishment, and often a compulsion to play with the system.

Experiments with children

The system also has obvious applications in music education. Preliminary experiments performed at a kindergarten in France show that children are sensitive to the system's imitative power. Even children with no musical training seem to develop instinctively personal playing modes. These modes might sound musically primitive, but careful study shows they are differentiated. For instance, as Figure 6 shows, a child can repeatedly hammer a single note with one finger or play chords with all fingers; stick to the keyboard's center or explore various regions; and play notes, arpeggios, and chords.

Experiments with the Continuator show that the use of the system in an imitative mode can benefit a child's musical behavior by

- pushing the child to explore new playing modes spontaneously;
- holding the child's attention longer (sometimes with a factor of 10); and
- helping the child develop listening skills, which are rare at primary-school age.

I am performing further systematic experiments in schools in Italy to further validate this hypothesis and others to better understand interactive music systems' impact and potential in musical education.

5 Continuator in use: (a) the musician listens to the Continuator during his performance and (b) makes gestures in a pretend play mode while listening to music produced by the Continuator.

6 A child playing with the Continuator. In imitative mode, the Continuator can help children develop musical behavior.

Extending the Continuator model

The Continuator is an instantiation of a larger class of interactive systems that can learn. The Continuator model (basically a tweakable real-time Markov generator) is a key element in many interactive, real-time systems that must satisfy both language-based consistency constraints and context-based adequacy constraints.

These systems include music playlist-generation systems, which produce music programs stylistically consistent with user taste. Developers can model stylistic consistency using a Markovian process, but user taste is typically a non-Markovian fitness function.

Dialogue-generation systems are another nonmusi-

cal application of the Continuator principle. From a given agent's viewpoint, an interesting dialogue occurs when another agent expresses a consistent individuality, but also remains focused on a shared subject of attention. Here also, developers can successfully model the personality consistency as a Markovian process, but the focus on a given subject is a typical non-Markovian fitness function.

The Continuator proposes a unified paradigm for modeling and implementing these systems. Furthermore, it explicates the compromise between the Markovian and non-Markovian forces as a parameter that users can control. This parameter (the *attachment*) lets users change the interactive system's personality to be more reactive or stylistically consistent.

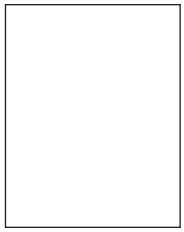
Conclusion

Whether the Continuator fulfills Ray Kurzweil's prediction is a statement we leave to the author of the *Age of Spiritual Machines*. However, the process of designing such a system has led to many interesting side effects that are at least as interesting as the initial goal.

In addition to the technical aspects pertaining to the implementation of a robust musical style-learning system, I believe this system is only one instance of a general class of systems that are both interactive and able to learn, whose main goal is not so much to produce consistent material, but to produce interesting interactions. This implies a radical shift in the software design process, departing from task-oriented systems to systems that primarily entertain. ■

References

1. S. Kay, *The Korg Karma Music Work Station*, Korg Inc., 2000.
2. J.-C. Risset and S. Van Duyne, "Real-Time Performance Interaction with a Computer-Controlled Acoustic Piano," *Computer Music J.*, vol. 20, no. 1, 1996, pp. 62-75.
3. D. Cope, *Experiments in Musical Intelligence,* A-R Editions, 1996.
4. R. Kurzweil, *The Age of Spiritual Machines*, Viking Press, 1999.
5. F. Pachet, "Music Interaction with Style," *Proc. Int'l Computer Music Conf.* (ICMC 2002), ICMA Eds., 2002, pp. 211-218.
6. F. Pachet, "Playing with Virtual Musicians: The Continuator in Practice," *IEEE MultiMedia*, vol. 9, no. 3, 2002, pp. 77-82.



François Pachet *is a researcher at the Sony Computer Science Laboratories Paris. His research interests include music feature extraction, music access systems, and music interaction. Pachet has a PhD in artificial intelligence from the University of Paris 6.*

Readers may contact François Pachet at Sony Computer Science Laboratories Paris, Audio Lab 6, rue Amyot, 75006, Paris, France; pachet@csl.sony.fr.