MUME 2016 - The Fourth International Workshop on Musical Metacreation, ISBN #978-0-86491-397-5

Building Blocks of Rhythmic Expectation

Jay Hardesty
Independent Researcher
Zurich, Switzerland
jayhardesty@gmail.com

Abstract

Rhythmic expectation is a key aspect of musical experience, but it has traditionally lacked a specific, low-level representation analogous to that provided by pitch and meter. A representation that encodes intersections of anticipation and repetition will be demonstrated to provide a degree of real-time control over variation and hybridization of rhythms. The encoding connects low-level musical structure to pattern formation of binomial coefficients on Pascal's triangle and self-similarity in the Sierpinski gasket. A generative music application demonstrates use of these encodings as building blocks upon a computational musical landscape.

1 Introduction

The domain of music creation typically includes rhythmic meter, which organizes time into discrete points with perceptually meaningful relationships to each other, and pitch, which organizes frequencies into scales that provide a basis for harmony. These are well-known low-level structures; no one assumes that musicians must ordinarily engage in raw frequency or temporal calculations to compose, improvise, or perform music.

Rhythmic expectation is another key element of musical experience that involves generic, low-level features (Meyer 1956, 35) (Huron 2006, 197). But the lack of traditional organizational components analogous to those of pitch and meter makes rhythmic expectation less ready to hand; its structure is not made particularly navigable by music representations such as standard music notation or digital piano roll interfaces.

This paper presents an application of rhythmic components described in (Hardesty 2016) to interactive, computer-assisted, music composition and improvisation. These components constitute building blocks that encode specific, low-level aspects of rhythmic expectation by fusing anticipation and repetition into atomic units, each of which is derived from other such units by very simple generative rules. The set of encodings provides a notably concise map of the branching possibilities for elaboration, syncopation and parallelism under those generative rules. The overall set has an approximately fractal structure, whereas the individual components have typical, non-fractal, rhythmic structure.

The goal here is to demonstrate that harnessing these building blocks provides an avenue for easing the technical barrier to hands-on music creation. They are not intended to model creative strategies or preferences (although they might provide inputs to preference rules). The building blocks instead form an axis along which formations of rhythmic expectation can be enumerated and interpolated. The (immediate) aim is not to simulate music creation, but rather to enable real-time navigation across a perceptually meaningful musical landscape.

A brief overview of the theory behind the application will be given in Section 3, followed by discussion of the application itself in Section 4.

2 Motivation

Beyond its application to computer-assisted composition, a further goal of this approach is to chip away at the immutable playback that has become an expected artifact of recording technology. The rhythmic building blocks described here facilitate adaptive treatment of musical material at the generative level. The resulting fine-grained spectrum of musical results has potential to support dual ecosystems of musical material, one where elements hybridize within emerging family trees of evolving variations, and one where music applications create, manipulate and consume those elements.

There are existing approaches, such as applications that incorporate user evaluation of previously generated results into the calculation of subsequently generated results (Hoover, Szerlip, and Stanley 2014). The approach described here makes adaptation a natural side effect of goal-driven behavior already familiar to composers, made more immediate by interactive shaping of variations on chosen musical influences. The aim is to leverage rather than disrupt creative processes, by selectively animating local musical structures within a larger composition.

3 Rhythmic Prediction

Metrical structure works in concert with parallelism to provide coherence when listening to music. As described by David Huron, musical meters are "predictive schemas for

temporal events" (Huron 2006, 197), where a "note onset in a weak metric position increases the probabilities that an ensuing note will occur at the next stronger metric position. [. . .] Syncopation occurs whenever a note onset fails to occur at the next higher metric position" (Huron 2006, 295). Fred Lerdahl and Ray Jackendoff stress the important role played by parallelism in making musical structure coherent (Lerdahl and Jackendoff 1983, 52); in the context of rhythm, repetition is the most basic form of parallelism.

Rhythmic building blocks that embody strict intersections of anticipation and repetition can be defined in terms of systematic elaboration and syncopation operations. The theory described in (Hardesty 2016) will be summarized in this section in order to provide background for describing an application that uses those building blocks.

The musical domain for this discussion is confined to short, looping contexts of quantized rhythms in strictly binary meter, as discussed in (Hardesty 2016). Even within those limitations, this domain encompasses a significant amount of music creation within genres such as EDM. This discussion will focus exclusively on rhythmic rather than pitch content.

3.1 Derived Rhythms

Rhythmic anticipation can produce either an elaboration or a syncopation, as shown in Figure 1.

Elaboration In an elaboration, anticipation is followed by the expected note arriving on the next stronger beat. The term *elaboration* is used in this discussion to refer to this specific musical operation or result thereof. That is, a note on a relatively stronger beat is elaborated by addition of a note at an immediately preceding, relatively weaker beat at some metric level.

Elaboration is the seed of repetition that embodies parallelism in this discussion; the resulting pair of notes can itself be elaborated at a different metric level, with that resulting pair of note pairs at yet another metric level, and so on.

Syncopation The term *syncopation* refers here to an anticipation that is not followed by the expected note at the next stronger beat. It is as if a note on that stronger beat had first been elaborated and subsequently deleted, leaving only the newly inserted note on the relatively weaker beat. Such syncopation can also be applied to a set of notes resulting from previous elaborations and/or syncopations.

3.2 Derivation Operations

A hierarchy of derived rhythms, starting from a single attack on the downbeat, can be generated using the following operations (Hardesty 2016):

1. shift the rhythm one beat earlier at metrical level m;
2. shift the rhythm one beat earlier at m and combine the result with the original rhythm.

Only a single operation can be applied to a particular rhythm at any given metric level. This key constraint, as well as the concept of a hierarchy of rhythmic components, is based on a theory of rhythmic derivation outlined by Arthur Komar (Komar 1971, 4, 55-57).

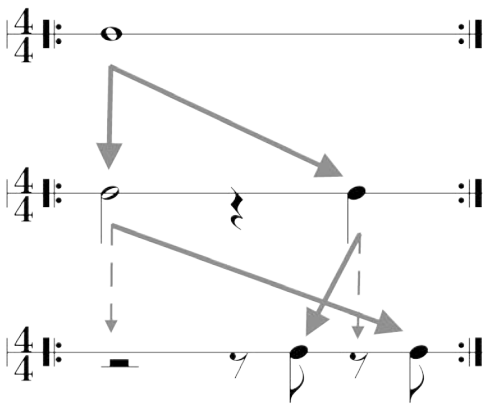

Figure 1: In the context of one looping measure, an elaboration is shown in the first step, followed by a syncopation of that elaboration in the second step. In the elaboration the existing attack is retained along with a copy shifted one quarter note earlier. In the syncopation both attacks are shifted one eighth note earlier, and the existing attacks are deleted, represented by the dashed lines. (The note durations are not significant.)

The first operation above constitutes a syncopation, the second constitutes an elaboration. The entire set of derived rhythms within n metric levels constitutes the building blocks that form the topic of this paper. Figure 2 shows the complete set of possible operations for two metric levels within one looping measure of $\frac{2}{4}$.

3.3 Emergence of Global Structure

The number of building blocks generated under the above operations might be expected to explode with branching possibilities. But as sometimes occurs upon iteration of scaling and shifting operations, self-similarity emerges that maps many of those possibilities onto shared outcomes (Peitgen, Jürgens, and Saupe 1992, 132).

Elaborations and Syncopations Mapped to Binomial Coefficients As described by (Hardesty 2016) the exclusive application of elaboration or syncopation at each metric level means that two binary numbers capture the sequence of operations encapsulated by a given building block. One number represents a combination of elaboration operations; the other represents a combination of syncopation operations. The pattern generated by each combination corresponds exactly to a patterns of odd binomial coefficients on some row of Pascal's triangle (Kummer 1852, 115-116) (Lucas 1878, 229-230). In the case of elaboration operations, a pattern of attacks is represented. In the case of syncopation operations, a pattern of possible offsets is represented.

Pascal's triangle is shown in Figure 3, followed in Figure 4 by rhythmic patterns formed by associating attacks with the odd entries on each row. The rhythms shown on the right result from the elaboration operations defined in Section 3.2.

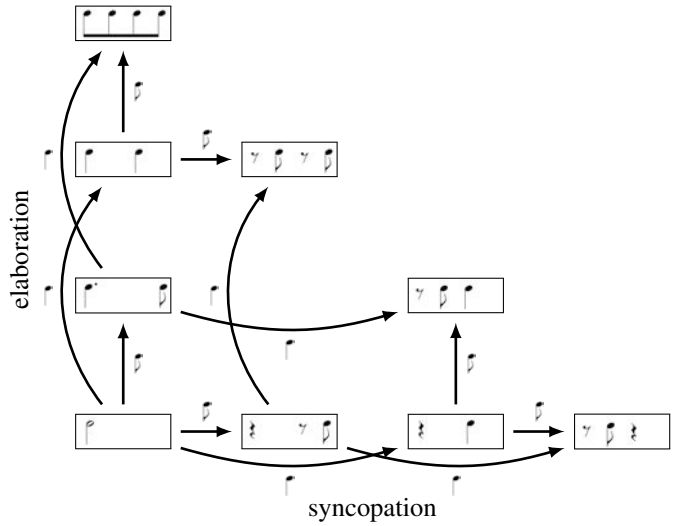

Figure 2: A hierarchy of elaborations and syncopations at eighth note and quarter note metric levels

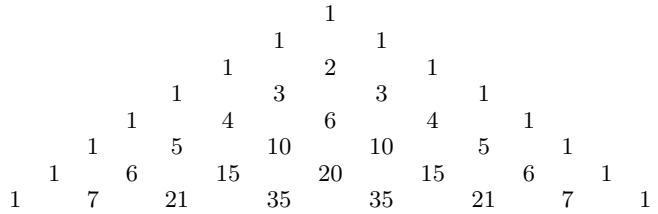

Figure 3: The first eight rows of Pascal's triangle.

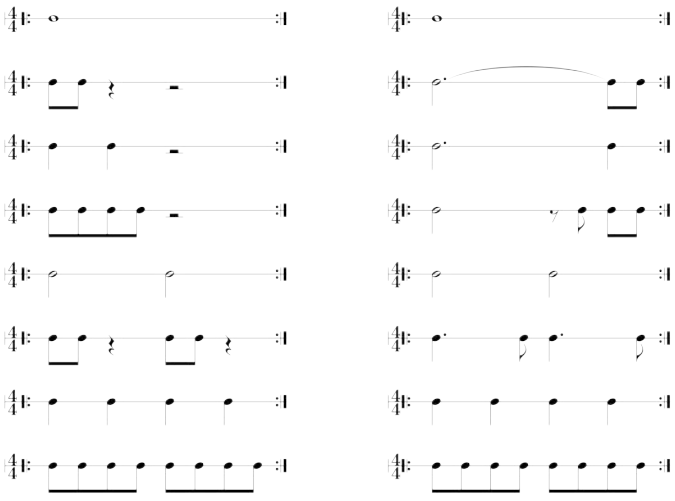

Figure 4: Rhythms arranged as rows on Pascal's triangle. Each attack corresponds to an odd binomial coefficient. On the right are elaborations, mirroring in time the rhythms on the left. Each is encoded as the binary expression of its row number, counting from 0.

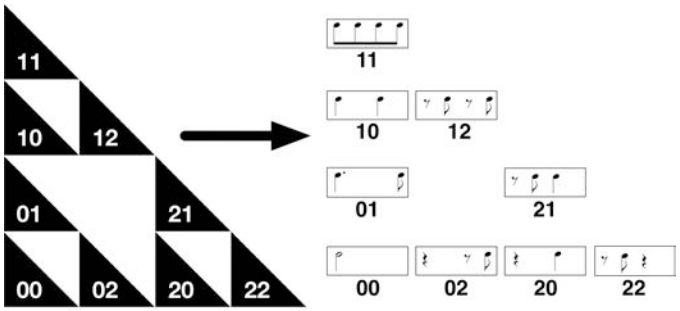

Figure 5: Elaborations and syncopations mapped to the Sierpinski gasket

Rhythmic Derivations Mapped to the Sierpinski Gasket Since operations are mutually exclusive, both with themselves and each other at any particular metric level, the two binary numbers, representing elaborations and syncopations respectively, can be combined into a ternary number which encodes a single building block (Hardesty 2016). Figure 5 shows the complete set of possible operations within two metric levels shown earlier in Figure 2 , now mapped to the Sierpinski gasket.

This mapping provides a concise means of enumerating and comparing strict intersections of anticipation and repetition. It makes the parsing of rhythms into building blocks computationally tractable.

In addition to direct manipulation as demonstrated in the following section, such a parsing has potential as a preprocessing layer for higher-level musical analysis and/or machine learning contexts. For instance, by operating on encoded rhythms, a machine learning approach might focus on discovering relationships that are a function of low-level rhythmic expectation rather than face the burden of rediscovering such generic structures, perhaps providing a starting point that is closer to that of an experienced listener.

4 Generating and Navigating Musical Variations

An application, Coord, has been developed to explore the manipulation of rhythmic components based on the representation summarized in the Section 3. The application generates on-the-fly musical variations and hybrids of monophonic melodies provided by the user. There has not yet been an empirical study of the results, largely because the theory behind the representation relies upon analytical results. But some examples will be provided to enable preliminary impressions.

Coord takes control of specific tracks within a larger music production, and it generates note patterns that replace the original contents of those tracks. The user selects two or three alternate input melodies per track, and Coord morphs between these to create variations that become the new output for that track. The user steers the results during playback, adjusting the relative influence of the input melodies by moving the pointer between icons representing input

melodies.

In short, musical moving parts are injected into a larger musical production, grafting generative analysis and variation onto selected musical elements. One consideration behind this application model is the subjective opinion that musical manipulations have a more dramatic character when they take place within a familiar context, in this case a composition containing multiple tracks, many of which are beyond any algorithmic control. For instance, a vocal track, with its specific nuances, might make a poor candidate for algorithmic manipulation in this context, whereas manipulation of a bass or lead melody against that vocal track could have a noteworthy effect.

This approach focuses exclusively on manipulation of short-term structure: short patterns in individual musical parts. This is in contrast to approaches that analyze long-term structures, including those with a focus on repetition (Paiement et al. 2008), and on meter (Roy and Pachet 2013).

4.1 Implementation

Coord is a Mac-based application written in the Swift programming language. It receives note data from, and transmits streamed note results to, a set of musical parts in Ableton's Live. Custom plugins, written using the Max/MSP programming language, extract note data from the tracks to be controlled by Coord, and handle commands, timing and note traffic between Live and Coord. The communication protocol used between the two apps is Open Sound Control (OSC).

Live's Session view, each track typically has a number of alternate MIDI loops that can be played in conjunction with loops currently being played on other tracks. These loops are treated as interchangeable music elements; within a given track one loop can be triggered at any time to the exclusion of the others on that particular track. For instance, a bass track might have three different note patterns that can be alternately triggered, automatically synced to the overall meter of the piece.

Rather than switch entirely between one input melody and another in this manner, Coord generates variations that are simultaneously a function of multiple input melodies. It analyzes the rhythm of each input melody and applies the combined influence of those analyses to generate a new rhythm and melody. The relative influence of the inputs is under the real-time control of the user; the application continually updates the note stream being streamed back to the relevant track in Live during playback.

4.2 Generative Analysis and Composition

The aim of this section is to describe an application of the theory outlined in Section 3 in pursuit of goals mentioned in Section 2. It is not claimed to be the most effective or sophisticated potential use of that theory, but rather to demonstrate how directly and simply the building blocks can be leveraged to generate musical results. In hopes of conveying a snapshot of this ongoing exploration, a high-level outline of the algorithm will be provided, rather than a formal description. The focus here is solely on generation of rhythmic

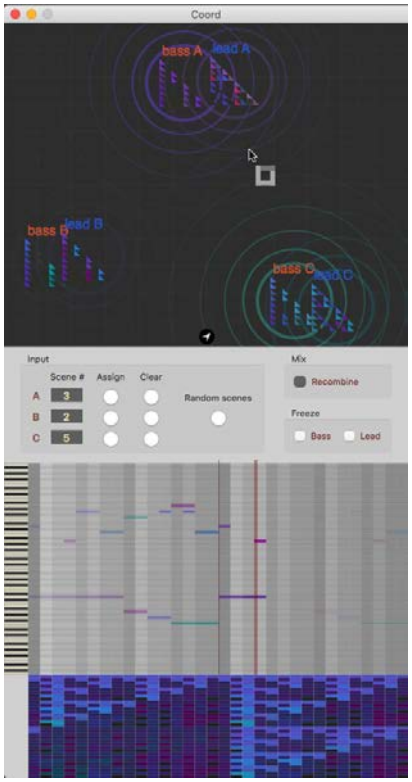

Figure 6: Coord application

variations. The pitch content of generated melodies is beyond the scope of this paper.

Coord's core algorithm profiles a given rhythm in terms of how it does or does not raise specific low-level expectations, and how it does or does not it fulfill those expectations. The algorithm does apply any preference rules to such profiles; it merely uses the profiles as a basis for comparison. It relies upon the building blocks described in Section 3.1 to capture and compress this information in a form that can be compared and interpolated directly. The algorithm thereby morphs between input rhythms, producing output that can be steered "closer" to this or that input rhythm.

The algorithm currently operates only on strictly binary, quantized, rhythms. Only attacks, not durations, are considered; each rhythm is a series of 2^n time points within n metrical levels, where each time point is interpreted as either an attack or a rest. First each input rhythm is analyzed independently. Then those analyses are interpolated to generate a new rhythm.

Analyzing Input Rhythms The algorithm analyzes each input rhythm and derives an attack probability for each time point, where that probability reflects not only the simple presence of an attack or in the input, but also the degree to which an attack there agrees or clashes with expectations raised by other attacks. Each time step is considered as a potential attack that could belong to a number of hypothetical building blocks. A notion of distance between those hypo-

thetical building blocks and the actual building blocks determines the likelihood of a generated attack at that time point.

This analysis requires a measure of distance between building blocks. As described in Section 3.2, each building block is encoded as a ternary integer; this encoding is called an *address* because it denotes a particular path through the possible operations at each metrical level. The rhythm that results from the combined operations represented by an address will be called a *branch*. The distance between two branches is taken to be the Hamming distance between the ternary string representations of the respective addresses, that distance being a count of the metrical levels where the two branches represent different operations.

Given this representation and distance measure, the algorithm is straightforward: a given input rhythm is parsed into a set B of (potentially overlapping) branches, the union of which forms that input rhythm. B excludes any branch which is a proper subset of any other branch in the set. [1]

There are 3^n hypothetical branches for n metrical levels, of which 2^n branches contain any given time point. In order to calculate the derived attack probability for a particular time point, the distance is measured between each of the hypothetical branches that contain that time point, and each branch in B. The derived probability for that time point is them set to 2^{-d} where d is the minimum of those distances. In effect, every mismatch between the operations encapsulated by a closest hypothetical and actual branches halves the probability of an attack at that time point. Note that for any time point containing an attack in the input rhythm, that minimum distance will be zero, because one of the hypothetical branches containing that time point will belong to B.

Generating Rhythmic Variations After the derived attack probabilities have been calculated for each input rhythm, a user-controlled scalar is applied to each set of probabilities, determining how much relative influence each input rhythm will have on the generated result. The resulting scaled probabilities are summed for each time point, which is then assigned an attack if the sum is equal to or greater than a user-defined threshold between 0 and 1.

A key to understanding the biases introduced by this approach is the mutual reinforcement, or lack thereof, of rhythmic structures produced by elaboration and/or syncopation operations within and between the input rhythms. When probabilities are combined from multiple sources, the cumulative probabilities that exceed the threshold necessary to generate attacks are projected to potentially different sets of time points than those containing attacks in the individual input rhythms.

Figure 7 displays the relative distances between input rhythms on the landscape where the user navigates musical result. Figure 8 visualizes the corresponding arrangement and distances of the building blocks on the Sierpinski gasket. Figure 9 gives a rough sense of the collective probability

[1] if A and B are branches where $A \subset B$, then B indicates a sequence of operations that includes an elaboration missing from the sequence of operations indicated by A.

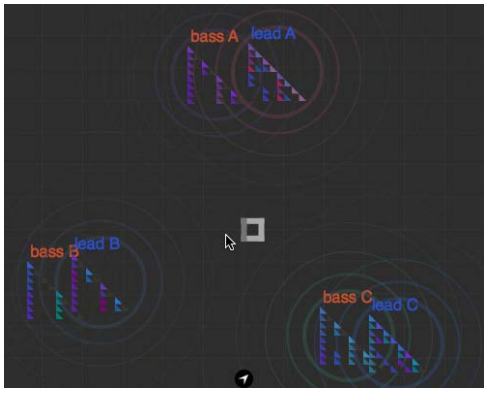

Figure 7: Pointer on landscape, determining relative weights of three pairs of (bass/lead) input melodies

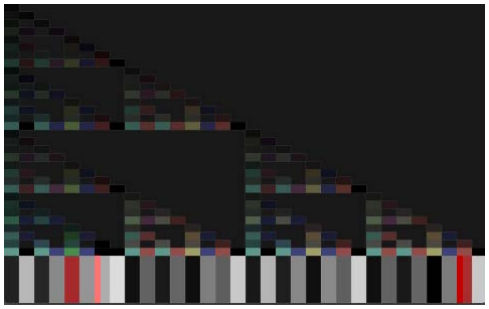

Figure 8: Visualization of probabilities corresponding to the location in Figure 7 mapped to points on the Sierpinski gasket. The grayscale slices at the bottom represent the cumulative probabilities at successive time steps. (The red slices display the current playback time and next attack time respectively.)

projected during playback onto each attack time, relative to the threshold required for sounding notes.

In each of the Figures 8 and 9, the color red indicates influence of the lead rhythm at the top of the landscape shown in Figure 7, green indicates influence of the lead rhythm at lower left of Figure 7, and blue indicates influence of the lead rhythm at lower right. (An accompanying set of building blocks, distances, and attacks derived from the corresponding bass melodies is not shown.)

Interactive Music Landscape Coord's GUI presents each of the selected input rhythms as icons placed on a two-dimensional landscape. Each melody acts as a landmark, and the user controls the influence exerted by each melody by moving the mouse pointer closer to this or that landmark. The pointer distance determines the weights assigned to the building blocks parsed from each melody.

This accomplishes a music creation feedback loop intended to mirror aspects of traditional music improvisation: the user chooses melodic influences, navigates a fine-grained spectrum of variations, and continually adjusts course based on judgement of the musical results during playback. The

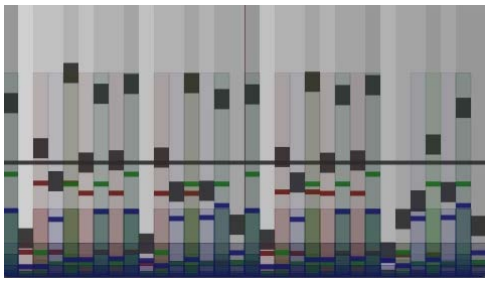

Figure 9: Attacks probabilities above and below threshold to emit output, corresponding to the location in Figure 7

user is acting directly upon familiar musical material, creating new material that is recognizably related to the selected inputs, but in non-obvious ways. The goal is to discover music that sounds like something one might have written or played using traditional means, within a spectrum of possibilities that could not otherwise be iterated without laborious effort and technical training. The algorithm makes no recommendation for navigation or judgement of the results; authorship and musicianship are genuinely in the hands of the user.

4.3 Demonstration of Results

Some video examples of Coord in use are online at http://coord.fm/mume-2016.

5 Conclusion and Future Directions

The approach described here proposes an axis of interaction with existing musical elements, providing a particular means of navigating material based on relationships that underpin specific low-level aspects of musical coherence and familiarity. Such interaction might make an important aspect of musicianship more accessible both to humans and to algorithmic frameworks.

Future development will focus on more sophisticated statistical frameworks, empirical evaluation of results, low-level incorporation of musical pitch, and on extending the representation to non-binary meters.

References

Hardesty, J. 2016. A self-similar map of rhythmic components. *Journal of Mathematics and Music* 10(1):36–58.

Hoover, A. K.; Szerlip, P. A.; and Stanley, K. O. 2014. Functional scaffolding for composing additional musical voices. *Computer Music Journal* 38(4):80–99.

Huron, D. 2006. *Sweet Anticipation: Music and the Psychology of Expectation*. Cambridge: MIT Press.

Komar, A. J. 1971. *Theory of Suspensions: A Study of Metrical and Pitch Relations in Tonal Music*. Princeton: Princeton University Press.

Kummer, E. 1852. Über die ergänzungssätze zu den allgemeinen reciprocitätsgesetzen. *Journal für die reine und angewandte Mathematik* 44:93–146.

Lerdahl, F., and Jackendoff, R. S. 1983. *A Generative Theory of Tonal Music*. Cambridge: MIT Press.

Lucas, É. 1878. Théorie des fonctions numériques simplement périodiques. In *Amer. J. Math.*, volume 1. 184–240.

Meyer, L. B. 1956. *Emotion and Meaning in Music*. University of Chicago Press.

Paiement, J.-F.; Grandvalet, Y.; Bengio, S.; and Eck, D. 2008. A generative model for rhythms. In *Neural Information Processing Systems, Workshop on Brain, Music, and Cognition*.

Peitgen, H.-O.; Jürgens, H.; and Saupe, D. 1992. *Chaos and Fractals: New Frontiers of Science*. Berlin and New York: Springer-Verlag.

Roy, P., and Pachet, F. 2013. Enforcing meter in finite-length markov sequences. In *27th AAAI Conference on Artificial Intelligence*.