Resonant Element: An Application for the Continuous Generation of Pop-Inspired Music

B.T. Franklin

Dunesailer Research
btfranklin@dunesailer.science

Abstract. This paper describes the Resonant Element music generation system, which is capable of generating a continuous stream of music based upon observed attributes of Western popular music. It explains the conceptual foundations of the architectural strategy, including the intended use cases of the system. It summarizes the representational structure and generation process of the application, with particular focus placed upon the use of chord progressions, stochastic note transitions, histograms, and LRU pattern caching. It provides a basic evaluation of the system, along with a summary of future development directions.

Keywords: generative music, stochastic model, Markov chain, continuous music, MIDI output, note histogram, background music, chord progression, computational creativity

1 Introduction

Algorithmic frameworks and systems for the generation of music are abundant, and can be segmented based upon various philosophical and practical characteristics. These include the underlying generative theory[1], whether or not the music produced will be continuous, whether or not the system will be interactive, what general style or genre of music will be produced, and so forth.

The Resonant Element application has been designed and implemented from an opinionated stance about where it should fall within this topology. The system applies a natural, or emergent, generative theory using stochastic modeling. The music is generated continuously, not discretely. The genre of the generated music is configurable, but consistently draws from progressions and techniques extracted by an analysis of a large corpus of popular music. Importantly, the primary use case for the software is intended to be for passive background listening, so the system aims to generate music that is pleasant to listen to for extended periods, but is not so novel that it becomes a distraction from other tasks.

Herein is described the design process and reasoning, algorithmic details, lessons and discoveries obtained during the implementation of Resonant Element.

2 Conceptual Foundation

As mentioned above, passive listening alongside the performance of other tasks is one of the primary guiding use cases for which Resonant Element is intended. In light of this, the approach to music generation needs to consider the characteristics that are most compatible with this use case, and the use case contains various modes depending on the task to be performed during the listening session. For example, the music generated for listening while trying to sleep or relax must be very different from the music generated for doing creative work such as writing prose or computer code, and these must in turn be different from the music generated for listening to while exercising or doing a similar high-energy activity.

While each of these use cases contains markedly different required musical attributes, the identification of common underlying characteristics was also necessary in order to formulate a coherent and cohesive system.

2.1 Popular Music

Western culture has given rise to an enormous variety of musical styles and genres, and it would be extremely difficult to attempt to reproduce all of them in any single generative system. Simultaneously, if music can be said to reflect the culture that creates it, then it stands to reason that the most popular music within a culture is likely to also be a strong reflection of the generating culture's aesthetic opinions and values. Of the subset of human emotions that music is commonly capable of expressing or evoking[2], popular music (commonly referred to as "pop" music) has been used to express or evoke all of them.

Significant analytical research has been done on the musical tropes that appear in the corpus of pop music, in particular with regard to the frequency of specific chord progressions[3]. This provides a solid probabilistic basis from which to construct a stochastic model.

Consequently, pop music was chosen as the guiding musical influence upon which to construct the *SongSkill* system that was the predecessor of Resonant Element, and which has been described in a previous publication[4].

2.2 Emotion Evocation

As mentioned previously, the immediate predecessor of Resonant Element was the *SongSkill* system, which was created based upon a use case of evoking specific selected emotions[4]. While the use case goals of Resonant Element are not the same as those of *SongSkill*, it did inform the design and implementation of the later application, and builds upon the research findings of that project. A cornerstone finding of the prior work was that it is effectively impossible to reliably evoke a particular emotion across a cross-section of listeners due to the nature of emotion evocation being strongly dependent upon personal experience and aesthetic preferences. In contrast, it is relatively easy to express a particular emotion musically, at least within the bounds of a specific set of cultural

expectations[5], even using an automated rule-based system. As a result of this finding, the latter approach became the focus of the newer system.

2.3 Aesthetic Clustering (Genres)

Although the generative model of Resonant Element is based upon the probabilities identified within the broad umbrella of popular music, a core thesis of the project is that the same stochastic model can be extrapolated into use for the production of other genres that reflect similar Western cultural and aesthetic expectations. To achieve this, the system allows configuration of the generative parameters based upon a set of pre-determined values, grouped by genre into "stations" (to reflect the nature of their use, which is similar to the selection of a radio station).

2.4 Background Listening

In order to satisfy the primary use case of passive listening in the background while the user performs other tasks, it was necessary to ensure the presence (or absence) of specific musical characteristics from the generated output.

For stations intended for use while relaxing, meditating, or attempting to sleep, use of melody is avoided, as this tends to produce too much "interest" in a listener, stimulating an unwanted level of alertness. Additionally, music with a tempo near 60 beats per minute (BPM) has been shown to be more conducive to relaxation[6], so this target value is used by the system.

Research by Haapakangas et al.[7] has shown that understandable lyrics in music hinder task performance in the listener due to distraction, and work by Huang and Shih[8] has shown that music a listener feels too strongly about (either positively or negatively) has a similar effect. In light of this, Resonant Element avoids the use of any vocal components, and is designed to generate music that will be received by most listeners as being of mediocre to moderately good quality, and because the music is generated continuously and does not replay previously-generated pieces, no particular attachment can be developed in the listener to a specific piece.

3 Representational Structure

The system is written entirely in the Swift programming language, and runs natively on both macOS and iOS platforms (though the user interface is somewhat different between the two platform versions).

3.1 Musical Encoding

The representation of musical structure is accomplished through a design based upon the various types of music tracker software that were popularized in the 1980s and 1990s[9], in which music is grouped into "patterns", which in turn

consist of "tracks" or "channels" that have "rows." This very orderly, row-and-column based system was selected because it provides a highly manageable way to access specific notes and ranges of notes for copying, manipulation, or examination at runtime. It is also highly compatible with a playback mechanism that operates during discrete moments of runtime availability, rather than requiring a continuous thread of operation to achieve uninterrupted playback.

In Resonant Element, this representation consists of "sections" which contain "note strips," which in turn contain "rows." Each row of a note strip is able to store an arbitrary number of instructions. These instructions represent commands to start a particular note, end a particular note, change the tempo of playback, set the playback volume, or change the instrument that is being used for the playback of the specific note strip.

3.2 Playback

Playback is accomplished through the built-in MIDI playback features that are available within both macOS and iOS. A playback layer was created in order to map the module-like structure described above to the native MIDI event triggering facilities. A sound font is installed in order to provide a higher degree of realism in the generated output.

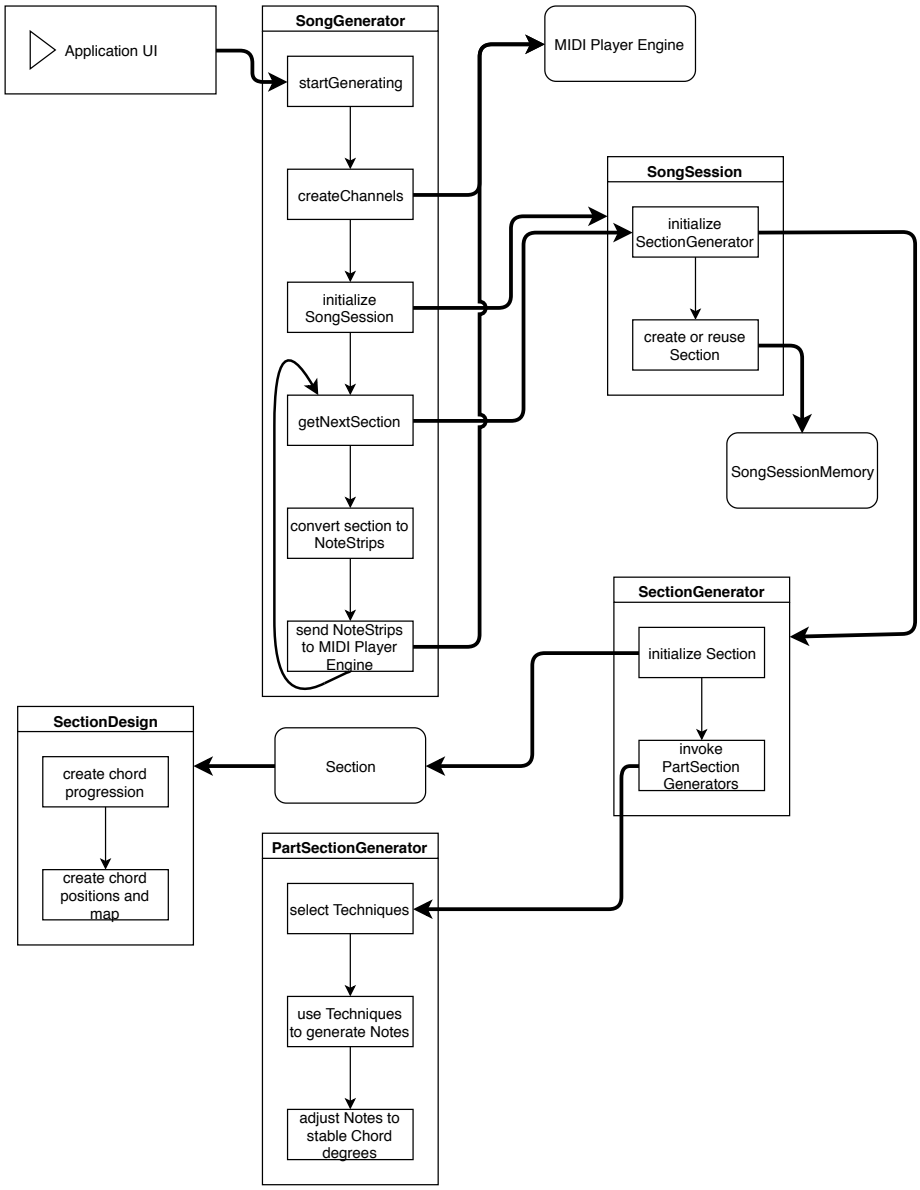
A timed thread execution process is used to advance playback through all of the note strip rows of each section to be played. Since beats are represented by a specific number of rows, tempo is implemented in a straightforward way by simply controlling the rate at which rows are read and executed from their containing note strip. This playback mechanism can most easily be envisioned as being similar to the paper strips of music used to control a pianola, in which notes that occupy the exact same horizontal row are triggered simultaneously. In fact, this is why the relevant data structures in Resonant Element are called "note strips."

4 Generation

When a new musical section is generated, the process occurs iteratively by focusing on each instrument part in sequence. The exact parts involved are selected randomly (with weighted probabilities) from a set of options that are specific to the genre of music being generated at the time. During the generation of each specific instrument part, various performance techniques are randomly selected from an available set of options, which also vary with the genre. These techniques are used to add notes to the note strip until the desired length of time has been fully populated, at which time the next instrument part is generated. This continues until every part has been fully generated for the section.

A diagram illustrating an overview of the architecture of the generation system is shown in Fig. 1

Fig. 1. Resonant Element Architecture



4.1 Chord Progression

A core element of the system's generation process is the determination of a particular chord progression to use across the section being generated. This is achieved by first randomly selecting a series of durations for each chord that will be active throughout the overall duration of the generated section. For example, a typical section is four measures in length. Each chord might be one measure in length, meaning four chords would be used in the progression. Another configuration uses three chords, each of which is one measure in length, followed by two chords, each of which is half a measure in length, producing a chord progression that involves five chords in total. The set of possible chord duration configurations was created by an informal summary analysis of the corpus of popular music made available as a part of the analysis performed by Miyakawa et al.[3]

Building upon the chord layout, a sequence of specific chords is selected through the use of a Markov chain, starting with the selection of a first chord using the probabilities shown in Fig. 2 and transitioning through each subsequent one until the entire sequence is populated. The stochastic probabilities used for the transitions are also based upon the analysis performed by Miyakawa et al.[3], with some small modifications to remove edge cases and make the data easier to understand. The model is described in Fig. 3.

Fig. 2. Starting chord probabilities. Chords are identified by degree.

Chord Degree	Probability (%)
I	40
II	5
III	5
IV	15
V	25
VI	10

After the entire sequence has been populated with specific chords, a final adjustment phase optionally modifies the cadence (ultimate) and pre-cadence (penultimate) chords in the series to conform with popular musical conventions as described in Miyakawa et al.[3]

Fig. 3. The starting chord is shown at left. Destination chords to are shown at top. Each value is expressed as a percentage chance of the transition occurring. Higher probabilities are indicated with darker color. Chords are identified by degree.

	I	II	III	IV	V	VI
I	0	5	5	30	45	15
II	15	0	10	20	25	30
III	5	10	0	40	10	35
IV	40	5	5	0	40	10
V	30	5	5	30	0	30
VI	15	5	5	40	35	0

After the chord progression layout has been completed, the chords are interpreted using a variety of different performance techniques, such as playing all the notes at once (block chord), playing the notes individually in a sequence (arpeggio), or playing a pattern using various combinations of note timings. The techniques that are available for selection vary with the genre that is being generated.

4.2 Stochastic Note Transitions

The system is capable of generating melodies that are harmonically compatible with the underlying chord progression. The melodies are monophonic, which means their production can be handled as a stochastic progression through individual note values (including rests) with varying durations.

After a starting note pitch value is selected, the duration is selected randomly using a stochastic probability model, with a duration of one beat being the most likely, and with one-quarter beat and three beats being the shortest and longest possible durations, respectively. All of the subsequent notes are selected using another stochastic model of transition directions and extent in which the most likely transition is zero (meaning to play the same note pitch again), and progressively larger transitions either up or down in the scale become less probable in an approximately normal distribution, with the largest delta being three scale degrees. Constraining the size of the transition in this way produces a more co-

herent and linear "feel" to the generated melodies. This process is repeated until the entire duration of the generated section has been populated.

The specific values of the stochastic models employed during the melody generation process vary with the genre, allowing changes in the "feel" of the melody by adjusting the median duration, curve shape of the pitch transition probabilities, and likelihood of the placement of rests.

4.3 Histograms

The use of dynamic histograms to determine or influence note placement is a novel feature of the Resonant Element application. Specifically, the system utilizes a variety of techniques that reference a shared histogram which is populated progressively as each subsequent instrument part is generated for a particular musical section. By selecting the instrument order carefully, the later instruments are thus enabled to respond adaptively to the note placement used by the instruments for which parts have already been generated. Normalizing the histogram at the moment when it is used allows simple percentage-based probabilities for determining note placement.

When these histogram-driven techniques are employed, the end result creates an illusion of musical intention across the various instruments. For example, a bass guitar part might play a note at every place where any other note by any other instrument is played. A snare drum might be played only at places with somewhat higher musical density. A crash cymbal might be played only at places where the highest musical density occurs. See Fig. 4 for an illustration of this.

Fig. 4. The table below illustrates a hypothetical case where three instrument parts have been generated for a single measure, each contributing to the shared histogram. The normalized histogram value (n) is then used in the generation of the later instruments with varying "trigger values" to determine placement of the final three instruments.

	Beat															
	1				2				3				4			
Instrument 1	■				■		■				■		■			■
Instrument 2	■		■						■		■		■	■		
Instrument 3		■			■	■			■				■		■	
Histogram Value	2	1	1	0	2	1	1	0	2	0	2	0	3	1	1	1
Normalized Value (n)	0.7	0.3	0.3	0.0	0.7	0.3	0.3	0.0	0.7	0.0	0.7	0.0	1.0	0.3	0.3	0.3
Bass Guitar (n > 0.25)	■	■	■		■	■	■		■		■		■	■	■	■
Snare Drum (n > 0.5)	■				■				■		■		■			
Crash Cymbal (n == 1.0)													■			

4.4 Section Caching

Repetition is an important element of popular music, and it lends a sense of coherence to the music, while also producing a sense of familiarity in the listener. Pereira et al.[10] have shown that familiarity via repetition is an important element in establishing emotional engagement in a listener. However, the passive listening use case of Resonant Element raises the concern that creating too much emotional engagement can actually become a form of distraction. The system attempts to address this by striking a balance between familiarity and novelty, achieved by storing recently-generated musical sections in a least-recently used (LRU) elimination cache with relatively small capacity (a maximum of four patterns can be stored at a time).

When a new section is needed for playback, there is a chance that one of the already-played sections stored in the cache will be used instead of a new one being generated. If this occurs, the section is moved to the top of the cache, since it becomes the most-recently used item. If a new section is generated instead, then it is placed at the top of the cache, and the least-recently used section is discarded. Because all of the sections in the cache are available with equal probability of selection regardless of their current cache position, this strategy strikes an effective balance between reuse of recent sections (to achieve a sense of familiarity) and ensuring that older sections are not replayed (to ensure the ongoing production of novel musical sections).

5 Evaluation

Providing a quantitative evaluation of a system such as this is outside of the scope of the research goals of this project, and it can be argued that it would be of little practical value to perform such an analysis. It is tempting to turn to devices such as Alan Turing's "Imitation Game" (often referred to simply as the "Turing Test") as a way of evaluating the quality of a generative music system, but closer examination reveals that this is not likely to be valuable, either. According to Ariza[11], use of the Turing Test

> in the evaluation of generative music systems is superfluous and potentially misleading; its evocation is an appeal to a measure of some form of artificial thought, yet, in the context of music, it provides no more than a listener survey.

This challenge in evaluating such a system may raise the question: What is the goal of such an evaluation to begin with? The argument can be made that if Resonant Element is able to successfully be used by any person for its intended use cases, then it is "good enough."

In discussing the evaluation of generative musical creativity, Loughran and O'Neill[12] identify a difference between the evaluation of the behavior of the system and the evaluation of its outputs or artifacts. In evaluating the system rather than its artifacts, the authors suggest that the act of submitting projects

to relevant conferences and journals for peer review is a form of evaluation, since the underlying processes and reasoning are the focus.

In the case of Resonant Element, the output is continuous and transient, making evaluation of any given artifact somewhat meaningless, other than in aggregate.

It is substantially more straightforward to evaluate some of the practical realities of the system's use, which are related directly to the quality of the user experience, and therefore to the fulfillment of the intended primary use case of supplying music for passive background listening while the listener performs other tasks.

5.1 Playback Timing

The native Apple MIDI playback engine that is used by Resonant Element is not really intended to be used in a timing-controlled, row-based playback methodology. The MIDI engine has extensive support for the playback of standard MIDI data streams, and has been optimized for this use paradigm. Unfortunately, this creates some problematic behaviors for an application such as Resonant Element. When being used on an iOS-based mobile device, for example, the row timings become unreliable if the device is "locked" or allowed to "sleep." This is a function of the fact that the devices attempt to save battery in these modes by changing the management approach for available active CPU time. There is no effective workaround for this shortcoming, which means Resonant Element can only work correctly while the application is active and the phone is "awake," which is not ideal for some of the use cases (such as for use as a sleep or meditation aid). Addressing this would require a re-architecture of the playback system itself.

5.2 Repetition Without Order

Resonant Element's use of an LRU cache for selecting generated sections that have already been used does produce a certain degree of repetition. However, subjectively this is not as effective as one might expect. In some cases, a single generated pattern is repeated too many times consecutively, which can evoke a feeling of the application being "stuck in a loop." In contrast, sometimes the system reuses patterns too infrequently, or with a degree of frequency that makes the reuse unclear to the listener, which can evoke a sense that the generation process is rudderless and overly chaotic. The underlying problem appears to be that there is no larger constraining structure dictating the use of preexisting sections. An earlier version of the system used a data structure called a "song flow" to organize this kind of repetition[4], but it was removed during further development. It may be necessary to revisit that approach.

6 Conclusions and Future Directions

The Resonant Element music generation system is an effective engine for the production of a continuous stream of musical output with reasonably good quality. The use of a combination of different stochastic models in a multi-layered generative process provides a viable organizational structure with which to generate sophisticated musical patterns. Section-level histograms have been shown to be a valuable and effective tool for creating the illusion of musical intention and raising the perceived sense of cohesiveness of the output. The use of an LRU cache to balance novelty and familiarity has shown promise, but has also revealed the limitations of pattern reuse without a higher-level guiding system of order and flow.

Future development work will focus on improving some of the identified limitations of the approach taken. Specifically, the playback system itself requires a deep re-imagining in order to achieve an acceptable level of reliable performance on the target operating systems and devices. This is likely to involve shifting from a proprietary note strip system to a more traditional MIDI stream structure. Additionally, the organizational structure that guides the pattern of repetition of generated sections needs to be revised to address the identified shortcomings.

Development of an organized methodology for the evaluation of creative outputs from systems similar to Resonant Element appears to hold promise for future research, as well. Though there has been some work in this area, such as the creation of the Standardised Procedure for Evaluating Creative Based Systems (SPECS)[13], this appears to be fertile ground for further exploration.

References

1. Wooller, R., Brown, A. R, et al. A framework for comparison of processes in algorithmic music systems. In: Generative Arts Practice, pp. 109–124. Creativity and Cognition Studios Press, Sydney (2005)
2. M. Zentner et al.: Emotions Evoked by the Sound of Music: Characterization, Classification, and Measurement. American Psychological Association, Emotion, 8(4), 494–521 (2008)
3. TheoryTabs: Famous Chord Progressions, `https://www.hooktheory.com/theorytab/common-chord-progressions`
4. Franklin, B.: SongSkill: A System for Continuous, Emotionally-Adaptive Music Generation. In: Proceedings of Generative Art 2016, pp. 125–137.
5. Gabrielle, A., Stromboli, E.: Music and Emotion: Theory and Research. Oxford University Press, Oxford (2001)
6. Vijayalakshmi, K., Sridhar, S., Khanwani, P.: Estimation of effects of alpha music on EEG components by time and frequency domain analysis. In: 2010 International Conference on Computer and Communication Engineering (ICCCE). IEEE (2010)
7. Haapakangas, A., Haka, M., Keskinen, E., Hongisto, V.: Effect of Speech Intelligibility on Task Performance- An Experimental Laboratory Study. In: Performance: 9th International Congress on Noise as a Public Health Problem (ICBEN) (2008)

8. Huang, R., Shih, Y.: Effects of background music on concentration of workers. Work 38, 383–387 (2011)
9. Parekh, R.: Principles of Multimedia. Tata McGraw-Hill, New York (2006)
10. Pereira, C. S., Teixeira, J., Figueiredo, P., Xavier, J., Castro, S. L., and Brattico, E.: Music and Emotions in the Brain: Familiarity Matters. PLoS ONE 6(11). (2011)
11. Ariza, C.: The Interrogator as Critic: The Turing Test and the Evaluation of Generative Music Systems. Computer Music Journal 33(2), 48–70 (2009)
12. Loughran, R., O'Neill, M.: Limitations from Assumptions in Generative Music Evaluation. Journal of Creative Music Systems, 2(1). (2017)
13. Jordanous, A.: A standardised procedure for evaluating creative systems: Computational creativity evaluation based on what it is to be creative. Cognitive Computation, 4(3). (2012)