TRoco: A generative algorithm using jazz music theory

Matthew Caren

Palo Alto, California
matt.t.caren@gmail.com

Abstract. This paper presents TRoco, a generative algorithm for the composition of music based on jazz music theory and driven by an input of the desired degree of musical tension over time. A method for the abstraction and analysis of musical structures based on jazz theory is detailed, as well as the application of this method in a generative algorithm that produces chord sequences with a desired tension-release contour. Also presented is an example implementation of TRoco, where the position of a user in a virtual environment is used to drive generation.

Keywords: algorithmic composition, generative algorithm, jazz music theory, tension and release

1 Introduction

There are many types of music theory originating from different cultures and types of music. In the creation of algorithms to generate music, however, classical Western music theory has been utilized more often, while other types, including jazz music theory, remain less explored. TRoco (Tension/Release-Oriented Composer) is a generative algorithm for the composition of music that uses the key concepts from jazz theory to create music to match a tension/release contour over time.

There are many examples of algorithms that use classical music theory rules to generate musical sequences. Lerdahl and Jackendoff (1983) is one well-known example of such a system, and there are many other examples similarly built on classical music theory rules (Loy, 1989; Robertson et al., 1998). Many algorithms also employ neural networks and other machine learning approaches to generate music (Eck & Schmidhuber, 2002; Liu et al., 2014). (It is worth noting that issues with these models can sometimes be improved with music theory rules (Jaques et al., 2017).)

Many methods of quantifying and mapping input have been used to generate music that elicits a desired reaction from a listener, including the valence/arousal plane (Wallis et al., 2008; Huang & Lin, 2013), and measures such as "scariness level" (Rutherford & Wiggins, 2002). The existence of "tension and release" in music—qualitatively described as a sense of rising stress or impending climax, followed by a feeling of relaxation or resolution (Farbood, 2012)—is an essential part of music that is closely interrelated with the experience of emotion

(Krumhansl, 1997; Margulis, 2005). Several generative algorithms use musical tension to drive generation (Nikolaidis et al., 2012; Herremans & Chew, 2017; Lerdahl & Jackendoff, 1983), though the development and evaluation of these algorithms are largely exclusive to Western classical music.

Just as composition students are taught music theory—a known set of effective techniques, abstractions, and guidelines—and are aware of this theory when they compose, it seems logical to arm a generative algorithm with this same knowledge. Jazz composers and improvisers are taught largely the same body of knowledge to use when creating music, and TRoco aims to utilize these same concepts in a generative algorithm.

2 Music Theory in TRoco

TRoco uses a jazz approach to music theory to inform generation. Jazz theory is not exclusive to jazz music; it is simply a flexible and powerful method of abstracting, analysing, creating, and communicating musical structures. It is important to note that, as stated in Levine (1995)'s foundational book on jazz theory, "There is no one single, all inclusive 'jazz theory'; however, that "there is a common thread of development in jazz theory, a thread that has evolved logically from the earliest days of jazz." This theory is related to but largely distinct from the classical approach to music theory. Basic jazz theory can be generalized and abstracted such that a few key concepts can be used to analyze very complex structures, and an additional benefit is that one state does not restrict the available choices for the next musical or emotional state, making it especially powerful when considering a wide variety of possible musical and emotional directions.

Many bodies of music theory have in common the concept of tension and release. There are many analyses and models of tension and release in classical music (Farbood, 2012, p. 390), but as noted by Farbood (2012) there are very few examples of similar research with non-classical music (Fredrickson & Coggiola, 2003; Rozin et al., 2004). However, despite the relative scarcity of research on tension and release in non-classical music theory, there are well-established techniques in jazz theory for creating and releasing tension. These methods are foundational to jazz composition and improvisation, and are actively used while creating music. TRoco utilizes the key elements of the system of jazz harmony taught specifically at the Grove School of Music and outlined by cornerstone jazz theory books such as (Levine, 1995; Harrison, 1995). This approach is similar to a jazz composer who draws upon their knowledge of jazz theory to create music.

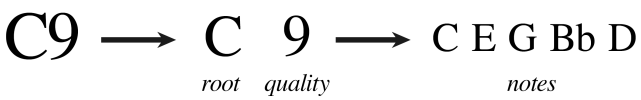

Fig. 1: Example Abstraction of a Chord

In TRoco, chords are represented by conventional jazz chord symbols, which can be decomposed into two components: a "root" and a "quality." The root of a chord is the tonal foundation of a chord and the note from which the rest of the chord is built. The chord quality determines the other notes in the chord relative to the root. As shown in Figure 1, in a "C9" chord (comprised of the notes [C E G B♭ D]), "C" is the root of the chord, and "9", which represents a dominant 9th chord, is the chord quality—the rest of the notes [E G B♭ D] are determined by the chord quality "9," relative to C. Chord qualities determine notes relative to a root; a "9" chord only represents the notes [C E G B♭ D] if the root is C. The same chord quality "9" in an "A9" chord denotes a different set of notes, [A C♯ E G B]. However, a chord quality often behaves in a similar way regardless of its root note, so this method of representation allows chords to be analyzed in terms of their functions rather than their manifestations. More generally, maintaining a level of abstraction ensures that elements retain musical meaning; it allows for the creation and analysis of musical structures rather than individual notes.

3 Algorithm Structure

3.1 TRQ, a "Tension/Release Quotient"

To use the concept of tension and release in TRoco, the methods from jazz theory to create and release tension must be rigorized and quantized. This is accomplished through the "tension/release quotient" (the "TRQ"): an integer between -10 and 10, representing the degree of tension or release imparted by the change from the current state to a possible next state, where -10 represents the maximum amount of release and 10 represents the maximum amount of tension.

A key consideration when evaluating tension/release is the existence of both "vertical" and "horizontal" significance of every musical element. Vertical significance refers to an element's relationship to other elements that are present at the given moment in time (including those held constant throughout an entire generation, such as a key), while horizontal significance refers to an element's relationship to elements present at a different time (an element's relationship to what preceded it). This approach supplies two axes of meaningful analysis of any structure. For instance, a higher degree of vertical chromaticism in a chord results in a greater, more tense, TRQ (thus a "♭9" in a chord, such as a D♭ in a C chord, would result in a more tense TRQ). Horizontal significance is also considered: for example, the interval between a chord's root and the previous chord's root (always calculated as an ascending interval) is considered—a 4th interval, for instance, would result in a more released TRQ.

The vertical factors that influence the TRQ are:
- chord quality
- chromaticism within a chord (i.e. through chord extensions and alterations)
- root note in/out of the key
- chord tones in/out of the key.

The horizontal factors that influence the TRQ are:
- root note interval
- notes in common with previous chord
- notes a half-step away from the previous chord (leading tones)
- chord quality of the previous chord in relation to the current chord.

The default TRQ is 0, and each vertical and horizontal consideration increases or decreases the value.

For instance, if the current chord is C major, the movement C major → A minor, which is diatonic, has a root note interval of a major 6th, and shares in common 2 chord tones with C major, has the slightly released TRQ of -2. The movement C major → A7♭9, however, has the same root note interval but is a dominant chord with a chromatic alteration (♭9) and two chord tones not in the key of C major, so it has a more tense TRQ of 4.

The input for TRoco is an array (for a generation of fixed length) or continuous stream (for a generation with unknown length) of TRQ values that represents the desired tension or release of the generation over time. Depending on the application, this tension/release profile can be obtained directly from the user or from another source—for instance, if TRoco is being used to generate music to accompany a video game, the events occurring in the game could be used to produce the profile.

3.2 Core Algorithm

The basic structure of the algorithm is displayed in Figure 2. Before generation starts, it is necessary to define the set of possible states that the generation could output. When generating chords, this is accomplished by specifying a domain of possible roots and chord qualities. For instance, a possible domain could include the root notes [C F G], and the chord qualities [major minor 7], yielding overall possible combinations of: C, C minor, C7, F, F minor, F7, G, G minor, and G7.

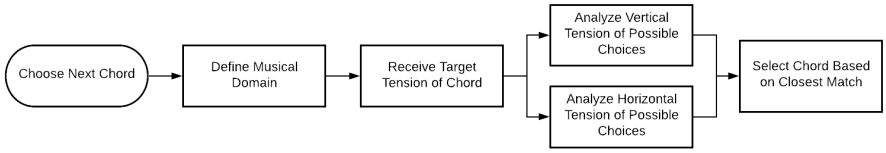

Fig. 2: Flowchart of algorithm structure

Whenever TRoco reaches a musical state, the TRQs of all possible next states are calculated, and the algorithm chooses the state with a TRQ that most closely matches the target profile. This state becomes the current state, and the process is repeated. The algorithm can be executed with multiple "threads," where several of the closest matches are selected at each stage of the algorithm, creating an

N-ary tree structure. At the end of generation, the best overall choice (evaluated by the sum of the deviations of the generations from the target) is selected. This results in generations that more closely match the target tension/release profile. With just a single thread, TRoco can be used for real-time generation of chords based on a TRQ input that changes in real time; for near-real-time generation, the algorithm can generate one or two states ahead and choose the best option. Since TRoco deals with the tension and release of state changes, not the actual states themselves, an initial state must be chosen at the beginning of generation.

```
Data: desired TRQ contour, intial chord, root domain, chord quality domain
Result: chords matching TRQ contour
GenerateChords
    foreach TRQ value in goal contour do
        goalTRQ = current TRQ value;
        last = last chord;
        currentBestFit = none;
        currentBestFitTRQ = infinity;
        foreach note in root domain do
            foreach quality in chord domain do
                thisChord = chord with current root, chord quality;
                if |FindTRQ(last, thisChord) − goalTRQ| <
                  |currentBestFitTRQ − goalTRQ| then
                    currentBestFit = thisChord;
                    currentBestFitTRQ = FindTRQ(last, thisChord);
                end
            end
        end
        append currentBestFit to chord array;
    end
    def FindTRQ(last, current):
        int TRQ = 0;
        TRQ += chord quality;
        TRQ += chromaticism within notes of current;
        TRQ += root of current in/out of key;
        TRQ += chord tones of current in/out of key;
        TRQ += root note interval last to current;
        TRQ += notes shared by last, current;
        TRQ += notes half-step away between last, current;
        TRQ += chord quality of last vs current;
        return TRQ
    end
end
```

Fig. 3: Pseudocode for basic algorithm

Figure 3 presents pseudocode outlining the basic algorithm for a single-thread system iterating over a TRQ contour. Note that the weightings of the various considerations within the *FindTRQ* function are subjective, and altering these values can provide different variations of generation types within the same general structure.

The TRoco code[1] is written in Javascript and utilizes the open source Tonal.js library for basic musical structures.

4 Example Implementation

TRoco was used to generate music for a simple video game[2] (Figure 4). In the game, a user is able to control the position of a character in a 3D environment that contains a skull (whose location is marked with a red sphere) and a treasure chest (marked with green sphere). The target TRQ is calculated every second by comparing the player's location to the location of the two spheres—the closer the player is to the red sphere, the higher the TRQ; the closer the player is to the green sphere, the lower the TRQ. Thus, the user being closer to the skull results in more tense chords being generated, while being closer to the chest results in a release of tension. The generated chords are passed to a series of arpeggiators that trigger samples to create the music heard in the game.

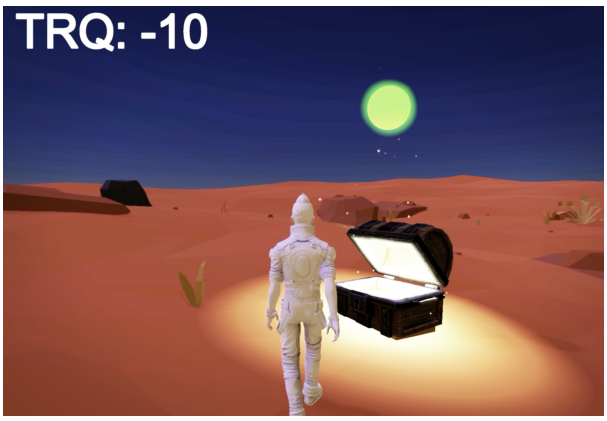

Fig. 4: Screenshot from example game

Table 1 displays the target TRQs and the generated chords in the recorded video. Note that chords tend to increase in dissonance and harmonic complexity with higher TRQs, as well as the presence of commonplace jazz figures, such as the harmonic resolution of the generation via a ii-V-I progression in the last three generated chords.

[1] `https://github.com/matthewcaren/troco`
[2] `https://youtu.be/1MMVgYcOZkw`

Target TRQ	Chord
–	Dm7
0	Am9
2	Fmaj7
3	Am9
5	B6
6	BmMaj7
9	FmMaj7
10	A7
7	F7
1	Dm7
-4	G6
-10	C6

Table 1: Example target TRQ input and generated chords in example

Though a straightforward example, this implementation displays how the TRoco and the TRQ input can be used simply and effectively to generate music that reacts to changes in a virtual environment in real-time.

5 Next Steps

TRoco's core tension/release-driven algorithm could be applied effectively to the generation of any musical structure, not just chords—the calculation of the TRQ would simply be adapted to calculate the tension or release imparted by each possible musical choice. If multiple musical structures are being generated (for instance, chords and melody), the tension or release of the individual components would be calculated, as well as the tension or release created by their coexistence.

The simplicity and flexibility of the single tension/release input allows TRoco to be adapted for a large variety of applications. Creators producing games, movies, installations, or VR experiences could use TRoco to create music that conforms to the intended tone. Using sentiment analysis, the emotional content of a text source could be used to calculate a TRQ over time, so the algorithm could be used to generate musical accompaniment for an online messaging conversation, e-book, or social network feed.

There still exists space to explore the use of contemporary jazz theory in composition algorithms, which could offer novel contributions to algorithms built on classical theory or provide a different perspective to inform the creation and analysis of machine-learning based models.

References

Eck, D., & Schmidhuber, J. (2002). Finding temporal structure in music: Blues improvisation with LSTM recurrent networks. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing* (p. 747-756).

Farbood, M. M. (2012). A parametric, temporal model of musical tension. *Music Perception*, *29*(4), 387–428.

Fredrickson, W. E., & Coggiola, J. C. (2003). A comparison of music majors' and nonmajors' perceptions of tension for two selections of jazz music. *Journal of Research in Music Education*, *51*(3), 259–270.

Harrison, M. (1995). *Contemporary music theory.* Milwaukee, Wisconsin: Hal Leonard Corporation.

Herremans, D., & Chew, E. (2017). Morpheus: generating structured music with constrained patterns and tension. *IEEE Transactions on Affective Computing*.

Huang, C.-F., & Lin, E.-J. (2013, June 11-15). An emotion-based method to perform algorithmic composition. In *Proceedings of the 3rd International Conference on Music & Emotion (ICME3).*

Jaques, N., Gu, S., Turner, R. E., & Eck, D. (2017). Tuning recurrent neural networks with reinforcement learning. *arXiv preprint arXiv:1611.02796*.

Krumhansl, C. L. (1997). An exploratory study of musical emotions and psychophysiology. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, *51*(4), 336.

Lerdahl, F., & Jackendoff, R. S. (1983). *A generative theory of tonal music.* Cambridge, MA: MIT press.

Levine, M. (1995). *The jazz theory book.* Petaluma, California: Sher Music Co.

Liu, I., Ramakrishnan, B., et al. (2014). Bach in 2014: Music composition with recurrent neural network. *arXiv preprint arXiv:1412.3191*.

Loy, G. (1989). Composing with computers: A survey of some compositional formalisms and music programming languages. In *Current Directions in Computer Music Research* (pp. 291–396).

Margulis, E. H. (2005). A model of melodic expectation. *Music Perception*, *22*(4), 663–714.

Nikolaidis, R., Walker, B., & Weinberg, G. (2012). Generative musical tension modeling and its application to dynamic sonification. *Computer Music Journal*, *36*(1), 55–64.

Robertson, J., de Quincey, A., Stapleford, T., & Wiggins, G. (1998). Real-time music generation for a virtual environment. In *Proceedings of ECAI-98 Workshop on AI/Alife and Entertainment.*

Rozin, A., Rozin, P., & Goldberg, E. (2004). The feeling of music past: How listeners remember musical affect. *Music Perception*, *22*(1), 15–39.

Rutherford, J., & Wiggins, G. (2002). An experiment in the automatic creation of music which has specific emotional content. In *Proceedings for the 7th International Conference on Music Perception and Cognition.*

Wallis, I., Ingalls, T., & Campana, E. (2008). Computer-generating emotional music: The design of an affective music algorithm. In *Proceedings - 11th International Conference on Digital Audio Effects, DAFx 2008* (pp. 7–12).