# Music Generation with Markov Chains

Patricia Tse

12/13/2021

Link to github repository: https://github.com/pktse/MarkovMusicGenerator

Link to survey questions: https://forms.gle/cgDKkcPECemZEf718

Link to survey results: https://tinyurl.com/55rhz7vv

Link to playlist of generated melodies (Check description of video): https://tinyurl.com/2kfhajes

## Introduction

Our perception of music is more than just reacting to a string of random notes. As we have learned in Music108 and through our readings of Huron's *Sweet Anticipation*, what we perceive as "pleasant" has a lot to do with our expectations when listening to a piece. For example, features like registral direction, intervallic difference, registral return, proximity, and closure can affect our expectancy of how a melody should continue—this can then affect how we feel about a piece because of the prediction effect (Huron, 2006). While composers may keep these things in mind, whether consciously or unconsciously, something like a computer may not.

For my final project, I implemented a Markov Model to generate music based on an inputted melody and evaluated if the generated melody could pass as a human-written melody primarily through survey participants' ratings. A Markov chain is a stochastic process, a sequence of random variables, where the probability of a given event happening depends on the previous event(s) (Moore, 2018; Shapiro, 2021). In mathematical terms, this can be written as: the probability of event j happening at time n + 1 given that event $i_n$ occurred at time n. This is the transition property Pr (i, j) which is shown in Equation 1.

$$\Pr(i, j) = \Pr[X_{n+1} = j \mid X_n = i_n] \tag{1}$$

A Markov chain has the freedom of modelling whatever we choose, whether that be the pitch of a note or the duration of the note. I used the inputted melody to generate the Markov chains and from there, chose the event with the highest probability for the next note, then generated a melody based on the probabilities of the inputted file.

## Method

I used the Midi Library in Java to analyze and generate my melody. With the data stored in the midi file, I was able to look at the specific pitch and duration of each note and store their numerical values to use later on. My method of getting the midi file was to manually transcribe right hand melodies from existing songs using MuseScore, a score writer software. I exported the MuseScore file as a midi file which I then inputted into my Java program. I then looped through each midiEvent in the inputted melody and generated Markov chains based on the note and the duration independent of each other. Using the Markov chains, I picked a likely pitch and duration independently given the notes that preceded it and created a melody with those chosen notes.

To implement the Markov chains, I created a wrapper class to take care of the abstraction of adding and getting the probability of events from our Markov model. At the core of it, I used a hashmap of a hashmap to store the transition states. The key of the outer hashmap was a list of previous events, the key of the inner hashmap was the predicted event, and the value of the inner hashmap was the probability of that event happening.

Originally, my plan was to build a Markov Model off the previous N events, also known as an N-order Markov chain. In mathematical terms, this can be written as: our probability of event at time n + 1 being j given that the event at time n is $i_n$, at time n - 1 is $i_{n-1}$, … at time n - N is $i_{n-N}$ which is shown in Equation 2 (Moore, 2018).

$$\Pr(i, j) = \Pr[X_{n+1} = j \mid X_n = i_n, \ldots, X_{n-N} = i_{n-N}] \tag{2}$$

While testing my program, I realized that with an inputted melody that was not long enough, the Markov Model had gaps between possible events. With the randomness that I added while generating notes, the program could generate events that never occurred in the original score and the melody would eventually spiral into a sequence of random notes. For example, I would try to find which note should come next given that the previous two notes were a C5 and a D5. If the sequence D5-C5 never occurred in the original score, then it would output a random note. This could lead to a cycle of randomness where the previous events sequence never occurs in the inputted score and was therefore never added to our Markov Model.

As a result, I had to create more Markov chains—specifically, I made N sets of Markov chains built of order N, … , 1. If I was unable to find a possible next note from the $N^{th}$-order Markov chain, I would then look at the $(N-1)^{th}$-order Markov chain and so forth. In the rest of this writeup, when I mention an $N^{th}$-order Markov chain, I am referring to these N sets of Markov chains that span the $N^{th}$-order Markov chain to the $1^{st}$-order Markov chain.

## Results

I must preface this by recognizing that there are sources of error in my method. First, because there is no way to directly convert an audio file to midi, I had to transcribe the melodies and cannot guarantee that these transcriptions are perfect. In addition, after generating the melodies, I had to translate the midiEvents into a score which may also have errors as well. The output of my code was comprised of multiple lines formatted similar to "PLAY: @120 key = 86 note = E" where the number following "@" is the duration of the note, the number following "key =" is the numerical value of the key, and letter following "note=" is the note letter. There is the chance that I may have repeated or missed a note, or misassigned a duration or key value to another note.

Using the program that I wrote, I generated three melodies using the 0th-order, 3rd-order, and 5th-order Markov chain for each inputted melody. Since I used three different input melodies, I totalled twelve melodies ( 3 x [3 generated melodies + 1 original melody]). For the first melody, I transcribed the right hand melody of Cornelius Gurlitt's *Sonatine F-Dur, op. 214, Nr. 2, Moderato* (hjk210, 2015; See Appendix A). For the second, I transcribed a portion of the right hand melody of Richard Wagner's *Entry of the Gods Into Valhalla* (TheWickedNorth, 2008; See Appendix B). For the third, I transcribed a portion of the right hand melody of Pyotr Ilyich Tchaikovsky's *Valse Sentimentale* (TheWickedNorth, 2009; See Appendix C).

Originally, I had also intended to include the time signature into the Markov chains that determined the duration of the note. By doing this, I would have ideally been able to decide on the durations of notes given how many beats are left in a measure as well as the previous events. This way, the program would not, for example, generate a half note when we only have enough room for an eighth note. Without having notes that end at the end of a measure or start at the start of a measure, we are unable to create the downbeat that we often use to build expectations about the piece (Roig, 2018; Huron, 2006). In addition, events happening at certain beats, like the downbeat and subbeats, are more probable and therefore evoke positive feelings because of the prediction effect (Huron, 2006). Unfortunately, I ran into some issues implementing that and did not have enough time to include it in my program.

## Analysis

I sent out a questionnaire for people to rate each melody on a scale of 0 to 10 where 0 was "this melody was definitely computer generated" and 10 was "this melody was definitely written by a human." I also had the participants explain their rating after rating the melodies. I received a total of twelve responses and used their responses to find trends and make observations (See Appendix D for the table of ratings, see link at top of report to get full table of responses with explanations).

Additionally, there could have been a confounding variable in the results that I did not consider. I failed to ask the participants about their musical background. In a few of the explanations of their ratings, participants referenced their experience with musical theory and/or their experience playing an instrument. In addition, as we had learned during lecture, there are multiple types of expectations: dynamic, schematic, veridical, etc. In particular, we often build expectations based on past experiences that we've had and what we've listened to previously (Huron, 2006). How a participant's response might be affected by the previous melodies was something that I was considering when deciding how to order the melodies in the survey. If I had put the original melodies first, that would have probably affected the ratings of subsequent melodies. In fact, for one participant's explanation of their rating, they said "as compared to the last one it is very mechanical.” Likewise, the ratings would also be different if I played all the melodies trained on Melody 1 back to back before moving to the melodies trained on Melody 2 or Melody 3. As a result, to mitigate the effect that previous melodies had on the current melody that the participant was listening to, I intentionally spaced out each melody such that the survey went in the order of melody trained on Melody 1, melody trained on Melody 2, melody trained on Melody 3.

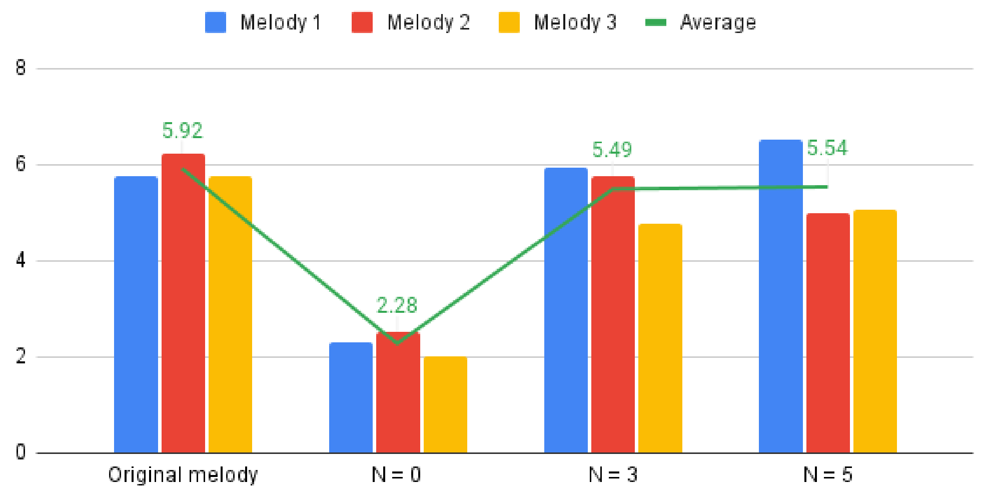


*Figure 1. Multiple box plots showing minimum, maximum, lower quartile, and upper quartile for each of the 12 melodies grouped by the inputted melodies.*

In Figure 1, the bar plot shows the average rating for each melody grouped by the order of the Markov chain and the overall average of each group (See Appendix E for the table used to generate this graph). On average, the original melody sounded the most human-written, followed closely by the melodies generated by the 5th-order, and then the 3rd-order Markov chain. While the average rating of the original melody and melodies generated by the 3rd-order and 5th-order Markov chains are close to 5, only the melodies generated by the 0th-order Markov chain are rated as distinctly more computer-generated sounding than human-written sounding.

A common explanation for the ratings of the 0th-order Markov chain was that the melody sounded very random and had strange jumps between high and low notes. This makes sense because listeners often don’t expect abrupt jumps since intervals “tend to be followed by pitches that continue in the same direction” (Huron, 2006, p. 77). At the same time, participants would give high ratings to other melodies because there were “more inflections” or there was “a lot of variation” and low numbers if a melody was “too simple” or “sound[ed] a little repetitive.” There seems to be an ideal balance between the amount of variation that keeps a melody interesting and the amount of order such that it does not seem random.

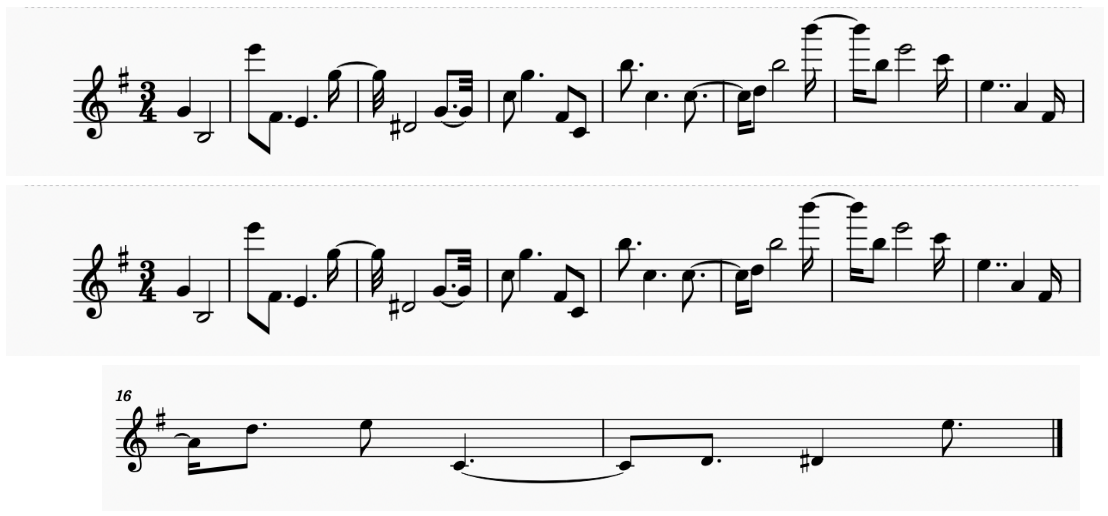


*Figure 2. Melody generated by 0th-Order Markov chain trained on a portion of Pyotr Ilyich Tchaikovsky’s Valse Sentimentale*

In addition, there is a “general tendency for phrases to rise upward and then descend in pitch, forming an arch-shaped contour,” and when we look at the scores of the generated melodies, the melodies generated by a 0th-order Markov Chain do not follow this trend (Huron, 2006, p. 86). An example is the melody generated by the 0th-Order Markov chain trained on a portion of Pyotr Ilyich Tchaikovsky's *Valse Sentimentale* shown in Figure 2. Instead of following an arch-shaped contour, the melody seems to zigzag, jumping between low and high notes.

Another common explanation for low ratings was that the ending didn't sound right. Participants would note that the "ending is strange," the "end sounded slightly random," the melody had an "abrupt ending," there was "no resolution," and the melody "did not resolve at the end." The resolution in a piece is a highly anticipated event, and when it fails to meet expectations, it "evoke[s] a negatively valenced prediction response" (Huron, 2006, p. 314). As a result, it's very common to have resolutions in music pieces and songs. As we listen to "normal" music, it helps build our schematic expectations about resolution. It then follows that if a melody does not match our expectations about the resolution, it may sound unnatural since we prefer stable tones during closures (Kristop, 2020). In fact, one participant noted that "there should have been one more note at the end and I'm annoyed there wasn't."

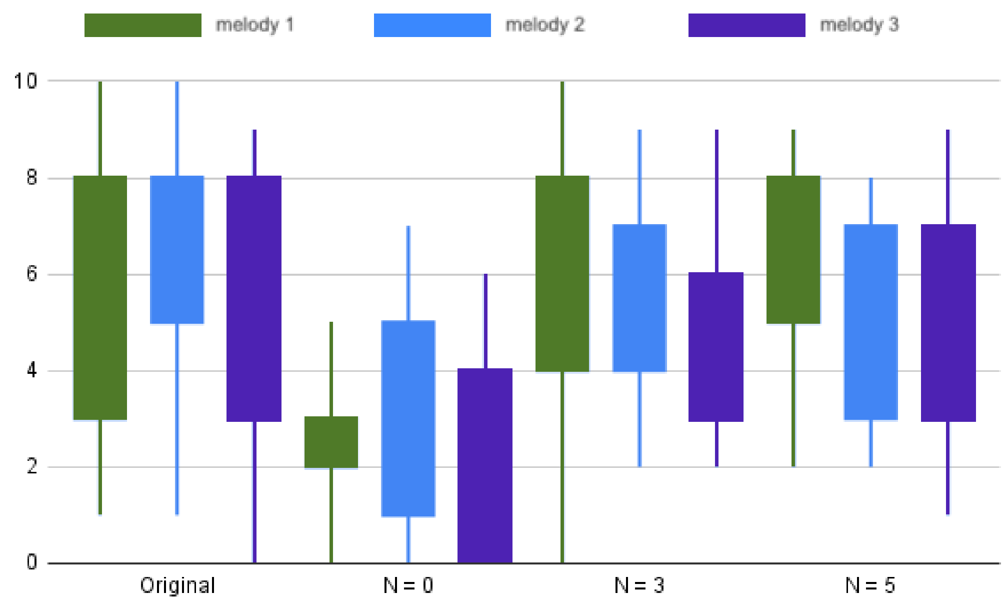


*Figure 3. Multiple box plots showing minimum, maximum, lower quartile, and upper quartile for each of the twelve melodies grouped by the inputted melodies.*

In Figure 3, I plotted the data points as a whisker and box plot to show the distribution and range of ratings for each melody (See Appendix F for the table). It shows that in general, the interquartile range for the original melodies tended to be larger than that of the generated melodies, meaning that there was a larger spread of the distribution of ratings. For a few of the explanations of the ratings of the original melodies, there was a trend where participants rated the melodies lower because there were "too many perfect sounding runs back to back," "sounded too clean and precise," or was "too simple." At the same time, there were participants who rated it highly for the same reasons.

Interestingly, some participants associated imperfection with human written melodies—participants gave low ratings for the original melody because it had "perfect sounding runs" and sounded "mechanical" but would give generated melodies higher ratings because the rhythm sounded "like a live piano" and had "the bit of complexity with change in rhythm and progression." The reason for this could perhaps be tied to microtiming. Microtiming variations are timing differences that help "evoke different kinds of rhythmic qualities, such as apparent accents or emotional mood, by playing notes slightly late or early relative to their theoretical metric location" (Mann, 2008). While the notes of the generated melodies would not be considered microtiming since the timing differences do not fall below the threshold of musical notation, the effect is similar—the imperfect rhythm helps establish a groove that sounds realistic. That being said, another participant noted that the melody "sounded unique in note length variation but not quite intentionally so." It seems that, similar to pitch, note duration must balance variation and order such that it has variety in its note durations but not to a point where it no longer sounds intentional.

## Conclusion

Overall, the program surprised me with its ability to fool participants into thinking that the generated melody could be written by a human. Simply by looking at the history of notes and with a high enough $N^{th}$-order, the program was able to generate a likely sequence of notes that on average seemed more human-written than computer-generated. Though it could not generate a melody that would on average be guessed as being written by a human, it was able to trick participants into not immediately thinking that it was written by a computer.

Currently, there seems to be a trend of improving average ratings for increasing values of N. If I had more time, I would have wanted to see if there is a point where a high enough N would result in a decrease in improvement. Likewise, I would have wanted to explore the question of  whether it is possible for computer generated melody to rank higher than a human written melody on average. I would have also connected the Markov chains that chose the note duration with the Markov chains that selected the note value so that the chosen note affects the note duration. I would have also tried running my program with different inputted melodies to see if certain pieces are more predictable than others as currently, all the inputted melodies I chose were classical pieces. Perhaps, a Markov chain could be the next Bach.

## References

hjk210. (2015, September 5). *Cornelius Gurlitt: Sonatine F-Dur, op. 214, Nr. 2*. YouTube. Retrieved December 4, 2021, from https://www.youtube.com/watch?v=nwW-XNHhP-c.

Huron. (2006). *Sweet Anticipation: Music and the Psychology of Expectation*. The MIT Press. https://doi.org/10.7551/mitpress/6575.001.0001

Kristop, Moreno, S. J., & Anta, J. F. (2020). What do listeners understand by "continuity" and "closure"? Tracking down the links between tonal expectancy, music training, and conceptualization. *Psychology of Music*, 48(3), 344–357. https://doi.org/10.1177/0305735618803000

Mann, Yotam. (2008, February 4). *Microtiming Studies*. CNMAT. Retrieved December 4, 2021, from https://cnmat.berkeley.edu/content/6-microtiming-studies.

Moore, Corrêa, D. C., & Small, M. (2018). Is Bach's brain a Markov chain? Recurrence quantification to assess Markov order for short, symbolic, musical compositions. *Chaos (Woodbury, N.Y.), 28(8)*, 085715–085715. https://doi.org/10.1063/1.5024814

Oracle. (n.d.). *Overview of the MIDI package*. Overview of the MIDI Package (The Java™ Tutorials > Sound). Retrieved November 30, 2021, from https://docs.oracle.com/javase/tutorial/sound/overview-MIDI.html.

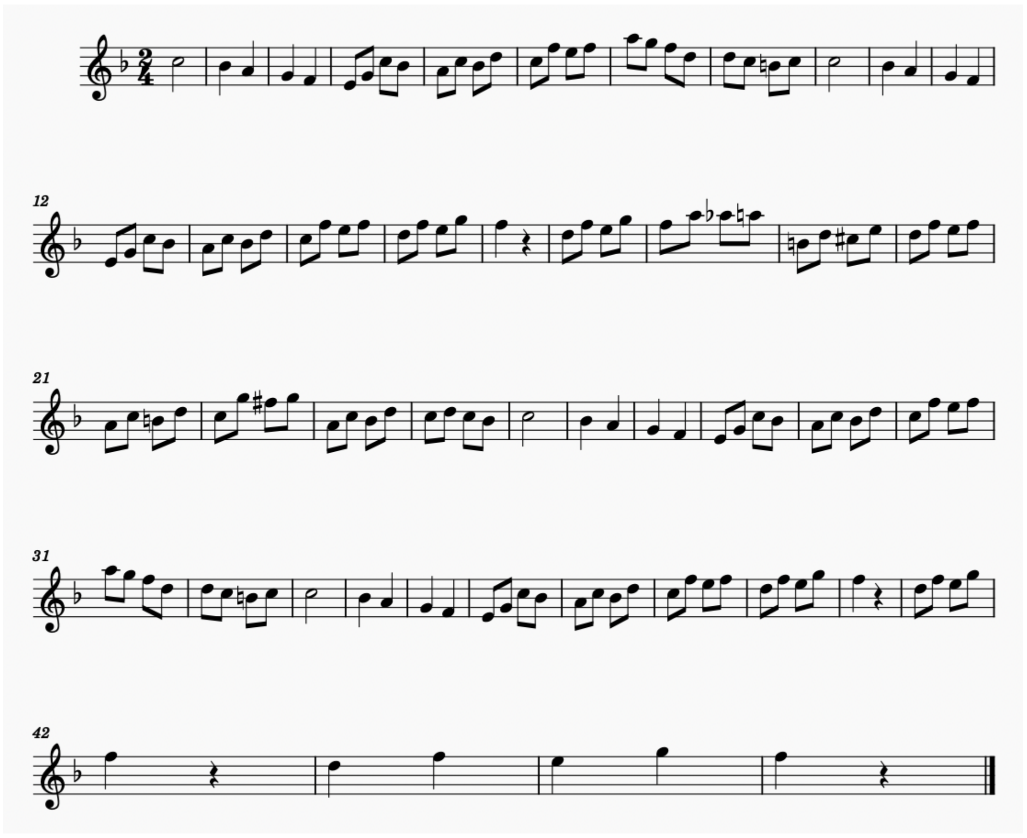
Roig, Tardón, L. J., Barbancho, I., & Barbancho, A. M. (2018). A non-homogeneous beat-based harmony Markov model. *Knowledge-Based Systems*, 142, 85–94. https://doi.org/10.1016/j.knosys.2017.11.027

Shapiro, & Huber, M. (2021). Markov Chains for Computer Music Generation. *Journal of Humanistic Mathematics*, 11(2), 167–195. https://doi.org/10.5642/jhummath.202102.08

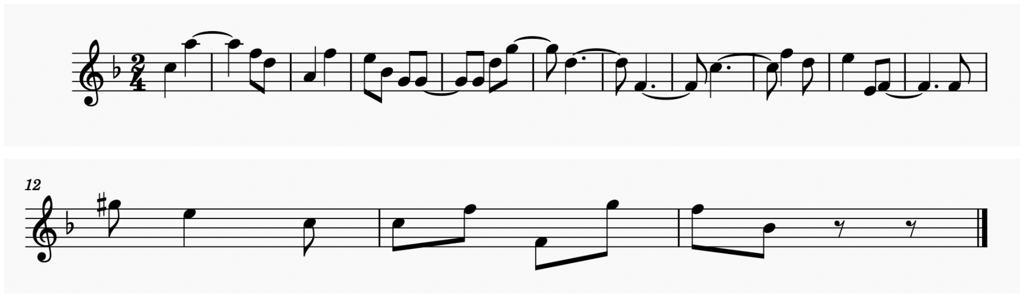
TheWickedNorth. (2008, December 31). *Wagner - Das Rheingold - entry of the gods into Valhalla*. YouTube. Retrieved December 4, 2021, from https://www.youtube.com/watch?v=b80Jw8MuZxo.

TheWickedNorth. (2009, February 11). *Tchaikovsky - Valse sentimentale*. YouTube. Retrieved December 4, 2021, from https://www.youtube.com/watch?v=rUuusqy50yk.

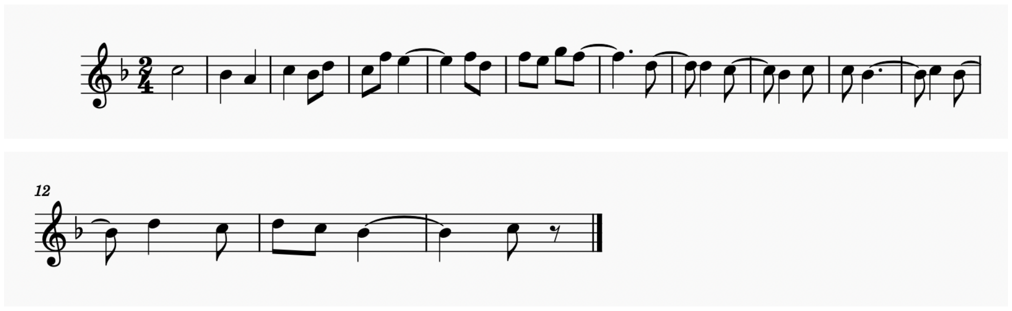
## Appendix A



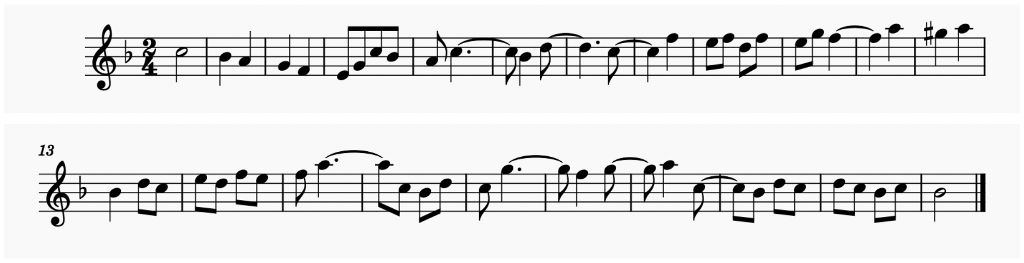
Transcription of Cornelius Gurlitt's *Sonatine F-Dur, op. 214, Nr. 2, Moderato*



Melody generated by 0th-Order Markov chain trained on Cornelius Gurlitt's *Sonatine F-Dur, op. 214, Nr. 2, Moderato*

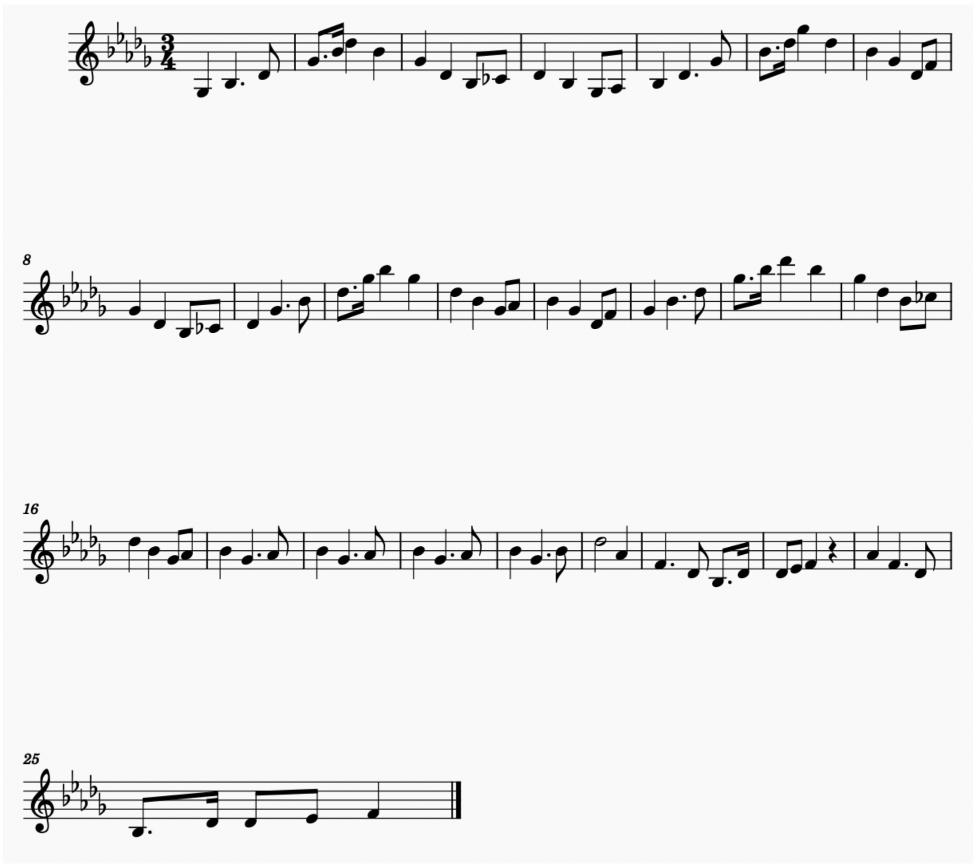


Melody generated by 3rd-Order Markov chain trained on Cornelius Gurlitt's *Sonatine F-Dur, op. 214, Nr. 2, Moderato*

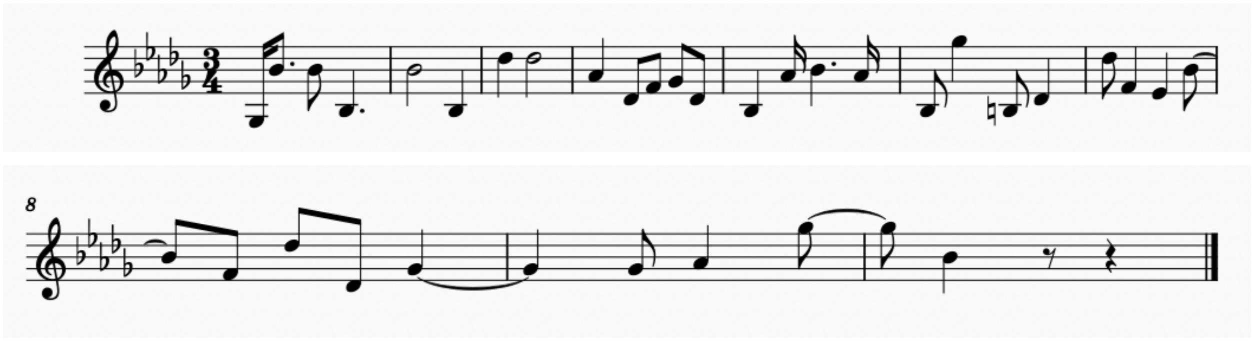


Melody generated by 5th-Order Markov chain trained on Cornelius Gurlitt's *Sonatine F-Dur, op. 214, Nr. 2, Moderato*

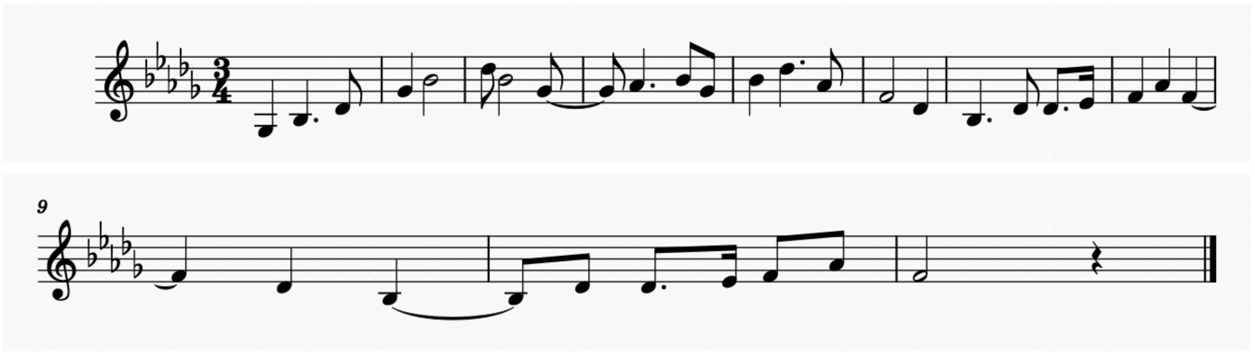
## Appendix B



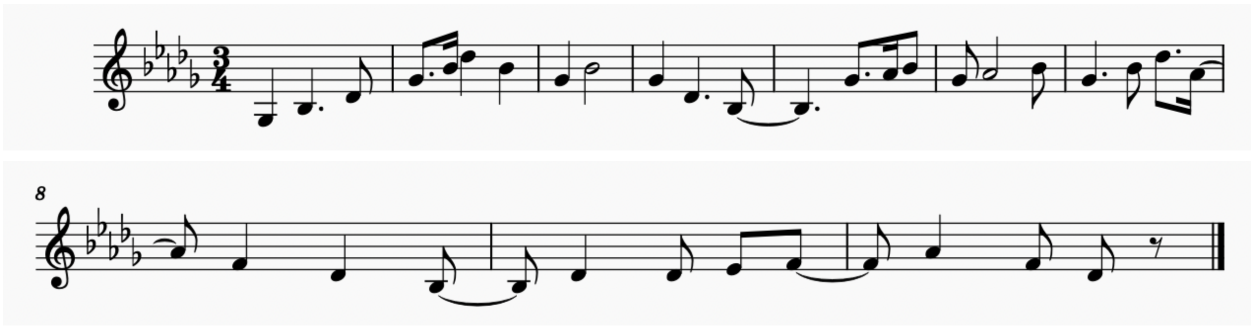
Transcription of part of Richard Wagner's *Entry of the Gods Into Valhalla*



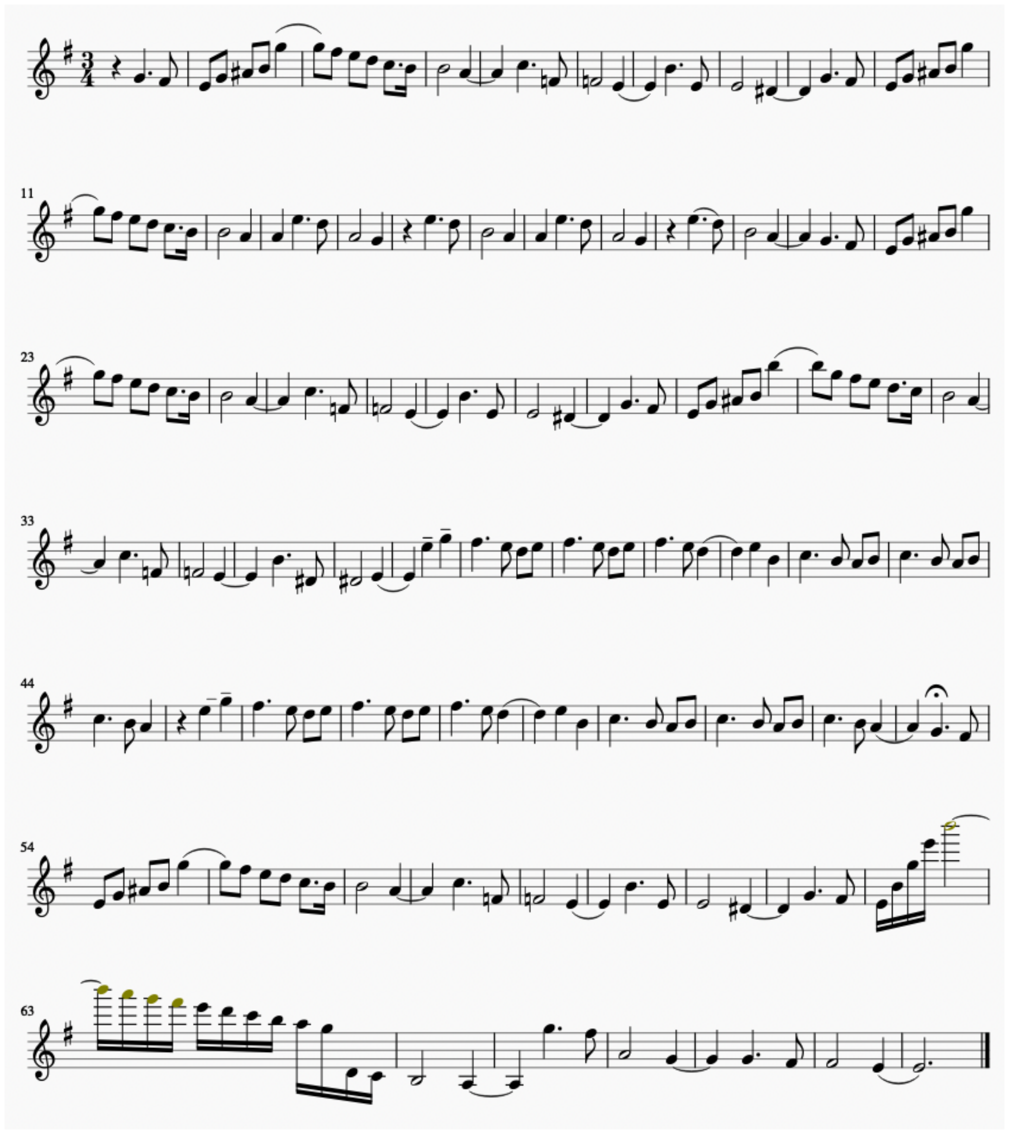
Melody generated by 0th-Order Markov chain trained on a portion of Richard Wagner's *Entry of the Gods Into Valhalla*



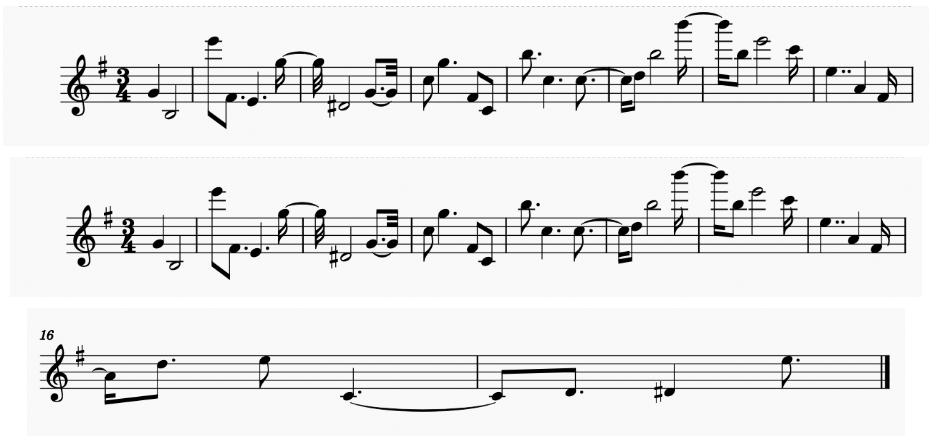
Melody generated by 3rd-Order Markov chain trained on a portion of Richard Wagner's *Entry of the Gods Into Valhalla*


Melody generated by 5th-Order Markov chain trained on a portion of Richard Wagner's *Entry of the Gods Into Valhalla*

Appendix C

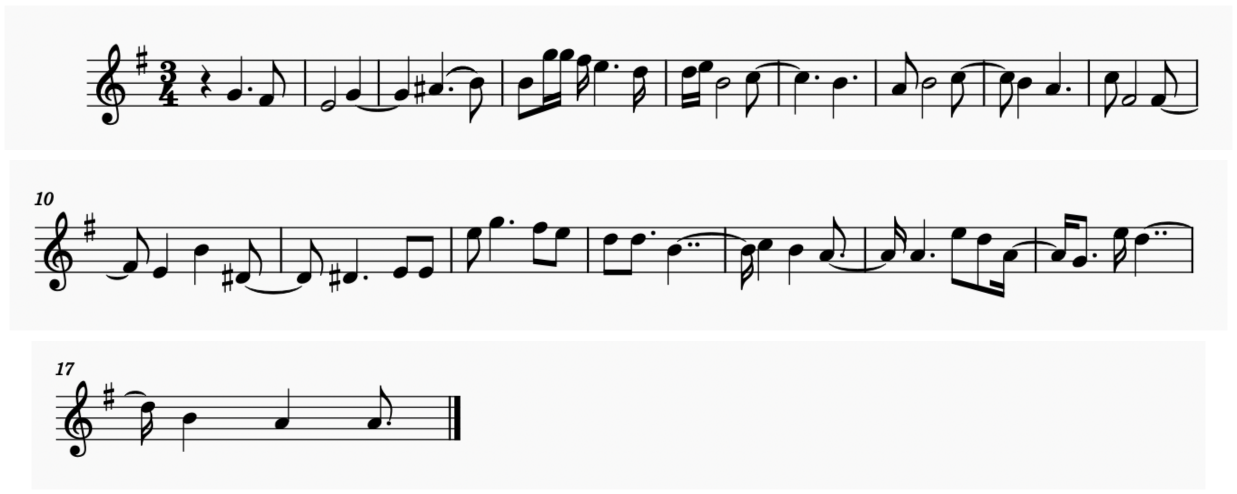


Transcription of a portion of Pyotr Ilyich Tchaikovsky’s *Valse Sentimentale*



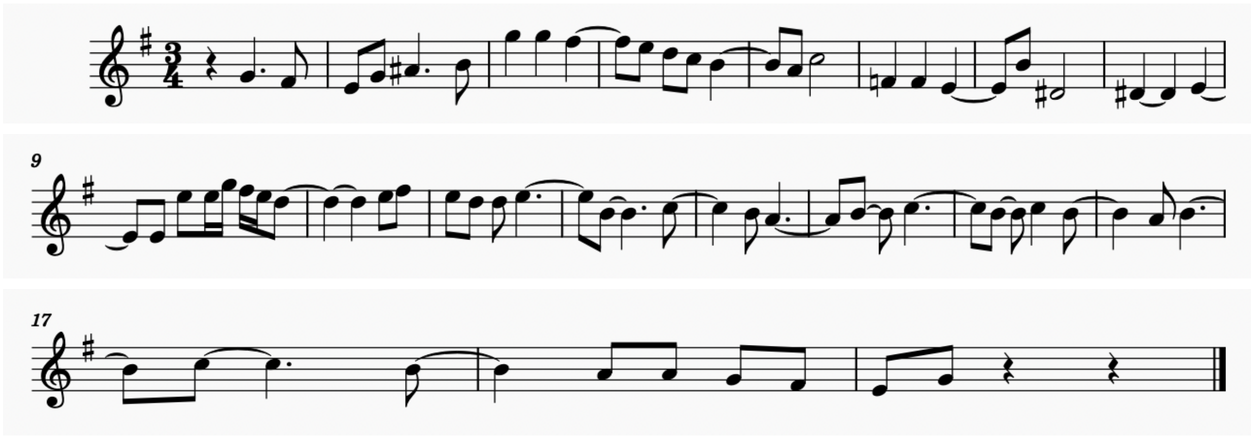
Melody generated by 0th-Order Markov chain trained on a portion of Pyotr Ilyich Tchaikovsky's

*Valse Sentimentale*



Melody generated by 3rd-Order Markov chain trained on a portion of Pyotr Ilyich Tchaikovsky's

*Valse Sentimentale*



Melody generated by 5th-Order Markov chain trained on a portion of Pyotr Ilyich Tchaikovsky's

*Valse Sentimentale*

## Appendix D

| Melody 1, Order 0 | Melody 2, Order 3 | Melody 3, Order 3 | Melody 1, Order 3 | Melody 2, Order 5 | Melody 3, Order 5 | Melody 1, Order 5 | Melody 2, Original | Melody 3, Original | Melody 1, Original | Melody 2, Order 0 | Melody 3, Order 3 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 8 | 7 | 7 | 8 | 6 | 8 | 10 | 8 | 10 | 2 | 1 |
| 3 | 7 | 3 | 10 | 5 | 8 | 8 | 6 | 3 | 3 | 7 | 3 |
| 2 | 9 | 6 | 3 | 8 | 5 | 9 | 7 | 3 | 3 | 7 | 6 |
| 5 | 5 | 2 | 5 | 2 | 4 | 3 | 7 | 7 | 8 | 1 | 0 |
| 0 | 2 | 3 | 5 | 4 | 3 | 4 | 8 | 5 | 10 | 5 | 0 |
| 3 | 2 | 6 | 8 | 4 | 6 | 7 | 4 | 9 | 8 | 1 | 0 |
| 3 | 7 | 4 | 10 | 3 | 3 | 9 | 1 | 8 | 5 | 1 | 0 |
| 3 | 3 | 5 | 0 | 7 | 1 | 7 | 10 | 0 | 3 | 2 | 5 |
| 3 | 7 | 2 | 7 | 7 | 4 | 2 | 5 | 9 | 5 | 2 | 6 |
| 0 | 5 | 2 | 3 | 3 | 7 | 7 | 5 | 7 | 4 | 0 | 0 |
| 1 | 4 | 7 | 8 | 3 | 7 | 5 | 3 | 8 | 9 | 5 | 4 |
| 2 | 9 | 6 | 4 | 4 | 3 | 8 | 6 | 2 | 1 | 0 | 0 |
| 2 | 7 | 9 | 7 | 7 | 9 | 8 | 9 | 6 | 6 | 0 | 1 |

Table of recorded ratings for how likely a melody was generated by a human from a scale of 0 to 10 where 0 was “definitely computer generated melody” and 10 was “definitely human written melody”

## Appendix E

| | Melody 1 | Melody 2 | Melody 3 | Average |
|---:|---:|---:|---:|---:|
| Original melody | 5.769230769 | 6.230769231 | 5.769230769 | 5.92 |
| N = 0 | 2.307692308 | 2.538461538 | 2 | 2.28 |
| N = 3 | 5.923076923 | 5.769230769 | 4.769230769 | 5.49 |
| N = 5 | 6.538461538 | 5 | 5.076923077 | 5.54 |

Table of calculated data points used to create the bar chart that summarized the average for each order markov chain for a specific melody compared to the average of other melodies and overall average

## Appendix F

| | | minimum | lower quartile | median | upper quartile | max |
|---|---|---|---|---|---|---|
| Original | Melody 1 | 1 | 3 | 5 | 8 | 10 |
| Original | Melody 2 | 1 | 5 | 6 | 8 | 10 |
| Original | Melody 3 | 0 | 3 | 7 | 8 | 9 |
| N = 1 | Melody 1 | 0 | 2 | 3 | 3 | 5 |
| N = 1 | Melody 2 | 0 | 1 | 2 | 5 | 7 |
| N = 1 | Melody 3 | 0 | 0 | 1 | 4 | 6 |
| N = 3 | Melody 1 | 0 | 4 | 7 | 8 | 10 |
| N = 3 | Melody 2 | 2 | 4 | 7 | 7 | 9 |
| N = 3 | Melody 3 | 2 | 3 | 5 | 6 | 9 |
| N = 5 | Melody 1 | 2 | 5 | 7 | 8 | 9 |
| N = 5 | Melody 2 | 2 | 3 | 4 | 7 | 8 |
| N = 5 | Melody 3 | 1 | 3 | 5 | 7 | 9 |

Table of calculated data points used to create a box and whisker plot for each melody’s 4 generated melodies.