Using Markov Chains to Generate Melodies

Trina Aguilana, Wystan Concepcion, Luigi del Rosario

I. Introduction

The rise of music technology has brought about more ways for musicians to express themselves. Automatic melody generation can be used to give musicians a boost in their songwriting process. However, simply generating random strings of notes with no basis will not yield realistic results. One possible solution is to use Markov chains.

A Markov chain is a mathematical system that consists of a set of *states*, as well as a *transition table* which contains the probabilities for jumping from one state to another [2]. By using musical notes as states, it is possible to use this system to generate more realistic melodies. Transition tables could be generated from already-existing melodies, as is done by Lin in 2016 [3] and by Wadi in 2012 [4].

The group shows how Markov chains could be used to produce more realistic and contextual melodies as compared with a completely random melody generator. Different transition tables were generated using three common songs as input data (namely “Happy Birthday,” “Mary Had a Little Lamb,” and “Twinkle Twinkle Little Star”). A program was coded in C to generate sample melodies from these transition tables.

II. Related Studies

Melody generation with Markov Chains. Alvin Lin from New York City has already tried creating music using Markov chains [3]. He generated a transition table based on Yiruma’s piano piece entitled “River Flows In You”. Using this data, he created a Python code that generated melodies using the given transition table. However, Lin takes into account notes from both the left hand (rhythm section) and the right hand (melody section). This means that the transition table generated was quite large. Hence, the output of the program Lin made had multiple jumps from lower to higher octaves, which resulted in melodic phrases that are not as musical. The group is only limiting itself to simple songs such as “Happy Birthday,” hence less notes will be taken into account, and the transition tables generated will be smaller than that of Lin.

Musical analysis with Markov Chains. The work of David M. Franz shows the use of Markov chains to analyze jazz pieces [1]. By generating transition tables from John Coltrane’s “Giant Steps” and analyzing the data using already-established methods of musical analysis, Franz showed that Markov chains can provide useful information in the analysis of jazz pieces. Although the group will not be analyzing creativity and style in the songs chosen, it is beneficial to know the depth of the topic chosen.

Similarly, Ala'a Wadi from the John Carroll University showed in 2012 how Markov chains could be used to determine copyright infringement between two songs [4]. Wadi also devised two different algorithms in determining the degree of difference between two songs. Wadi tested these algorithms with two pairs of songs, and concluded that with a larger sample size, a threshold could be found to determine whether a song was plagiarized or not.

Although the group will not be focusing on the latter part of Wadi’s work, a lot of inspiration may be drawn from the methodology used. These resources may aid the group in creating simple transition tables from common melodies, and execute the generated tables through code. The group may also take inspiration from the code Lin made in order to automate the process of melody generation.

III. Methodology

Transition tables for single-order Markov chains were constructed from the tunes of “Happy Birthday”, “Twinkle Twinkle Little Star”, and “Mary Had a Little Lamb”. Wadi’s work already features a detailed transition table construction for the first stanza of “Happy Birthday” [4]. The group will use this method in order to construct the other tables (and to extend Wadi’s table to account for the entire song). First, the amount of transitions from one note to another were listed down. These numbers were then put into a matrix, with each row representing the source note, and each column representing the destination note, all sorted from A to B, as shown in Figure 1:

Figure 1. Initial transition matrix [4]

$$FirstStanzaOfHappyBirthday := \begin{bmatrix} 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 0&0&3&2&0&1&1 \\ 0&0&2&0&0&0&0 \\ 0&0&1&0&0&0&0 \\ 0&0&1&0&1&0&0 \\ 0&0&0&0&0&1&0 \end{bmatrix}$$

In order for the matrix to consist of probabilities, each row was normalized as shown in Figure 2:

Figure 2. Normalized transition matrix [4]

$$FirstStanzaOfHappyBirthday := \begin{bmatrix} 0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0 \\ 0&0&\frac{3}{7}&\frac{2}{7}&0&\frac{1}{7}&\frac{1}{7} \\ 0&0&\frac{2}{2}&0&0&0&0 \\ 0&0&\frac{1}{1}&0&0&0&0 \\ 0&0&\frac{1}{1}&0&\frac{1}{1}&0&0 \\ 0&0&0&0&0&\frac{1}{1}&0 \end{bmatrix}$$

The process was repeated for the other two melodies, as well as the rest of “Happy Birthday.” Moreover, another “note” which represents the end of the song was added to the table, in order for the chain to be able to produce melodies of varying lengths. Another “hybrid” transition table was made from all 3 tunes combined; that is, the transition tables of all the songs were tabulated into one. Finally, a transition table consisting of a single value for every cell was generated and used as the *Random Melody Generator (RMG).*

In order to see how these matrices generate melodies as compared with the RMG, a C program was made to automate the melody generation process, given a transition table.

IV. Results and Conclusions

The group has constructed a total of four different transition matrices, with the hybrid transition matrix shown in Figure 3. Each row and column is arranged in the order of the C Major scale (C-D-E-F-G-A-B), with the 8th column signifying the end of the song.

Figure 3. Hybrid transition matrix

$$\begin{bmatrix} 1/5 & 1/5 & 0 & 0 & 4/15 & 0 & 2/15 & 1/5 \\ 4/9 & 5/18 & 2/9 & 0 & 1/18 & 0 & 0 & 0 \\ 2/21 & 3/7 & 3/7 & 0 & 1/21 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1/2 & 0 & 0 & 0 & 0 \\ 1/21 & 1/21 & 2/21 & 4/21 & 3/7 & 4/21 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3/7 & 4/7 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 & 0 & 0 \end{bmatrix}$$

A program was made in C in order to automate the process of melody generation. The program takes the starting note as an input, and generates a melody based on the specified transition matrix. A total of 5 programs were compiled. However, because some notes were nonexistent in some melodies, their corresponding matrices had to be edited in such a way that if the user inputted such a note, it would jump to the end state immediately.

Each program was run 7 times with varying starting notes, which gives a total of 35 automatically-generated melodies. Some results are shown in the table below.

Table 1: Sample results

Chain Origin	Sample Melody
Random Melody Generator (RMG)	E B G F F B B F F B
Happy Birthday	D C B A F F E C B

	G C C
Mary Had a Little Lamb	G E D C D E D C D E D C D C D C D E D E D E D E E D E D E E E E G G G E D C D C
Twinkle Twinkle Little Star	C G G G F F E D G G G F E D C
All 3 Melodies	E D C G D C G G G A G E D D C C G E E C G G G E D C B A A G G G F E E E D E E E E E E D C

The melodies generated by the Markov chains seemed to have more of a contextual flow as compared with those made by the RMG. Many visible differences between the two types of melodies could be seen at first glance.

The source melodies always ended with the root note (which is, in this case, C), and as a result, each generated melody ended with the same note, except for those whose starting state does not exist in the chain. The RMG ended with notes other than the root note; notice how the it ended with a B in the sample table.

The melodies generated by the Markov chains also avoided the 7th note of the scale (which is B), and circled around more of the commonly-used notes in the scale (C, D, E, G, and F). The 6th note (A) is hit more frequently than the 7th. The Markov chains also generated longer melodies more frequently than the RMG; similarly, the Markov chains also produced melodies that repeated the same note multiple times. Moreover, the hybrid Markov chain was able to generate contextual melodies based on any starting note.

This shows that the use of Markov chains is one possible way to generate realistic-sounding melodies. More complex transition matrices could be generated from more complex pieces of music. This type of model could be extended to involve chord patterns and note length, and could possibly be used to generate rhythmic sections, and eventually, even full band/orchestra arrangements. Given more training data, it might also be possible to generate a “universal” transition table from different pieces of music. Filtering out the sample data to only involve specific genres or styles could also lead to Markov chains that could generate specific kinds of music.

References

[1] Franz, D. (23 April, 1998). Markov Chains as Tools for Jazz Improvisation Analysis (published Masters thesis). Virginia Polytechnic Institute and State University, Virginia, United States.

[2] Grinstead, C., & Snell, J. L. (1997). Markov Chains. Introduction to probability (pp. 405-442). Rhode Island: American Mathematical Society.

[3] Lin, A. (01 December, 2016). Generating music using Markov chains. Retrieved from https://medium.com/@omgimanerd/generating-music-using-markov-chains-40c3f3f46405

[4] Wadi, A. (2012). Analysis of of music note patterns via Markov chains (senior honors project). John Carroll University, Ohio, USA. Retrieved from http://collected.jcu.edu/honorspapers