definition

June 7, 2023

1 Introduction

The hearing of a music can be seen as a complex procedure of recognition. When we hear a piece of music, we can recognize the genre, characteristic, and style traits of its composer. Sometimes, we can immediately recognize the similarity between two pieces of music, be it the reference from one to another , same chord relation, or a signature musical device being used. In this project, we will make an attempt to examine the similarity between two pieces of music using various metrics ranging from the cosine similarity to DFT. The project will survey on the existing methods of music similarity proposed by music theorists Dimitri Tymoczko, David Lewin, and Jason Yust. As a related topic, the project has also investigated the key-finding algorithm orginally proposed by Krumhansl and Schmuckler, modified by David Temperley. The final section of the report will be questions raised in the progress of reviewing these works and implementing the algorithms.

2 Cosine Similarity

Cosine similarity measures the cosine value of the angle between two vectors. Given two melody, we can represent them as vectors of their pitch classes. (in set or collection or ordered list) The angle decreases as the cosine value gets closer to 1. The equation can be written as:

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{|\mathbf{A}| \cdot |\mathbf{B}|}$$

where $\mathbf{A}$ and $\mathbf{B}$ are two vectors. The intuition behind this is that the more notes in the melody are the same, the smaller the angle between the two

vectors represented by the melodies. The cosine similarity is a good measure of similarity between two melodies when it comes to the overall "contour" of the melody. Given two melodies, one being Transposition of the other, will still have the same cosine similarity. However, it does not measure anything harmonic wise such as the scale the melody is in, or the chord progression implied when there are multiple voices in the music. So we cannot simply put all the notes in the melody into a vector and calculate the cosine distance between them.

Key Finding Algorithm

Krumhansl-Schmuckler Key Finding Algorithm The notion of key in tonal music can provide critical information of a music. Using the total duration of each pitch class in a piece of music, we can calculate the probability of each key being the key of the piece. We shall construct a profile of each pitch class in major and minor keys then find the corrilation coeficient between the profile and the pitch class distribution of the piece of music. It measure the linear correlation between the profile value and the duraiton for each pitch class. The key with the highest correlation is the key of the piece of music. The algorithm is as follows:

$$R(x,y) = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 \sum_i (y_i - \bar{y})^2}}$$

where x is the profile of the pitch class distribution of keys, and y is the duration of each pitch class in the piece. Code can be found https://github.com/StefanHeng/Symbolic-Music-Generation/blob/master/musicnlp/preprocess/key$finder.pyhere.Note, thecodeusesTemper Kosta-Payneprofilehereaftertrialanderrors.However, thereareafewproblemswiththisalgorith$

The notes are "weighted" by duration, which means repeated notes would take on more weight which might alter the final decision.

It does not take into the account of modulation between different keys.

The original key-profile model did not distinguish between different spelling fo the same pitch (A#=Gb).

Temperley Key Finding Algorithm David Temperley proposed a new key finding algorrithm to address these problems suing a Bayesian approach.

The algorithm is as follows:

$$\arg \max_{strcuture} \Pr[strcuture|surface]$$

. Since $\Pr[surface]$ stays the same, we only need to concider $\Pr[surface|strcuture] \cdot strcuture$. To calculate the second term, we can design the following heuristics: Start off with a probability of $\frac{1}{24}$ for every key. Then assign a very high probability to the same key meaning the composer will likely continue compose in the key, but a low probability to any other key. Here Temperley uses 0.8 for the same key and $\frac{0.2}{24}$ for the rest. Bt intuitively, we want the rest 0.2 probability to be distributed according to likelyness of closely related key such as modulation or dominant. So a furhter discussion can be made on how to distribute the probability. As for the first term, Temperley porposed a analyzing that first involves in segmentaiton of the piece:

1. Partition the piece into segments of arbiturary length.
2. Given a key-profile that contain probability of certain pitch class played in the (major and minor) key, take the product of the present pitch and the probability of the all absent pitch classes, which is simply $1-$ the probability of its presence in the segment. Written as:

$$\prod_{p} S(p) \prod_{\neg p} S(\neg p)$$

Thus we get: $\prod_m \Pr[currentkeyofsegmenm|keyofthelastsegment] \cdot \prod_p S(p) \prod_{\neg p} S(\neg p)$. We shall take the logarithm for a better numerical stability. The final equation is:

$$\sum_{m} (\ln \Pr[m|m'] \sum_{p} \ln S(p) \sum_{\neg p} \ln S(\neg p))$$

A couple questions should be raised here:

- How to determine the length of the segment?
- Should every notes in the segment be weighted equally?

Still, the algorithm discussed above is suitable only for a piece that is considered to have "one piece". Some times, the composer might trick the audience by writing multiple voices in different keys. For example the left hand of Bartok's Bagatelles' No1 is in C Locrian, while the right hand is entrily in E Major. So we need to take into account of different voices as well.

Similarity in terms of Voice Leading and Discrete Fourier Transform

Like cosine similarity, we might want to talk about the similiarity between two set classes in terms of the difference between them, or the minimum movement required for one to become the other, this is called voice leading. Dimitri Tymoszko showed a new method of analyzing the voice leading distance using Discrete Fourier Transform. The n-th Fourier component is (probably) inversly related to perfectly even n note set class (chord).

Voice Leading Distance Between two n-note set classes A and B, the minimal Euclidean voice leading distance can be founded as such:

1. Find the prime form of A, the sum the its pitch classes is m.
2. Find n transposition of B that sum up to m.
3. For each transposition, find the minimal l_2 distance between prome form of A and circular permutations (permuation with same ordering) of transpositions. Do the same for all n inversions of B (?).
4. The minimal distance within $2n^2$ numbers is the voice leading distance.

Discrete Fourier Transform The DFT of a set class can be written as:

$$\sum_p (\cos 2\pi pn/12, \sin 2\pi pn/12)$$

. For $n \in \{1, 2, 3, 4, 5, 6\}$, each nth Fourier component is the sum above. The paper has showed, DFT converts the set class in a "reduced" circle with $12/n$ circumference. Each pitch in the set class is represented by a point on the circle. The sum would be vector sum of all the points. If the pitch class lies closely to the set represneted by the perfectly even n-notes chord, the vector will be close to the 0 point. Thus the larger the magnitude of the sum of the n-th Fourier component, the less movement is required in voice leading to the perfect n-note chord. Thus they form a inverse relationship. Jason Yust , in his paper, gave another way of interpreting the DFT in terms of phase spaces and the vector in the phase spaces represented by pitch set class. Phase space 1 to 6 are all cicles with 12 units of circumference. Phase

space 1 is the pitch class circle, and other phase spaces can be obtained by multiply by $2, 3, 4, 5, 6$. They are all "superimposition" of the first phase space. Any pitch class set can be taken as the "circular average" of each pitch inside. The process is actually reversible using a more general form of

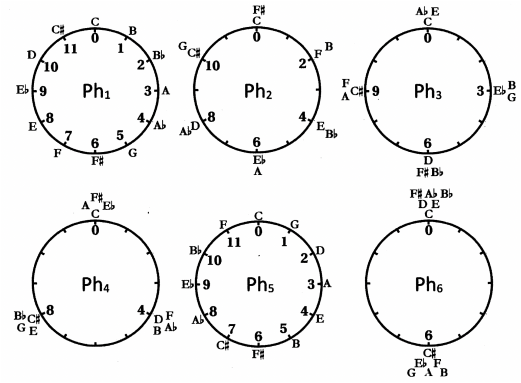

Figure 1: Phase Space

DFT:

$$f_n = \sum_{j=0}^{m} p_i e^{i2\pi jn/m} \frac{1}{m}$$

, given a full DFT of a set class by summing up all 12 components, we can find the set class by applying the IDFT (inverse DFT process). According to Yust, "We can think of the DFT, then, as a way of converting between two representations of the same object. One is the raw pitch-class vector, which simply tells us how much of each pitch class is present. The other, the DFT, represents the pcset (or multiset) as a sum of periodic functions." Each component is simply a periodic function of the phase space.

Application

Another interesting property of the Fourier component is each of them represents a typical harmonic quality (scales). According to Yust, " which we may describe as (1)chromaticism (2) quartal quality, (3) hexatonicity, (4) octatonicity, (5) diatonicity, (6) whole-tone." Since f_n is large when the set class has a small voice leading distance to the perfect n-note chord, we can use the DFT to apply harmonic analysis on separate voices.

In terms of scales, we want to avoid even distribution in semitones as much as possible (chromaticism) and focus on maximaly even set class. This means we want f_1, f_2 component to be as small as possible. An more intuitive explanation given by Yust is:"The scale-theoretic preference for even collections reflect a presupposition that no region of the pitch-class circle is privileged over any other. Hence, a high f_1 or f_2 collection on the surface of the music will be assumed to be a subset of a larger collection that reduces $|f_1|$ and $|f_2|$ by filling in gaps." All of the even five or seven notes set class has a high $|f_5|$ value, which means its affitnity to the distonicity. Low $|f_5|$ value might be indication of symmetric scales such as wholetone, hexatonic and octatonic scales.

One last interesting usage of DFT is analyzing the common tones between two set classes. According to Yst, "A well-known result from analysis, the convolution theorem, states that cross-correlation is to multiplying Fourier magnitudes and subtracting Fourier phases." We can calculate the common tones between two pitch set classes with the following formula:

$$\frac{1}{12}\sum_{n=0}^{11}|f_n(A)||f_n(B)|cos(\phi_n(A)-\phi_n(B))$$

From discussion above, we know common tones shows a voice leading distance which means harmonic closeness. This agains shows the magnitude of the distance in the phase space for these components gives a good indication of common tones function. By looking at the phase difference and common tone approximation, we can also find which scale contributes most to the common tones og two pitch set classes.

References

[1] Temperley, David. "A Bayesian Approach to Key-Finding." In Music and Artificial Intelligence, edited by Christina Anagnostopoulou, Miguel Ferrand, and Alan Smaill, 195–206. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2002.

[2] Tymoczko, Dmitri. "Set-Class Similarity, Voice Leading, and the Fourier Transform." Journal of Music Theory 52, no. 2 (2008): 251–72.