Probabilistic Approach in Measuring the Temporal Dynamics of Melodic Complexity of Jazz Improvisation

Philip Pincencia,
University of California, San Diego

***Abstract*—Jazz improvisation has been a main challenge to a content-based Music Information Retrieval. Previous studies have proposed different approaches in defining and analyzing complexity of a song, however there has not been enough effort to measure the complexity of the jazz solo as the solo progresses. This study aims to measure how the melodic complexity of jazz solo changes over time. A few methods from previous studies will be discussed and then a model will be proposed that hopefully achieves this intended measure and compare it against existing analysis methods. Promising approaches include using Information Theory or Probabilistic Models, but we are still determining if these approaches are valid. Due to the subjective approach in analyzing music, assumptions need to be made, but if time permits different or weaker assumptions will be considered. Whether this approach is meaningful or not hopes to bring an interesting insight into how we quantify jazz improvisation.**

***Index Terms*—Melodic Complexity, Probabilistic Approach, Jazz Improvisation, Temporal Dynamics.**

I. Introduction

Measuring complexity in music has been done quite a few times. There are different types of complexities that people have considered, notably Rhythmic Complexity, Harmonic Complexity, but in this research, we are only concerned with the melodic complexity, which I will define it later. Previous works used Information theoretic methods, entropy, or even novel methods such as chroma vector have been developed. Regardless of the method, most are, however, only concerned with the complexity of the song as a whole and not much has been done to measure complexity throughout the song itself.

Particularly in jazz improvisation, which is a complex creative process. It requires musicians to spontaneously play according to what is happening in the moment. And so as a jazz musician myself, I want to come up with something that can measure how we listeners think about a jazz solo and makes sense out of it. And with this measure, I am hoping that it can give insight into how jazz musicians structure and develop their solo, which can help beginners learn how to improvise well.

II. Considered Methods

Before getting into the main method used, I will briefly discuss two methods I previously considered and implemented as a potential solution. This discussion will show some of the key qualities that I am looking for in measuring the melodic complexity.

The first measure is based on what [1] uses to measure the dissonance between the harmony and the melody.

III. Basics

The representation of each MIDI note is *pitch:duration*, with $pitch \in \{n \in \mathbb{Z} : 10 \leq n \leq 98\}$ and $duration \in \mathbb{Z}^+$. *Although MIDI numbers ranges from 0 to 128, the physical range of a piano (which has the widest range) is from MIDI number 21 to 108, and to make all the notes to be represented by only 2 symbols, we shift all MIDI number down by 10. Adding rest, which is denoted as 10, the resulting range is 10 - 98.* (Since the MIDI files we are playing with have constant volume, we assume it to be irrelevant and therefore we will ignore it.) The alphabet is therefore $\Sigma = \{0, 1, ..., 9, :\}$ with $|\Sigma| = 11$.

Denote training sequence as $q_1^n = q_1 q_2 ... q_n,\ q_i \in \Sigma$. Notation: q_a^b means a subsequence from q_a to q_b, $a \leq b$. Given q_1^n, we want to learn the probability distribution $\hat{P}(s_n|s_1^{n-1}),\ \forall s_n \in \Sigma$. s_1^{n-1} is the prediction context. For a D-order Markov Model, then $\hat{P}(s_n|s_1^{n-1}) = \hat{P}(s_n|s_{n-D}^{n-1}) \approx \dfrac{N(s_n|s_{n-D}^{n-1})}{\sum_{\sigma \in \Sigma} N(\sigma|s_{n-D}^{n-1})}$, where $N(\cdot|s_{n-D}^{n-1}) = \#$ of times $\cdot$ appears after the context s_{n-D}^{n-1}.

In practice, we add initial count δ at each entry to avoid trouble when $N(s_n|s_{n-D}^{n-1}) = 0$:

$$\frac{N(s_n|s_{n-D}^{n-1}) + \delta}{\sum_{\sigma \in \Sigma} N(\sigma|s_{n-D}^{n-1} + \delta)},$$

where δ is a parameter chosen carefully.

However, we do not want to this s since this is just some approximation. We want more structured way of solving this *zero-frequency problem.*

IV. Variable-Order Markov Model (VMM)

Since we would want the context length be flexible, a variable-order Markov Model is needed. One of the most popular VMM algorithm is Prediction by Partial Matching (PPM), which is an adaptive statistical data compression technique. This method is presented in [2]:

- **Exclusion Principle:** If a context at a lower level is a suffix of a context at a higher level, this context is excluded at the lower level → **Prioritize higher level**

P. Pincencia is an undergraduate Computer Engineering student at the University of California San Diego

- **Escape Mechanism:** For each context s of length $k \leq D$, we allocate a probability mass $\hat{P}_k(esc|s)$ for all symbols that did not appear after the context s (in the training sequence). The remaining mass $1 - \hat{P}_k(esc|s)$ is distributed among all other symbols that have non-zero counts for this context (i.e. appear after s).

The general expression for all PPM algorithm is:

$$\hat{P}(\sigma|s_{n-D+1}^{n}) = \begin{cases} \hat{P}(\sigma|s_{n-D+1}^{n}), & s_{n-D+1}^{n}\sigma \in q_1^n \\ \hat{P}(\sigma|s_{n-D+2}^{n})\hat{P}(esc|s_{n-D+1}^{n}), & \text{else} \end{cases}$$

where σ represents the symbol after the context s.
Here, we ONLY consider the PPM-C (Method C) variant.
For each sequence s and symbol σ

- $N(s\sigma) = \#$ of $s\sigma$ seen in the training sequence.
- $\Sigma_s = \{\sigma : N(s\sigma) > 0\}$ is the set of symbols appearing after the context s (in the training sequence)

The formula for each $\hat{P}_k(\cdot|s)$ is

$$\hat{P}_k(\sigma|s) = \frac{N(s\sigma)}{|\Sigma_s| + \sum_{\sigma' \in \Sigma_s} N(s\sigma')}, \text{ if } \sigma \in \Sigma_s$$

$$\hat{P}_k(esc|s) = \frac{|\Sigma_s|}{|\Sigma_s| + \sum_{\sigma' \in \Sigma_s} N(s\sigma')}$$

Graphically looking at $\mathcal{T}$, $|\Sigma_s|$ = number of children of the node s and $\sum_{\sigma' \in \Sigma_s} N(s\sigma')$ = total of the counters of those children. There's no justification for these escape mechanisms.

Implementation using a Trie $\mathcal{T}$

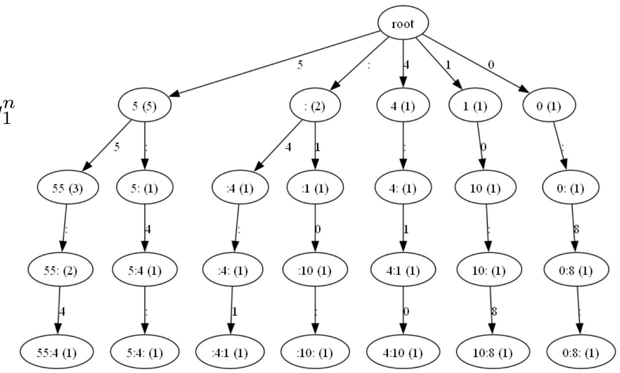
Each node in $\mathcal{T}$ is associated with a symbol and a counter.
Max Depth of $\mathcal{T} = D + 1$
Algorithm:

Algorithm 1 ConstructTrie

Require: $q_1^n = q_1 ... q_n, D \geq 0$
Ensure: $\mathbf{P}(\sigma|s)$
  root $\leftarrow \epsilon$
  $i \leftarrow 1$
  while $i \leq n$ **do**
    $x \leftarrow q_{\max(i-D,1)}^{i-1}$ $\triangleright$ $|x| \leq D$
    for σ' in x **do**
      $N(\sigma')++$
    end for
    for all $s \leq D$ **do**
      $s' \leftarrow$ Path ϵ to longest suffix of s
      if $N(s'\sigma') > 0$ **then**
        $\Sigma_{s'} \leftarrow \sigma'$
      end if
      Use the equations above to induce $\hat{P}(\sigma|s)$
    end for
    $i++$
  end while

Note that after parsing the first D symbols, each newly constructed path is of length $D + 1$
Fact: counter of any node with corresponding path $s\sigma$ (where σ is the symbol associated with the node) is $N(s\sigma)$

Realization: The way the tree $\mathcal{T}$ is built is like a Multiway Trie (MWT).
As an example, let $q =$ "55 : 4 : 10 : 8 : " with $D = 3$. Then, the trie is represented in Figure 1.



Trie built from the training sequence $q =$ "55 : 4 : 10 : 8 : " with maximal Markov order $D = 3$

Implementation is available at https://github.com/1618lip/mel_vomm.
With this, we can start to measure the melodic complexity of the music.

V. Sliding Window Analysis

To get the temporal dynamics complexity measurement, I use the sliding window technique. Given the processed representation of the entire jazz solo $\mathcal{W}$, the idea is to take a subinterval $W \in \mathcal{W}$ of a particular length as a parameter. This is the *window* length.

1)

Given the XML (or MXL) file, I parsed in

VI. Pitch Probability Distribution over Chord

Let $\Omega = \{C, C\#, D, ..., A\#, B\}$ be the sample space and denote $X : \Omega \to \mathbb{R}$ be a random variable. Denote the underlying chord as $C = \{c_1, c_2, ..., c_M\}$ with $c_k \in C$ as the chord tones. Typically $M = \{3, 4, 5\}$.
Let $\omega \in \Omega$ be an arbitrary note, then

- If $\omega = c_1$, then $X(\omega) = \mathcal{N}(w_1, \sigma^2)$
- If $\omega \in C$ and $\omega \neq c_1$, then $X(\omega) \sim \mathcal{N}(w_2, \sigma^2)$
- If ω is in the *first priority scale* associated with C, $X(\omega) \sim \mathcal{N}(w_3, \sigma^2)$
- If ω is in the *second priority scale* associated with C, $X(\omega) \sim \mathcal{N}(w_4, \sigma^2)$
- If ω is in the *third priority scale* associated with C, $X(\omega) \sim \mathcal{N}(w_5, \sigma^2)$
- Else, then $X(\omega) \sim \mathcal{N}(w_6, \sigma^2)$,

where $w_1 > w_2 > ... > w_6 \geq 1$ and $\sigma^2 \ll 1$. Each $X(\omega)$ can be thought of having a fixed weight with an additive Gaussian Noise, to account for variety in pitch likelihood. If we do this $\forall \omega \in \Omega$, we get $X(\Omega)$. Then, to get the probability of each note ω_k, we do

$$p(\omega_k) = \frac{e^{X(\omega_k)}}{\sum_{\omega \in \Omega} e^{X(\omega)}}.$$

A. *Example*

1) Subsubsection Heading Here: Subsubsection text here.

VII. Conclusion

The conclusion goes here.

Appendix A
Proof of the First Zonklar Equation

Appendix one text goes here.

Appendix B

Appendix two text goes here.

Acknowledgment

I would to thank Professor Massimo Franceschetti and PhD. Student Tornike Karchkhadze in giving me feedback and helpful insights. I would also especially like to express gratitude to my family for the endless support they have given me while I am working on this project.

References

[1] V. Tapiavala, J. Piesner, S. Barman, and F. Fu, "Reinforcement learning jazz improvisation: When music meets game theory," 2024.

[2] R. Begleiter, R. El-Yaniv, and G. Yona, "On prediction using variable order markov models," *J. Artif. Intell. Res*, vol. 22, 12 2004.