Nulitta: Generating Stochastic Grammars from Temporal Chord Probability Distributions in a Jazz Piano Corpus

Jeremy Zitomer

Advisors: Donya Quick, Ian Quinn, Andrew Jones

Yale University

In Partial Fulfillment of the Requirements for the Degree of Bachelor of Science in Computer Science
Abstract

This project attempts on-the-fly generation of stochastic context-sensitive jazz grammars from temporal probability histograms, in an effort to produce original jazz scores based on the JUMPY jazz piano corpus, via algorithmic composition. These histograms convey the probabilities of different chords in this corpus occurring at varying distances from one another. I present a novel generative algorithm that transforms the descriptive information in these histograms into a productive framework for jazz chord progression creation. Additional modifications to this novel generative algorithm, including distance-weighting schemes, chord space reduction, and template bindings using Donya Quick’s Kulitta program, are employed as attempts to refine this generative model (to increase the quality of the music produced). The implementation of this algorithm serves as an extension to Kulitta, whose generative capacities have thus far been centered on learning and leveraging probabilistic grammars for automated composition.

Overall, while the scores produced by this new algorithm differ drastically in their unigram probability distributions from the corpus from which they were generated, the qualitative aspects of those scores demonstrate the viability of this algorithm as a tool to produce human-written-sounding music. Most notably, many of the versions of this algorithm yield scores whose local chord progressions and broader syntactic structure mimic real jazz music in the style of the JUMPY corpus. Specific algorithmic modifications, including limiting chord spaces based on chord unigram probabilities and cardinality, as well as weighting source chord temporal probability histograms by a Gaussian function of their distance from destination chords, seem to improve this model further and contribute to more reasonable-sounding music that is dissonant (or even consonant) within acceptable bounds.
Introduction

A variety of methodological approaches to algorithmic music composition have been taken in recent years, including compositional schemes based on formal grammars, Markov chains, artificial neural networks, and evolutionary algorithms, to name a few. Of these approaches, grammar-based composition has proven to be particularly popular and useful; large bodies of literature exist on the generation of phrasal structures and entire pieces from pre-established musical grammars.

Fernández and Vico define a formal grammar as “a set of rules to expand high-level symbols into more detailed sequences of symbols (words) representing elements of formal languages.” In a musical context, those high-level symbols frequently take the form of broad classes of chords (groupings of three or more musical tones) distinguished by their function in a larger phrase or piece, while low-level terminal symbols (the musical equivalent of words) typically represent specific chords, like V or ii♭. Canonical high-level function classes include tonic, subdominant, and dominant—many musical phrases, especially in classical music, take the form of Tonic → Subdominant → Dominant → Tonic, but this and other common paradigms vary in structure and relevance across genres, becoming especially difficult to define or apply in the realm of jazz, notorious for its relatively freeform and often experimental nature. Further complications in the development of broadly applicable jazz grammars arise in that traditional models of jazz chord progressions use classically-derived categories and labels that assume chords

---

constructed as consonant triads or as seventh chords composed of stacked thirds; in practice, limiting the alphabet of a jazz grammar to such chords excludes cases where chord categorizations are ambiguous, or best framed as collections of tones that defy standard Roman numeral labels. For these reasons, the usefulness of hand-written rules in generative jazz grammars is limited, and algorithms that can generate their own (possibly unconventional) rules from a jazz corpus appear to be valuable.

One data model already in use among corpus analysts and linguists alike is the temporal probability distribution, in which the probabilities of the appearance of various words/chords (henceforth referred to as “units”) or classes of units are plotted as a function of position in a document/musical score, where this position can either be defined absolutely (e.g. in the 33rd measure of a score) or relative to other units (e.g. 8 beats after a IV chord). Such distributions are easily calculated from relative frequency chord histograms assembled based on a given musical corpus; however, their isolated applicability in automated composition is currently narrow, as they typically do not suggest an obvious structure to a music/textual document, which makes the generative process less straightforward and especially complicates the problem of trying to compose a piece that sounds human-written.

From this, a question arises: how can we convert a series of corpus-derived temporal probability distributions to a stochastic grammar—i.e. a grammar whose rules have corresponding probabilities, indicating the likelihood of a given higher-level symbol being expanded into the particular sequence of symbols associated
with that rule—where the symbols in that grammar have associated metric durations?

The inside-outside and forward-backward algorithms both approach similar questions to this one, but neither quite responds to this question exactly. While the inside-outside algorithm learns generation probabilities for a probabilistic context-free grammar (PCFG), the forward-backward algorithm establishes the state transition probabilities in a Hidden Markov Model (HMM). Indeed, it would be useful to have an algorithm that learns the generation probabilities for a probabilistic context-sensitive grammar (PCSG); however, it is crucial that such an algorithm keep space-efficiency in mind, since depending on the degree of context sensitivity in the grammar, keeping track of all possible generation probabilities in all possible contexts could quickly become wildly space-inefficient.

This project seeks to answer these questions and takes a novel approach in stochastic grammar generation/application in the context of automated composition of chord progressions, in that it produces a progression by following the below (new) algorithm:

1. Select a first chord at random based on the unigram probability distribution of chords in a given jazz corpus.
2. Loop the following steps until the desired number of chords have been generated:

---

a. For each chord that has been generated, find the probability
distribution for the set of chords $k$ beats after occurrences of said
chord in the corpus, where $k$ is the number of beats between said
chord and the chord currently being generated. (Note: $k$ will be
different for each chord that has been generated.)
b. Add those probability distributions together, and normalize them.
c. From that aggregate probability distribution, choose a chord at
random. This chord will be the next chord in the sequence being
generated.

Specifically, the probability of generating a given chord $c'$ on beat $k$ of a progression
is as follows:

$$ P(c_k = c') = P^U(c'), \quad k = 1 $$

$$ P(c_k = c') = \frac{\sum_{i=1}^{k-1} P(c' \rightarrow_i c_{k-i})}{\sum_{i=1}^{k-1} \sum_{c \in C} P(c \rightarrow_i c_{k-i})}, \quad k > 1, \text{ where} $$

- $c_k$ denotes the $k^{th}$ chord generated, which is also the $k^{th}$ chord in the
  final sequence (or the $k^{th}$ to last chord if the sequence is being
generated from back-to-front).
- $P^U(c)$ denotes the unigram probability of chord $c$ in the original
corpus
- $P(c \rightarrow_j c_n)$ denotes the probability that chord $c$ occurs $j$ beats after
  chord $c_n$ (these probabilities are evaluated via relative frequency
  analysis of chord occurrence in the corpus).
• \( C \) denotes the set of all chords that can be generated in a given score (this may be the set of all chords in a corpus, or the 100 chords with greatest unigram probabilities, or all chords in a corpus with cardinality-\( 4^3 \), e.g.).

Certainly at first glance, this algorithm doesn’t seem quite like a context-sensitive grammar. However, consider the following rewrite of the above formulas in a stochastic grammatical form, where \( c_a, c_b, \ldots, c_{|C|} \) comprise the set of chords \( C \) in the corpus, and \( c_1, c_2, \ldots \) represent the 1st, 2nd, etc. generated chords in the sequence.

\[
S \rightarrow c_a, \quad P = P^u(c_a) \\
S \rightarrow c_b, \quad P = P^u(c_b) \\
\ldots
\]

\[
c_1 \rightarrow c_1c_a, \quad P = \frac{P(c_a \rightarrow c_1)}{\sum_{c \in C} P(c \rightarrow c_1)}
\]

\[
c_1 \rightarrow c_1c_b, \quad P = \frac{P(c_b \rightarrow c_1)}{\sum_{c \in C} P(c \rightarrow c_1)}
\]

\[
\ldots
\]

\[
c_1 \ldots c_n \rightarrow c_1 \ldots c_n c_a, \quad P = \frac{\sum_{i=1}^{n} P(c_a \rightarrow_i c_{n+1-i})}{\sum_{i=1}^{n} \sum_{c \in C} P(c \rightarrow_i c_{n+1-i})}
\]

\(^3\) For our purposes, cardinality signifies the number of distinct pitch classes contained in a chord. For example, \([C4,E4,G4,C5]\) has a cardinality of 3.
\[ c_1 \ldots c_n \rightarrow c_1 \ldots c_n c_b, \quad P = \frac{\sum_{i=1}^{n} P(c_b \rightarrow \ i \ c_{n+1-i})}{\sum_{i=1}^{n} \sum_{c \in C} P(c \rightarrow \ i \ c_{n+1-i})} \]

...  

Admittedly, the term ‘context-sensitive grammar’ is still arguably a slight misnomer here, given that the aggregate probabilities used for stochastic chord generation (defined in the formulas above) aren’t computed prior to the beginning of the chord progression generation process; rather, they are computed as needed. In other words, the probability of a specific chord being generated as the 6\textsuperscript{th} chord in a sequence, following 5 other specific chords, will not be computed unless those 5 specific chords actually comprise the opening of the generated sequence.

This new algorithm leverages the descriptive value of a corpus’ temporal chord probability histograms for generative purposes, but the question remains: how does this algorithm perform? How closely does the music it generates mimic that of the original corpus, and more importantly, does it sound good? The remainder of this project seeks to answer that question.

**Methodology and Data**

The corpus from which the temporal probability distributions in this project are extracted, is JUMPY, a recently created jazz piano corpus assembled by Ian Quinn and Andrew Jones. The temporal probability distributions contain the data corresponding to probabilities like \( P(c'' \rightarrow \ i \ c') \), as defined above; precisely, they display the probabilities that later chords (destination chords) appear at various
beat-distances after earlier chords (source chords), where those beat distances vary from 1 to 99. The probabilities themselves are reflected in two different styles: absolute probabilities (self-explanatory) and log-relative probabilities—i.e., the following:

$$\log\left(\frac{\text{the likelihood of Destination Chord } d \text{ appearing } x \text{ beats after Source Chord } s}{\text{the unigram probability of Destination Chord } d}\right)$$

The temporal probability distribution tables themselves also enable both forward and backward chord progression generation. Specifically, this is possible because for a given early chord E and later chord L, separated by n beats, the tables contain both $P(L \rightarrow _n E)$ (which enables forward-generation) and $P(E \rightarrow _{-n} L)$ (which enables backward-generation). (Note that the alterations required to the overall generation algorithm, in order to backward-generate a chord progression, are trivial.) The “chords” themselves are 50 ms slices of the corpus MIDI files, where all pitch classes contained in a given slice become part of the corresponding chord.

In addition to forward- and backward-generation, several other variations on this overall algorithm were implemented. Most notably, a distance-weighting feature was added, e.g. to allow destination chord probability distributions of more proximal source chords to have greater bearing on the next generated chord than the distributions of more distant chords. Specifically, the probabilities described above were altered as follows:

$$P(c_k = c') = P^U(c'), k = 1$$

$$P(c_k = c') = \frac{\sum_{i=1}^{k-1} P(c' \rightarrow_i c_{k-i}) \ast w(i)}{\sum_{i=1}^{k-1} \sum_{c \in C} P(c \rightarrow_i c_{k-i})}, k > 1$$
where \( w \) is a weighting function on \( i \), the distance between source and destination chords. Several different distance-weighting functions were attempted, including:

- a uniform weighting model
- an inverse-square model that favored nearer chords by a factor of \( 1/n^2 \), \( n \) being the distance between source and destination chords
- various n-gram models, e.g. for trigram: \( w(n) = \text{if } (n>(3-1)) \text{ then } 0, \text{ else } 1 \)
- a Gaussian weighting model of the form \( w(n) = e^{-\frac{(n-25)^2}{2\sigma^2}} + e^{-\frac{(n-25)^2}{2\sigma^2}} \). This privileges predictive distributions from immediate neighbor chords and chords approximately 25, 50-ms slices away from the chord currently being generated. This allows both immediate passing neighbors and more distant (but important) syntactic/functional neighbors (i.e. a V chord that should be followed eventually by a I chord) to have significant bearing on the chords that follow them.

Other alterations on this algorithm included:

- Limiting the generative chord space to the top 100 or 200 most unigram-probable chords in the original corpus
- Generating \( k \) times more chords than the original algorithm, then taking only every \( k^{th} \) chord of the generated progression (this would aim to increase the speed of the harmonic progression among the generated chords)
- An “early-vote” system in which each previously generated source chord, based on its individual destination chord probability
distribution at the appropriate beat-distance, randomly recommends a destination chord; the “weights” of those destination chord recommendations are then assigned according to \( w \), or uniformly if there is no weighting function. Those recommendations (with their corresponding weights) are then pooled together, from which a destination chord is finally selected.

- By default, whenever this algorithm generated consecutive identical chords, my implementation combined them into one chord whose duration was equivalent to the sum of all of its constituents’ original durations.

- By default, this algorithm only actually used the temporal destination chord probability histograms from the 99 most recently generated source chords, since the histograms themselves did not contain information on probabilities of chord generation by source chords 100 beats or greater away from a destination chord.

The implementation for this project was Haskell-based, in order to use this new generative framework as an extension on Kulitta, Donya Quick’s AI framework for automated music composition that uses generative musical grammars to create original scores.\(^4\) This extension also involved expanding Kulitta’s grammatical alphabet (which is currently best-suited for classical sonorities) to include unconventional chord descriptions, e.g. lists of pitch-classes (which can best

accommodate non-classical jazz sonorities). However, the majority of this work is not directly dependent on Kulitta, and can largely stand on its own.

Notably, after these chord sequences were generated, they were fitted to a series of common jazz voice-leading templates, provided by Andrew Jones, and then published to MIDI files.

**Results**

For each variation on the algorithm, two evaluations were made:

1. A Chi-Square Goodness of Fit Test between a) the probability distribution of the selected generative chord space, and b) the relative frequency distributions of the chords generated by that algorithm in a 1000-chord sample of music.

2. A qualitative evaluation of that 1000-chord sample.

*Baseline Algorithm (Forward-Generated, Absolute Probabilities, No Special Distance-Weighting, Unrestricted Chord Space, No Take-Every-N Chords, No “Early Voting”)*

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 2131.03 with 1356 degrees of freedom. This is substantially different from the natural chord space unigram probability distribution, at a p-value of p < 0.00001.

Qualitative Evaluation: These chords are seemingly dissonant and random. There are only rare 3-4 chord flashes in which a key is established. The voice-leading quality is highly variable. Neighboring chords only occasionally seem to reflect harmonic functions.

*Baseline Algorithm + Log Probabilities*
Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 62896.90 with 1356 degrees of freedom. This is substantially different from the natural chord space unigram probability distribution, at a p-value of $p < 0.00001$.

Qualitative Evaluation: These chords are even more dissonant and random. There are only very rare 2-3 chord flashes in which a key is established. The voice-leading quality is poor. Neighboring chords never really seem to reflect harmonic functions.

**Baseline Algorithm + Backward Generation**

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 2801.43 with 1356 degrees of freedom. This is substantially different from the natural chord space unigram probability distribution, at a p-value of $p < 0.00001$.

Qualitative Evaluation: These chords are more dissonant and random than the baseline, but less dissonant than Baseline + Log Probs. There are rare 3-4 chord flashes in which a key is established. The voice-leading quality is variable but generally poor. Neighboring chords rarely seem to reflect harmonic functions.

**Baseline Algorithm + Backward Generation, Log Probabilities**

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 32098.81 with 1356 degrees of freedom. This is substantially different from the natural chord space unigram probability distribution, at a p-value of $p < 0.00001$.

Qualitative Evaluation: These chords are still rather dissonant and random. There are very few times at which a key is established. The voice-leading quality is poor. Neighboring chords never seem to reflect harmonic functions.

**Baseline Algorithm + Only Top 100 Most Frequent Chords from Corpus**
Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 449.64 with 99 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of $p < 0.00001$.

Qualitative Evaluation: These chords are more dissonant and random than the baseline. There are essentially no times at which a key is established. The voice-leading quality is horrendous. Neighboring chords never seem to reflect harmonic functions.

Baseline Algorithm + Only Top 200 Most Frequent Chords from Corpus

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 787.62 with 199 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of $p < 0.00001$.

Qualitative Evaluation: These chords are even more dissonant than Baseline + Top 100 Most Freq Chords. There are essentially no times at which a key is established. The voice-leading quality is horrendous. Neighboring chords never seem to reflect harmonic functions.

Baseline Algorithm + 3-4-Cardinality Chords Only

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 9787.37 with 701 degrees of freedom. This is substantially different from the natural chord space unigram probability distribution, at a p-value of $p < 0.00001$.

Qualitative Evaluation: These chords are slightly less dissonant than the baseline. There are longer phrases which, while dissonant, do seem to have a tonal center. The voice-leading quality is variable but generally not great. Neighboring chords only occasionally seem to reflect harmonic functions.
Baseline Algorithm + 3-5-Cardinality Chords Only

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 1891.57 with 1248 degrees of freedom. This is substantially different from the natural chord space unigram probability distribution, at a p-value of $p < 0.00001$.

Qualitative Evaluation: These chords are slightly less dissonant than the baseline. Longer phrases seem to reach a point of tonicization for a little while. Voice-leading quality is generally not great. Neighboring chords occasionally reflect harmonic functions.

Baseline Algorithm + Only Top 100 Most Frequent 3-5-Cardinality Chords from Corpus

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 449.64 with 99 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of $p < 0.00001$.

Qualitative Evaluation: These chords are roughly as dissonant as the baseline. There are a small handful of times when a key is briefly established. The voice-leading quality is not good. Neighboring chords rarely seem to reflect harmonic functions.

Baseline Algorithm + Only Top 200 Most Frequent 3-5-Cardinality Chords from Corpus

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 787.62 with 199 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of $p < 0.00001$.

Qualitative Evaluation: These chords are generally less dissonant than the baseline. There are a small handful of times when a key is established for several measures. The voice-leading quality is generally not good. Neighboring chords rarely seem to
reflect harmonic functions, but there are some nice brief chord progressions scattered throughout the score.

**Baseline Algorithm + Gaussian Distance-Weighting, Sigma = 5**

**Chi-Square Goodness of Fit:** This comparative evaluation yielded a Chi-Square value of 2178.88 with 1356 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of p < 0.00001.

**Qualitative Evaluation:** These chords are somewhat less dissonant than the baseline. A key can be discerned occasionally. The voice-leading quality is subpar. Neighboring chords occasionally reflect harmonic functions.

**Baseline Algorithm + Gaussian Distance-Weighting, Sigma = 5 + 3-5-Cardinality Chords Only**

**Chi-Square Goodness of Fit:** This comparative evaluation yielded a Chi-Square value of 4667.69 with 1248 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of p < 0.00001.

**Qualitative Evaluation:** These chords are somewhat less dissonant than the baseline. However, a key can be discerned relatively constantly. The voice-leading quality is generally mediocre. Neighboring chords occasionally reflect harmonic functions.

**Baseline Algorithm + Gaussian Distance-Weighting, Sigma = 5 + Only Top 100 Most Frequent 3-5-Cardinality Chords from Corpus**

**Chi-Square Goodness of Fit:** This comparative evaluation yielded a Chi-Square value of 4493.70 with 99 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of p < 0.00001.

**Qualitative Evaluation:** This piece was the best observed thus far. These chords are somewhat less dissonant than the baseline at first, then become gradually less
dissonant until by the end they are mostly viably consonant. A key can be discerned almost the entire time. The voice-leading quality is generally okay. Neighboring chords begin to reflect harmonic functions more as the piece goes on, occasionally reflecting syntactic functions/relationships as well.

**Baseline Algorithm + Gaussian Distance-Weighting, Sigma = 5 + Only Top 200 Most Frequent 3-5-Cardinality Chords from Corpus**

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 1163.16 with 199 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of p < 0.00001.

**Qualitative Evaluation:** This piece was roughly equivalent to the piece produced by the last algorithm (Top 100 3-5 Cardinality Chords, etc.), except it was a bit more dissonant throughout.

**Baseline Algorithm + Gaussian Distance-Weighting, Sigma = 1 + Only Top 100 Most Frequent 3-5-Cardinality Chords from Corpus**

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 1946.51 with 99 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of p < 0.00001.

**Qualitative Evaluation:** This piece was more dissonant than when Sigma was equal to 5. The chords do become more consonant throughout the piece, but not by much, until the end. A key can be discerned almost most of the time. The voice-leading quality is generally subpar. Neighboring chords begin to reflect harmonic functions more as the piece approaches its end, occasionally reflecting syntactic functions/relationships as well. Some of the chord progressions at the end of this piece are compelling.
Baseline Algorithm + 7-Gram Distance Weighting + Only Top 100 Most Frequent 3-5-Cardinality Chords from Corpus

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 37950.60 with 99 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of p < 0.00001.

Qualitative Evaluation: This piece was relatively consonant (perhaps the most consonant thus far) and in-key, but with lots of repetitive clustering. The voice leading here was good, and the relationships between adjacent chords often reflect those of passing and syntactic neighbors.

Baseline Algorithm + Trigram Distance Weighting + Only Top 100 Most Frequent 3-5-Cardinality Chords from Corpus

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 118910.35 with 99 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of p < 0.00001.

Qualitative Evaluation: This piece was mostly consonant and in-key, but centered mostly around a pretty limited set of chords. The voice leading here was fine, but perhaps simply because of the lack of harmonic variety. Because of that lack of variety, the relationships between adjacent chords are uninteresting—i.e. mostly passing or variations on a central theme.

Baseline Algorithm + Inverse-Square Distance Weighting + Only Top 100 Most Frequent 3-5-Cardinality Chords from Corpus

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 25870.15 with 99 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of p < 0.00001.
Qualitative Evaluation: This piece was extremely consonant and in-key, but also centered mostly around a pretty limited set of chords. There were a number of compelling chord progressions that predominantly began and ended with the same root chord. The voice leading here was good, but perhaps because of the lack of harmonic variety. Adjacent chords provided many compelling reflections of passing neighbor relationships.

Baseline Algorithm + 7-Gram Distance Weighting + Only Top 100 Most Frequent 3-5-Cardinality Chords from Corpus + Early-Voting

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 44362.26 with 99 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of p < 0.00001.

Qualitative Evaluation: This piece was a tad less consonant than its non-early voting equivalent but mostly had a steady key throughout. However, the same small set of chords appeared again and again. That said, the voice leading here was good, and the relationships between adjacent chords often reflect those of passing and syntactic neighbors. Some of the chord progressions were especially compelling.

Baseline Algorithm + Gaussian Distance-Weighting, Sigma = 5 + Only Top 100 Most Frequent 3-5-Cardinality Chords from Corpus + Early-Voting

Chi-Square Goodness of Fit: This comparative evaluation yielded a Chi-Square value of 1719.42 with 99 degrees of freedom. This is substantially different from the new chord space unigram probability distribution, at a p-value of p < 0.00001.

Qualitative Evaluation: This piece was interesting, but more dissonant than its non-early-voting equivalent. A key can be discerned almost the entire time, but a small amount of persistent ugly dissonance can as well. The voice-leading quality is
generally okay. However, adjacent chords do reflect a lot of neighborhood and especially syntactic relationships in this piece.

**Table 1: Chi-Square Goodness of Fit Statistics under Different Algorithm Modifications**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$X^2$ statistic</th>
<th>Deg. Freedom</th>
<th>$X^2$ Crit. Val. at p &lt; 0.05</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline (Fwd-Gen, Absolute Probs., No Weighting, Unrestricted Chords, No Take-Every-N, No Early-Vote)</td>
<td>2131.03</td>
<td>1356</td>
<td>1442.78</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + Log Probs.</td>
<td>62896.90</td>
<td>1356</td>
<td>1442.78</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + Bkwd-Gen</td>
<td>2801.43</td>
<td>1356</td>
<td>1442.78</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + Bkwd-Gen, Log Probs.</td>
<td>32098.81</td>
<td>1356</td>
<td>1442.78</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + 100 Most Freq. Chords</td>
<td>449.64</td>
<td>99</td>
<td>123.22</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + 200 Most Freq. Chords</td>
<td>787.62</td>
<td>199</td>
<td>232.91</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + 3-4-Cardinality Chords</td>
<td>9787.37</td>
<td>701</td>
<td>763.70</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + 3-5-Cardinality Chords</td>
<td>1891.57</td>
<td>1248</td>
<td>1331.30</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + 100 Most Freq. 3-5-Card. Chords</td>
<td>283.98</td>
<td>99</td>
<td>123.22</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + 200 Most Freq. 3-5-Card. Chords</td>
<td>468.87</td>
<td>199</td>
<td>232.91</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + Gaussian Weighting, Sigma = 5</td>
<td>2178.88</td>
<td>1356</td>
<td>1442.78</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + 3-5-Cardinality Chords + Gaussian Weighting, Sigma = 5</td>
<td>4667.69</td>
<td>1248</td>
<td>1331.30</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>Baseline + 100 Most</td>
<td>4493.70</td>
<td>99</td>
<td>123.22</td>
<td>&lt;0.00001</td>
</tr>
</tbody>
</table>

5 Such a low p-value indicates substantial difference between the chord frequency distribution of the generated score, and the unigram chord probability distribution of the original corpus.
| Freq. 3-5-Card. Chords + Gaussian Weighting, Sigma = 5 | 1163.16 | 199 | 232.91 | <0.00001 |
| Baseline + 200 Most Freq. 3-5-Card. Chords + Gaussian Weighting, Sigma = 5 | 1946.51 | 99 | 123.22 | <0.00001 |
| Baseline + 100 Most Freq. 3-5-Card. Chords + Gaussian Weighting, Sigma = 1 | 37950.60 | 99 | 123.22 | <0.00001 |
| Baseline + 100 Most Freq. 3-5-Card. Chords + Trigram Weighting | 118910.35 | 99 | 123.22 | <0.00001 |
| Baseline + 100 Most Freq. 3-5-Card. Chords + Inverse-Square Weighting | 25870.15 | 99 | 123.22 | <0.00001 |
| Baseline + 100 Most Freq. 3-5-Card. Chords + 7-gram Weighting + Early-Voting | 44362.26 | 99 | 123.22 | <0.00001 |
| Baseline + 100 Most Freq. 3-5-Card. Chords + Gaussian Weighting, Sigma = 5 + Early-Voting | 1719.42 | 99 | 123.22 | <0.00001 |
| Baseline + 100 Most Freq. 3-5-Card. Chords + Gaussian Weighting, Sigma = 5, Every-4th-Chord | ***6 | *** | *** | *** |
| Baseline + 100 Most Freq. 3-5-Card. Chords + Gaussian | *** | *** | *** | *** |

6 Long run-times prevented me from running algorithms with every-nth-chord modifications, as those algorithms take n times longer to run than the others, which already take upwards of 20 minutes each. See Future Directions.
Discussion

These results suggest that this algorithm, with the proper modifications, is indeed capable of producing somewhat viable music. As can be deduced from the specific combinations of modifications chosen for algorithms in Table 1, modifications that produced worse sounding music than their alternatives were discarded and eventually ignored throughout the testing process.

However, specific modifications for which the corresponding $X^2$ statistic is closer to the relevant $X^2$ critical value, do not seem to necessarily produce scores that sound less dissonant. In fact, some of the most consonant scores (yielded particularly by n-gram and inverse-square weighting models) had the highest $X^2$ statistics. This suggests that consonant-sounding jazz scores don’t necessarily reflect the unigram chord probability distribution of the corpus from which they were generated.

As for specific modifications that increased sound quality, Gaussian distributions were important contributors to good sound, especially at Sigma = 5. This makes sense, given that Gaussian distributions allow for both immediately adjacent and more distant syntactic neighbors to influence the production of a destination chord, which is music-theoretically justified.
Limiting a chord space based on cardinality and unigram probabilities also increased sound quality, because doing so eliminated a lot of noise generated by the presence of less common/more convoluted chords in a score.

Some modifications notably had adverse impacts on the score quality. In particular, n-gram/inverse-square models kept the chord space of their scores very limited, falling much more frequently into cycles of the same chords again and again. This is likely due to the short window of chords that these models use for predicting destination chords. Also, while Early-Voting seemed to increase syntactic neighbors, it simultaneously added more dissonance, which made the scores it generated sound less pleasant.

**Conclusions**

Overall, while the scores produced by this new algorithm differed drastically in their unigram probability distributions from the corpus from which they were generated, the qualitative aspects of those scores demonstrated the viability of this algorithm as a tool to produce human-written-sounding music. Specific algorithmic modifications, including limiting chord spaces based on chord unigram probabilities and cardinality, as well as weighting source chord temporal probability histograms by a Gaussian function of their distance from destination chords, seem to improve this model further and contribute to more reasonable-sounding music that is dissonant (or even consonant!) within acceptable bounds.

**Future Directions**

Several future directions for this work include:
• **Temporal Phrase Structure Grammars:** One of the obvious limitations of the current algorithm is that, beyond the temporal combining of adjacent identical chords in the generated chord progressions, there is almost no rhythmic variation in the scores produced. Future work should look to incorporate Kulitta’s temporal phrase structure grammars—which expand chord sequences into progressions of sonorities with learned, probabilistically varying durations—into my generation scheme.

• **Variation by place in piece:** One way to avoid the very repetitive clustering/harmonic homogeneity that some algorithms produced, would be to additionally vary the weighting schemes as a function of a destination chord’s absolute place in a piece, i.e. closer to the beginning or the end. This would hopefully add more variety to the harmonic progressions produced.

• **Non-Linear Generation:** It would be valuable to attempt different (i.e. non-front-to-back/back-to-front) chord generation schemes, including a replacement scheme similar to the way Kulitta implements temporal phrase structure grammar expansion, which is to say, top-down and branching-out, instead of solely forward or backward. However, this would likely require different temporal probability histograms to be created from the jazz piano corpus.

• **Every-nth-chord Algorithms:** Due to exceptionally long run times for every-nth-chord algorithms, I was unable to produce and evaluate scores generated by those algorithms. It would be wise to look into these
algorithms more deeply as they have important potential for speeding up the harmonic progressions in a generated score.

**Acknowledgements**

I would like to extend my deepest thanks to my thesis advisors: Donya Quick, Ian Quinn, and Andrew Jones, without whom this project would not have possible, particularly in granting me access to JUMPY (the jazz piano corpus), and offering guidance in navigating Kulitta and computational jazz theory. They invested many hours of work both in meetings and behind-the-scenes on my behalf, and for that I am extremely grateful. I would also like to thank the Yale University Department of Computer Science.
References

