Learning Style-Specific Rhythmic Structures

Emir Kapanci and Avi Pfeffer

Division of Engineering and Applied Sciences, Harvard University
{kapanci,avi}@eecs.harvard.edu

Abstract

The goal of computational music modeling is to construct models that capture the structure of music. We present our work on learning style-specific models for the rhythmic structure on a single line of music. The task by which the model is evaluated is to predict the duration of the next note given the sequence of durations leading to that note. We construct several different models that we train using works by a given composer (Palestrina), and assess the success of our models by looking at the prediction accuracy on unseen works by the same composer. We show that introducing style-specific musical knowledge improves the predictive ability of our models.

1 Introduction

The goal of computational music modeling is to construct models that can capture the structure of music. A computational model is often useful in systems based on artificial intelligence techniques: compositional, improvisational, and performance systems (Lopez de Mantaras and Arcos 2002), as well as systems for music transcription and music information retrieval. Computational models for music have been employed since as early as Hiller and Isaacson's work (1958) on machine composition of music using Markov chains and Simon and Sumner's description (1968) of a formal pattern language for music. Such models for structural organization in music are still studied in computational musicology.

We are particularly interested in the practical application of computational models to the transcription of music. From the rhythm perspective, music transcription requires associating onset times with discrete locations on a score. This involves both keeping track of the tempo and rhythm quantization. Tempo tracking and rhythm quantization are widely studied tasks, as they are problems that need to be addressed when working with real musical data such as audio or MIDI input. A significant number of computational methods have been suggested for these two tasks, both for audio input (Goto and Muraoka 1998; Raphael 2001; Scheirer 1998) and for inputs that are presented as a list of onset times (Cemgil and Kappen 2003; Desain and Honing 1994).

One particular aspect that is neglected in previous works is a language model for rhythm that contains substantial knowledge on the style of the pieces under consideration. It is well know, in the case of speech recognition, that a language model can greatly enhance the recognition accuracy (Pereira and Riley 1997). Consider two musicians equally proficient in transcription, trying to reproduce a musical score, where one musician has never listened to that specific style, and the other is an expert in that style. It is reasonable that the expert will be more accurate and faster in this task. We therefore suggest that methods for transcription of music in a particular style should be supported with a rhythm model that is specific to that style. Such models are more likely to produce accurate and fast (speed is especially important for online systems) rhythm quantifiers. An additional benefit of having a successful rhythm quantifier is that it will help in resolving uncertainty during tempo tracking, since the two problems are highly connected.

2 Problem Definition and Approach

2.1 Problem Definition

The goal of this paper is to build an accurate probabilistic model of single line rhythm in a specific musical style. There are three major limitations of this goal: first, we only consider rhythm and no other musical elements (such as pitch, dynamics, etc.); second, we look at single lines of music and disregard the parallel lines; and finally, we construct a model for a specific musical style, as opposed to a general model of musical rhythm. We discuss each limitation in turn.

We have decided to focus on the musical domain of rhythm alone, which simplifies the model space and allows us to investigate it more deeply. By considering rhythm alone, we do lose information that can be inferred from other musical elements. However, we believe that the models for rhythm and pitch are to some degree separable and there is a lot that can be inferred using a model for rhythm alone. Furthermore, to develop a coherent integrated model for pitch, rhythm and other musical elements, we must first understand models for each element individually.

Our focus is on modeling the rhythm on a single line of music. Modeling multiple lines of music is clearly a more complex task. We suggest that a model for monophonic rhythm can be used as a building block for polyphonic rhythm. One might argue that because of the parallel dependency of musical lines, an investigation of multiple lines is necessary for an accurate model of single line rhythm. However, rhythm is a primarily linear phenomenon and while the rhythmic dependency between parallel lines is sometimes important, the linear dependencies predominate. Therefore we choose to model monophonic rhythm first, and intend to extend our model to polyphonic rhythm in future work.

Modeling a specific style, often represented by a single composer, and modeling music in general are two distinct problems. Although modeling a specific style could appear to be a subset of the latter problem, the two are fairly different since modeling a single style allows more musical knowledge to be incorporated into the model. As a result, models of specific styles can actually be more complex but also more accurate, and they can be used in more sophisticated tasks for the given style (e.g. stylistic harmonization, melody completion). Most research in computational analysis of music has been on a single style due primarily to the ability to incorporate musical knowledge. Another intuitive reason for considering single styles follows from the observation that humans have different expectations when listening to specific styles. We believe that having separate models for a range of styles is a reasonable approach even for modeling music in general. In this paper we focus on the style of Palestrina.

The task by which we test the efficiency of our rhythmic model is prediction. In rhythm modeling, the two types of events are pitches and silences. We specify the task as predicting the duration and type of the next event, given the sequence of durations and types of events leading to it. There are several reasons for the choice of a prediction task. First, the predictive ability of a model is objectively measurable. We can learn a model using some works by a given composer, and use unseen works from the same composer to assess its success. We avoid the problems that can be encountered when using a subjective measure such as the similarity of the outputs generated by the model to the specific style.

Second, prediction of durations can assist in a number of practical problems such as automated accompaniment, beat tracking, rhythm quantization, and music transcription in general. In all of these problems, a predictive model can be used to relate continuous performance durations to discrete notated durations and vice versa.

Finally, the prediction task is an interesting problem in itself for both computers and humans. Prediction is inherent to the process of listening to music, and appreciating music has a lot to do with expectation (Narmour 1990). Music is based on developing, satisfying and breaking expectations, and humans that have the knowledge of a certain style have the "right" expectations when listening to a composition in that style. We hypothesize that when musical knowledge is incorporated into computational models, they will have a greater predictive capability, just as humans familiar with a style have more accurate expectations.

2.2 Methodology

In building and assessing models for rhythm, we took much of the methodology from work on natural languages, since there are a lot of similarities between natural languages and music. Both are means of communication and perceptual faculties of humans. In fact, it is widely accepted that the cognitive system organizes both linguistic and musical information using similar hierarchical descriptions (Chomsky 1965; Lerdahl and Jackendoff 1983; Longuet-Higgins 1976). Also, the data that is processed is sequential in both music and natural languages, which suggests that the employment of sequential models that work well for languages can also be beneficial for music.

Our methodology is as follows: First, following work in statistical natural language processing, we build probabilistic models using well-known frameworks. We then augment these models by incorporating musical knowledge explicitly. We hypothesize that musical knowledge will improve the predictive ability of the models. Finally, to check this hypothesis, we compare "informed" models to the "uninformed" models.

We learn and evaluate models by using a musical corpus that contains many works by a given composer. First, the majority of the compositions are used to learn a model. Then to test the predictive ability of the resulting model, we use unseen compositions from the same composer. We employ a cross-validation technique where we repeat the same procedure by using different pairs of training and test sets, and present the average result.

To evaluate the models, we use the perplexity metric, which is based on the information theoretic measure of entropy. When we don't know the true frequency of rhythmic events (analogous to the unknown frequency of words in natural languages), we can estimate the entropy of the model by the average negative log-probability of correct prediction on unseen compositions. Since for our purposes each composition is a sequence of rhythmic events, we take the average negative log-probability under our model, of the observation of each event given the subsequence leading to it. In Equation 1 below, k is the number of test sequences, n_j is the number of elements in the jth sequence, and d_{ji} denotes the ith element of the jth sequence. The probability that the model M assigns to an event e given the previous sequence s is given by $p_M(e|s)$.

$$\mathrm{logpr}(M) = -\left(\sum_{j=1}^{k} n_j\right)^{-1} \sum_{j=1}^{k}\sum_{i=1}^{n_j} \log p_M\left(d_{ji} \mid d_{j1},\ldots,d_{j(i-1)}\right) \qquad (1)$$

The perplexity of the model is then defined as $2^{\mathrm{logpr}(M)}$.

An intuitive interpretation of perplexity is the number of elements from which the next event is chosen given the rhythmic sequence up to that point. Therefore, the more predictive accuracy a model has, the lower its perplexity will be. An ideal perplexity would be as close to 1 as possible, while a perplexity of 1 would mean that the model always predicts the next duration correctly (it assigns a 100% probability to the correct event). This cannot be the case for real musical styles since the only time a perplexity of 1 is possible is when the model only generates the different-length prefixes of a unique sequence. Unfortunately, the true perplexity of any style is unknown. Therefore, we can never tell how close to the optimal we are, but perplexity can still be used to compare different models.

3 Data Collection and Representation

3.1 Choice of Composer

Our choice of composer for the experiments is Giovanni Pierluigi da Palestrina (*circa* 1525-1594), who is considered to be the classic model of Renaissance polyphony (Fux 1725; Jeppesen 1931), and is the primary composer of that style. There are a number of reasons for choosing Palestrina. Unlike some composers who write in multiple distinct styles, Palestrina has a relatively homogenous style[1], and therefore a successful rhythmic model learned using his compositions would be a general model for Palestrina-style rhythm. Also, rhythm is a key element in Palestrina's works: his compositions are primarily melodic (horizontal), subject to harmonic (vertical) constraints. Since we are focusing on single line rhythm, which is again a primarily horizontal phenomenon, and the task by which we evaluate our models is prediction, Palestrina's compositions yield an interesting and nontrivial corpus.

3.2 Musical Corpus

For music file format of the corpus, we chose the Enigma Transportable Format (ETF), which is a cross-platform format for use with Coda's Finale notation editor. Unlike the widespread MIDI standard (Loy 1985), which is a communication protocol between electronic instruments and computers, and primarily a representation of the performance of a composition, ETF provides a high-level, structured and exact representation of the musical composition. Almost all of the available ETF files for Palestrina are taken directly from the score, and therefore are accurate.

To obtain the rhythmic information from the ETF files, we implemented an ETF parser that converts the music files into an OCaml[2] structure, from which we can easily extract the desired information. The current work only uses the rhythmic information of the music files, but the conversion maintains most of the fundamental musical information in the ETF files, and can be used for different purposes such as the analysis of lyrics, pitch or multiple lines.

A relatively large ETF corpus on Palestrina is available online. After checking many files and eliminating the ones that were not suitable because of transcription errors, we obtained a fairly large and balanced corpus. Since Palestrina has somewhat different sub-styles for sacred and secular music, we included a representative number of each sub-style. To ensure homogeneity of the corpus, we discarded a small number of compositions in triple meter, and only kept compositions in duple or quadruple time. The final corpus we use for the experiments is composed of the soprano lines of 25 madrigals, 39 pieces from masses, and 37 motets or other sacred works (a total of 101 compositions).

[1] Palestrina's secular and sacred compositions have somewhat different characters, but can still be considered to be in the same "Palestrina style".

[2] OCaml is an object oriented version of Caml (a functional programming language from the ML family) developed at INRIA,http://caml.inria.fr/ocaml

3.3 Data representation

Rhythmic information on single lines is encoded by integer sequences. We first identify the possible note and rest durations in Palestrina's compositions by counting the occurrence of each symbol in the corpus. The eighth note was chosen to be the unit, since this is the shortest common duration in our corpus[3]. Examining the frequencies in the corpus, we have eight cases for notes of duration 1, 2, 3, 4, 6, 8, 12, and 16 units; and five cases for rests of duration 1, 2, 4, 6 and 8 units. Two special cases of "long note" and "long rest" are respectively used to represent notes longer than 16 units and rests longer than 8 units. These cases cover all occurrences of notes and rest in the corpus. Since we have a total of 15 categories, the perplexity of the models can at most be 15. However, we do not have a lower bound for Palestrina-style, so all we can say is that our perplexity results should be between 1 and 15.

4 Baseline Models

4.1 N-gram Model

The first baseline model we use is the ordinary n-gram model (Shannon 1948) widely employed in the field of computational linguistics. In an n-gram model, only the previous *n*-1 elements of the sequence are used to condition the probability of the next rhythmic event *d*, instead of the whole sequence leading to it. Under this assumption, the conditional probability of observing *d* is simply a function of the *n*-1 preceding events, as shown in Equation 2.

$$p_{n-gram}\left(d_{ji} \mid d_{j1},....d_{j(i-n+1)},....d_{j(i-1)}\right) = p_{n-gram}\left(d_{ji} \mid d_{j(i-n+1)},....d_{j(i-1)}\right) \qquad (2)$$

N-gram probabilities are estimated by looking at the n-gram frequencies in the training set, as shown in Equations 3 through 5.

$$f_{1-gram}\left(d_{ji}\right) = \frac{count\left(d_{ji}\right)}{\sum_{j=1}^{k} n_j} \qquad (3)$$

$$f_{2-gram}\left(d_{ji} \mid d_{j(i-1)}\right) = \frac{count\left(d_{j(i-1)}, d_{ji}\right)}{count\left(d_{j(i-1)}\right)} \qquad (4)$$

$$f_{3-gram}\left(d_{ji} \mid d_{j(i-2)}, d_{j(i-1)}\right) = \frac{count\left(d_{j(i-2)}, d_{j(i-1)}, d_{ji}\right)}{count\left(d_{j(i-2)}, d_{j(i-1)}\right)} \qquad (5)$$

Normally, when employing n-gram models, some n-grams will not be observed in the training set, although they can occur in reality. We need to assign non-zero probabilities to these unseen n-grams, and this may require an estimation

[3] The only durations smaller than an eight note in the corpus are rare sixteenth notes, which always occur in pairs. We consider each pair of sixteenth notes to be functionally equivalent to a single eighth note.

of the probabilities of all n-grams given the observed frequencies. This is not an important issue for the current problem given the small vocabulary size and the relatively large corpus. All unigrams and bigrams in the test set are observed. Therefore, we use a simple linear interpolation of n-gram frequencies as shown in Equation 6:

$$p(d_{ji} \mid d_{j(i-2)}, d_{j(i-1)}) = \lambda_3 f_3(d_{ji} \mid d_{j(i-2)}, d_{j(i-1)}) + \lambda_2 f_2(d_{ji} \mid d_{j(i-1)}) + \lambda_1 f_1(d_{ji}) \quad (6)$$

where the coefficients λ_1, λ_2, and λ_3 are constrained to be positive with their sum equal to 1. The values were obtained by comparing several reasonable assignments of values. The results of the experiments are not affected by the choice of coefficient values as long as λ_2 is sufficiently larger than λ_1, and λ_3 is sufficiently larger than λ_2.

The results of the experiments with n-grams are presented in Figure 1. In these experiments, λ_2 is equal to $100\lambda_1$, and λ_3 is equal to $100\lambda_2$. As outlined in section 2.2, we obtain the n-gram probabilities by looking at a training set, and the results are on a held-out test set. We get a perplexity of about 6 with unigrams, and observe a significant improvement when the model is extended to bigrams. A small additional improvement is obtained when we use trigrams. The significant gain from bigrams is explained by the fact that bigrams can readily capture some of the musical information that is binary in nature (most of the cases covered by rule 1 of section 5.1).

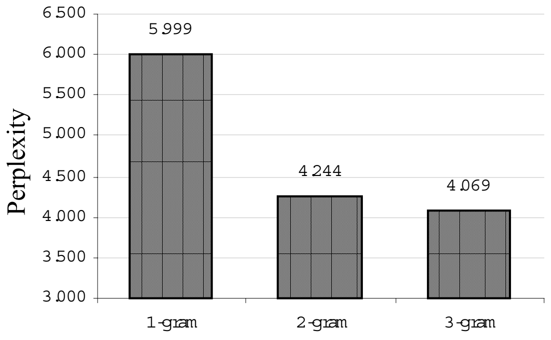

Figure 1. N-gram perplexities

4.2 Hidden Markov Model (HMM)

The second baseline model is a HMM (Rabiner and Juang 1986), where we have h hidden states and 15 possible observations from each state. In an HMM, the world moves through a sequence of unobserved states and an observation is output at each state. The observations only depend on the current state. Also, the Markov assumption states that the next hidden state depends only on the current state and not on previous states. An HMM is defined by three components: A transition model that determines the probability of the next hidden state given the current state, an observation model that determines the probability of an observation given the current state, and an initial model. For this work, observations are rhythm events, represented by one of the 15 categories. The integer sequences that represent the rhythm of compositions correspond to observed sequences from the HMM. We use the Baum-Welch algorithm (Rabiner and Juang 1986) to learn the model parameters for different numbers of hidden states.

Figure 2 presents the results of the experiments, and compares HMMs to the trigram model. The fluctuations in the curve result from the stochastic nature of the Baum-Welch algorithm. As the number of hidden states increases, we observe a general improvement in the perplexity, becoming more or less stable beyond 30 hidden states. We also see that HMMs with more than 15 hidden states are slightly better than the trigram model, although the difference may not be large enough to justify the added complexity and learning time of the HMMs with large number of hidden states.

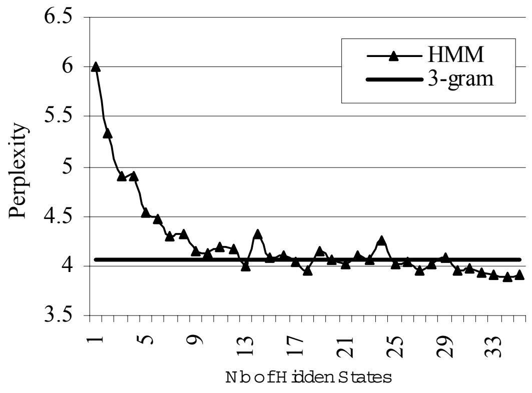

Figure 2. HMM vs. N-gram perplexity

5 Informed Models

In the models of section 4, the only style-specific knowledge was the set of outputs that are possible, which was obtained by searching through the corpus. In this section, we introduce more style-specific musical knowledge and incorporate it into the baseline models in different ways. The musical knowledge we introduce in this section is based on some rhythmic rules musicologists have identified for the style of Palestrina, and therefore the improved models in this section are specific to that style. However we believe that the general approach for determining and incorporating rhythmic information will be similar for different styles.

5.1 Musical Knowledge

According to musicologists (Jeppesen 1931), Palestrina's works always obey the following two rules: (1) There cannot be syncopated[4] notes on off-beats, and (2) a note cannot be

[4] Syncopation is an onset on a beat of less importance than one that intervenes prior to the next onset (i.e. a note is syncopated if it starts at a weak beat and there exists a stronger beat before the start of the next note).

longer than the previous note it is tied to. The first rule can be restated as disallowing notes longer than a half-beat on the off-beats (locations 2,4,6 and 8 when quadruple measures are divided in half-beats). Figure 3 illustrates the restrictions imposed by the second rule. In notating music, when a note traverses the mid-measure as in Figure 3(a) or the bar-line as in 3(b), we use a tie to emphasize the metric accent. In tied notes starting at locations 3 and 7, if the second note of the tie is longer than a quarter note (two half-beats), the second rule will be violated. As a result, we are not allowed to have a note longer than a half note (four half-beats) starting at locations 3 and 7.

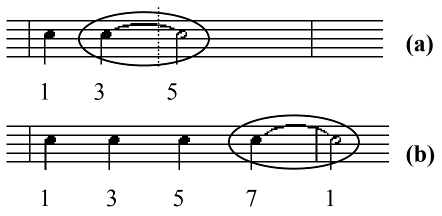

Figure 3. Tied notes

To verify that our Palestrina corpus obeys these rules and to possibly extract other rhythmic information, we annotated the rhythm sequences with measure locations for each event and collected all possible events (notes or rest durations) for each of the eight locations of the measure. As expected, the only event observed at the off-beats was an eighth note (one half-beat) and no notes longer than four half-beats were observed at locations 3 or 7. The results are summarized in Table 1.

Table 1. Rhythmic Restrictions

Location	Allowed Outputs	Number of Legal Outputs
1,5	Anything	15
2,4,6,8	Half-beat note	1
3,7	Note equal to or less than 4 half-beats; Any rest	10

5.2 Modified Models

A naive way to modify the calculation of probabilities in the baseline models is described in this section. As will be argued in section 5.3, this approach does not compute the probabilities from a proper model, but is a quick approximation to the probabilities that would be obtained from a model into which musical information is incorporated. We call these models "modified models" keeping in mind that they are not proper, and in the next section we present a proper model with musical information.

For the N-gram model, we collect N-gram frequencies as usual during the learning phase. However, the probabilistic model is defined differently (b_k denotes the beat after d_{jk}):

$$p_{\text{mod}-3gram}\left(d_{ji} \mid d_{j(i-2)}, d_{j(i-1)}, b_i\right) = \begin{cases} \tilde{p}_{3gram}\left(d_{ji} \mid d_{j(i-2)}, d_{j(i-1)}\right) & \text{if } d_{ji} \text{ allowed at } b_i \\ 0 & \end{cases} \quad (7)$$

where
$$\tilde{p}_{3gram}\left(d_{ji} \mid d_{j(i-2)}, d_{j(i-1)}\right) = \frac{p_{3gram}\left(d_{ji} \mid d_{j(i-2)}, d_{j(i-1)}\right)}{\sum_{d \in b_i allowed} p_{3gram}\left(d \mid d_{j(i-2)}, d_{j(i-1)}\right)} \quad (8)$$

We keep track of the beat and redistribute the probability mass assigned to the outputs that are not allowed as shown in Equation 7. To keep track of the beat, we simply add up the durations of the events in modulo 8, which is the length of a quadruple measure in half-beats. Then, once we identify the location of the next event with respect to its measure, we look at Table 1 to identify the outputs that are allowed. The events that are not allowed have zero probability. Equation 8 ensures that the sum of the probabilities of the events that are allowed is 1.

The results comparing the baseline N-gram models to modified ones are presented in Figure 4. There is an improvement for all cases, but as expected, the gain is smaller with higher order N-grams[5]. An interesting result is that the modified bigram model has a lower perplexity than the plain trigram model. This suggests that adding the musical knowledge can be more beneficial than including one extra previous event in the model.

For the HMM, we use a similar approach: The HMM is

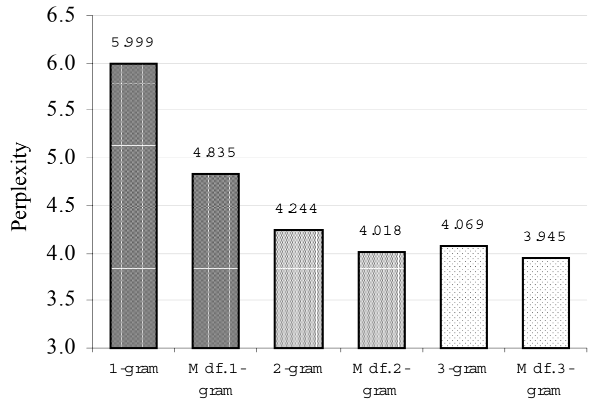

Figure 4. Modified N-gram Model Perplexities

learned as before, but we include the beat in the definition of the probabilistic model. Events that are not allowed given the beat are assigned zero probabilities and the remaining probabilities are normalized so that their sum is 1. The computations are given in Equations 9 and 10.

$$p_{\text{mod}-HMM}\left(d_{ji} \mid d_{j1}, \ldots, d_{j(i-1)}, b_i\right) = \begin{cases} \tilde{p}_{HMM}\left(d_{ji} \mid d_{j1}, \ldots, d_{j(i-1)}\right) & \text{if } d_{ji} \text{ allowed at } b_i \\ 0 & \end{cases} \quad (9)$$

[5] Note that we do not have an exact understanding of how close to the "true perplexity" (TP) the results are. This statement is more meaningful if TP is much smaller than 4, and less if TP is close to 4.

where $$\tilde{p}_{HMM}\left(d_{ji} \mid d_{j1},\ldots,d_{j(i-1)}\right)=\frac{p_{HMM}\left(d_{ji} \mid d_{j1},\ldots,d_{j(i-1)}\right)}{\sum_{d\in b_i allowed} p_{HMM}\left(d \mid d_{j1},\ldots,d_{j(i-1)}\right)} \qquad (10)$$

The results comparing plain HMM to both modified models are presented in Figure 5. Similar to Figure 4, modified HMMs have lower perplexities compared to the plain HMMs, with smaller differences as the number of states is increased. We also see that with about 25 or more hidden states, modified HMMs have lower perplexities than the modified N-gram model.

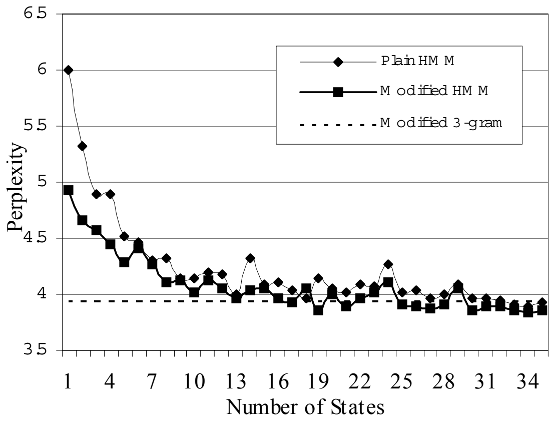

Figure 5. Modified HMM Perplexity

5.3 Informed Models

In the previous models, the musical rules were applied locally at each note or rest without taking into account the ways in which the sequence will continue to be valid. However, this approach is not equivalent to applying musical knowledge to complete sequences and it distorts the probabilities of the sequences. Technically, local application of the rules does not yield probabilities based on maximum likelihood estimates (MLE).

To conserve the relative probabilities of valid sequences, when computing probabilities for a sequence S_n of length n, we should consider the probability that S_n will be generated and also the probability that the model will generate a valid sequence of length n. The probability of a sequence S_n under the informed models, $P_{\text{inf}}(S_n)$, is given in Equation 11:

$$P_{\text{inf}}(S_n)=P(S_n \mid valid)=\frac{P(S_n \cap valid_n)}{P(valid_n)} = \begin{cases} \dfrac{P(S_n)}{P(valid_n)} & \text{if } S_n \text{ is valid} \\ 0 \end{cases} \qquad (11)$$

$P(S_n)$ denotes the probability of S_n under the baseline model. $P(valid_n)$ denotes the probability that a valid sequence of length n is generated from the baseline model.

Unlike the previous approach, this probability model is based on MLE: N-gram frequencies and Baum-Welch algorithm both yield MLE parameters, and invalid sequences are now assigned a zero probability. To illustrate the difference between the modified and the informed models, consider the following case of a sequence of length two, without loss of generality. Let $S=d_1d_2$ be the sequence of interest. Since d_1 is the start of a sequence, all events are allowed. However d_2 is restricted to the events that are allowed at beat b_1, where b_j denotes the beat after event d_j. In Equation 12, $P_{\text{mod}}(S)$ denotes the probability of S calculated as in section 5.4.

$$P_{\text{mod}}(S=d_1d_2)=P(d_1)P_{\text{mod}}(d_2 \mid d_1)=\frac{P(d_1)P(d_2 \mid d_1)}{\sum_{d_i\in b_1 allowed} P(d_i \mid d_1)} \qquad (12)$$

$$P_{\text{inf}}(S=d_1d_2)=\frac{P(d_1d_2)}{P(d_2\in b_1 allowed)}=\frac{P(d_1)P(d_2 \mid d_1)}{\sum_{d_j}\left[P(d_j)\sum_{d_i\in b_1 allowed} P(d_i \mid d_j)\right]} \qquad (13)$$

The denominators of Equations 12 and 13 are not equal: The former is a function only of d_1 whereas the latter has a summation over all possible events. The fundamental difference between the two approaches is that we have a local correction in the modified models, whereas in the informed models the correction is over the whole sequence.

We take the following approach for computing the probabilities under the informed models: The numerator $P(S_n)$ in Equation 11 is readily computed. Computing $P(valid_n)$ is more difficult since it requires considering all valid sequences of length n. We describe the computations for HMM only, since it is the more complex case. Dynamic programming is employed to avoid repeated computations. We denote by $f_i(b,h)$, the probability of getting a valid sequence of length i that starts at beat b and ends in state h. We first compute $f_1(b,h)$ for all beats and all states directly from the initial state probabilities. The remaining probabilities $f_2(b,h)$ through $f_L(b,h)$ are computed using Equation 14. The probabilities $P(h' \mid h)$ and $P(o \mid h)$ are taken respectively from the transition model and the observation model. The sum is over all states and all outputs such that when the duration of the output is added (in modulo 8) to the current beat b, b' is obtained.

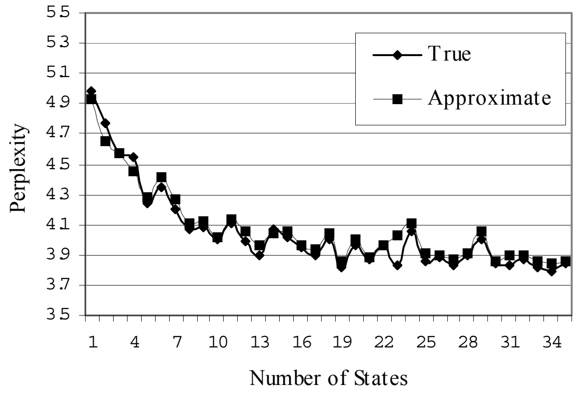

Figure 6. Approximate vs. True HMM Perplexity

$$f_{i+1}(b',h') = \sum_{h,o:b'=b+o} P(h'|h)P(o|h)f_i(b,h) \quad (14)$$

The perplexity of the true model computed as such is actually very close to the approximate perplexity computed in section 5.2, as Figure 6 shows. However, the true model is consistently better than the approximate model.

Figure 7 compares both informed trigram model and HMM to the baselines. It can be seen again that incorporation of musical knowledge is beneficial in both cases and informed HMMs with high number of hidden states have lower perplexities than the informed trigram model.

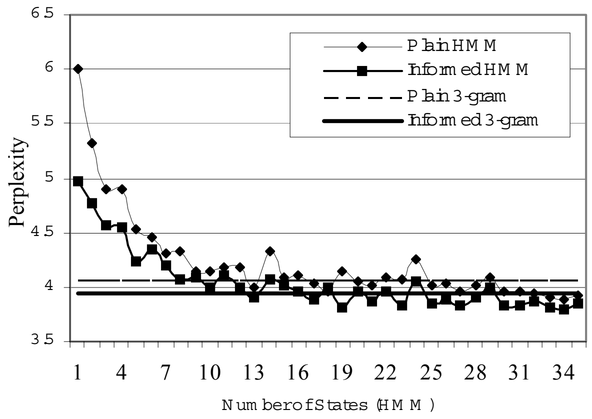

Figure 7. Informed vs. Non-informed Models

5.4 HMM with Beat Classes

The approach of the models in Section 5.3 was to learn a model without using the musical information, and then to apply the rules to reject invalid sequences generated by the model. In this section we present a HMM that takes into account the musical knowledge during the learning phase.

In the HMM with beat classes, we have different submodels each corresponding to a subset of measure locations with similar characteristics. Examining Table 1, we identify three different sub-classes: (1) Locations 1 and 5 corresponding to the beginning and middle of the measure; (2) locations 2,4,6 and 8, which are off-beats; (3) locations 3 and 7 that correspond to the second and fourth beats. As a result, we have three different sub-models. Once such a model is learned, when we are at beat b, we use the sub-model that corresponds to b's sub-class to compute transition and output probabilities. When learning the parameters of the HMM with sub-classes, we keep track of the beat at every point in the sequence and the statistics are collected for the corresponding sub-model. The usual Baum-Welch equations are modified to reflect that the beat is a conditioning variable for both the observation and transition models.

When comparing HMM with subclasses to previous models, we need to account for the difference in the actual number of parameters. Table 2 compares the number of parameters required to learn models with and without sub-classes, given the number of hidden states h. Because of the rules that restrict the outputs according to the beat, the sub-models have different sizes from each other. The observation models for classes 1 and 3 have $15h$ and $10h$ parameters respectively. Observation Model for class 2 has no parameters since the rules require the output to be a half-beat note. Also we only have one initial model (for class 1) since the sequences start at the first beat.

Table 2. Number of Parameters in HMMs

	Ordinary HMM	HMM with sub-classes
Transition Model	h^2	$3h^2$
Observation Model	$15h$	$15h + 10h$
Initial Model	h	h
Total	$h^2 + 16h$	$3h^2 + 26h$

Figure 8 compares the perplexities of all three HMMs: ordinary, informed and sub-classed. The horizontal axis is the number of states in the HMM with subclasses. For the other HMMs, we use Table 2 to compute the number of hidden states that will result in the same number of parameters, and compare models that have the same number of parameters (a weighted average is used if the number of parameters in the HMM with subclasses falls between two numbers of hidden states for ordinary HMMs).

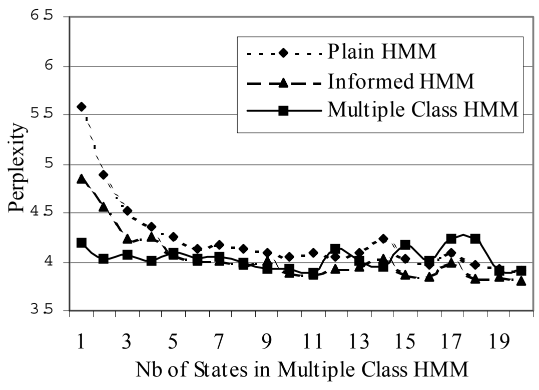

Figure 8. Comparison of HMM perplexities

With small number of parameters, HMM with sub-classes has lower perplexity than the other models. This suggests that when the models are limited, treating sub-classes separately is beneficial and sub-classes act as meaningful separations of hidden states. On the other hand, as the models are sufficiently complex, the perplexities of models become closer and even higher for the HMM with subclasses. A possible explanation is that the training data is not distributed efficiently among sub-models. Each sub-model receives approximately a third of the training data. For instance, sub-model 2 receives an equal share of the training data, which is unnecessary since its observation model is fixed. In the HMMs without sub-classes, training data is shared across beat-classes and this seems to result in models with better (lower) perplexities when we have a sufficiently high number of hidden states. The increasing perplexity for Multiple Class HMM is an indication that with too many hidden states, HMM with sub-classes starts overfitting the training data, which is expected when the models are too complex with respect to the amount of available training data.

6 Conclusion and Future Work

We have presented our work on learning a style-specific model for Palestrina-style rhythm. We suggest that for practical problems such as transcription of music, models augmented with style-specific information can be more successful than generic models. Human listeners bring a lot of stylistic knowledge when performing musical tasks, and it would be limiting to ignore this information in computer systems designed for doing the same tasks. The probabilistic models we implement in this paper are aimed at learning style-specific information that can be extracted from an off-line corpus.

Of course, modeling the rhythm of a single line of music, in the style of a single composer, is only a beginning of the task of developing probabilistic models of music. One next step is to develop a model for single line pitch, while still focusing on a single composer such as Palestrina. The methodology employed in this paper, utilizing well-known temporal models such as hidden Markov models as a basis, and augmenting them with constraints derived from musical knowledge, should carry over quite well to that task. Another direction to extend our work is to model the single-line rhythm of multiple styles. Again, the methodology employed in this paper should work well, though the details of the representation and the constraints will surely be different. A third direction to extend the work in this paper is to model rhythm in a polyphonic composition. Richer models will need to be developed for that task, to capture phenomena such as imitation.

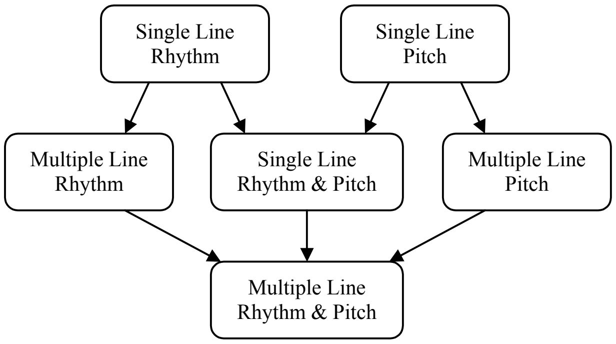

Figure 9. Hierarchy of Music Modeling Tasks

Each of these tasks extends our work here along one dimension. Our long-term goal is to cover all these dimensions, by combining models for pitch and rhythm, including polyphonic models, for multiple styles. A hierarchy of tasks for a single style is shown in Figure 9. In parallel to this hierarchy is another one involving multiple styles. The task of developing general probabilistic stylistic models of music is an important one. While our work in this paper is only a first step towards this task, it lays the foundation and presents the basic methodology for accomplishing it.

7 Acknowledgements

This research was supported in part by a generous grant from the Alfred P. Sloan Foundation.

References

Cemgil, A.T. and Kappen, H. (2003); Monte Carlo Methods for Tempo Tracking and Rhythm Quantization, in *Journal of Artificial Intelligence Research*, 18:45-81, 2003

Chomsky, N. (1965); *Aspects of the Theory of Syntax*, Cambridge, The MIT Press.

Desain, P., and Honing, H. (1994); A brief introduction to beat induction, in *Proceedings of ICMC*, San Francisco.

Fux, J.J. (1725); *Gradus ad Parnassum,* published as *The Study of Counterpoint*, ed. Alfred Mann, Norton, 1943.

Goto, M., and Muraoka, Y. (1998); Music Understanding at the beat level: Real-time beat tracking for audio signals, in *Computational Auditory Scene Analysis*, eds. D.F. Rosenthal and Okuno, H.G., pp.157-176, Lawrence Erlbaum Assoc, 1998

Hiller, L., and Isaacson, L. (1958); Musical Composition with a High-Speed Digital Computer, in *Machine Models of Music*, eds. S.M. Schwanauer and D.A. Levitt, pp 9-21. Cambridge, Mass.: MIT Press, 1993.

Jeppesen, K. (1931); *Counterpoint, the Polyphonic Vocal Style of the Sixteenth Century*, trans. Glen Haydon, Prentice Hall, 1939.

Lerdahl, F. and Jackendoff, R. (1983); *A Generative Theory of Tonal Music*, Cambridge, The MIT Press.

Longuet-Higgins, H. (1976); Perception of Melodies, in *Nature* 263, pp 646-653.

Lopez de Mantaras, R., and Arcos J.L. (2002); AI and Music: From Composition to Expressive Performance, in *AI Magazine*, Fall 2002 pp. 43-57.

Loy, G. (1985); Musicians make a standard: The MIDI phenomenon, in Computer Music Journal, USA Vol. IX/4 (winter 1985) pp. 8-26.

Narmour, E. (1990); *The Analysis and Cognition of Basic Melodic Structures: The Implication-Realization Model*, Chicago: University of Chicago Press.

Pereira, F.C.N. and Riley M. (1997); Speech recognition by Composition of Weighted Finite Automata, in. *Finite-State Devices for Natural Language Processing*, eds. E. Roche and Y. Schabes. Cambridge, MA: MIT Press, 1997.

Rabiner, L.R. and Juang, B.H. (1986); An Introduction to Hidden Markov Models, in *IEEE ASSP Magazine*, 1986, pp. 4-16

Raphael, C. (2001); A Mixed Graphical Model for Rhythmic Parsing, in *Proceedings of 17th Conference on Uncertainty in Artificial Intelligence*, pp. 462-471, Morgan Kaufmann, 2001.

Scheirer, E.D. (1998); Tempo and beat analysis of acoustic music signals, in *Journal of Acoustical Society of America*, 103:1, pp. 588-601.

Shannon, C.E. (1948); A mathematical theory of communication, in *Bell System Technical Journal*, 27(3), pp. 379-423

Simon, H.A., and Sumner, R.K. (1968); Patterns in Music, in *Machine Models of Music*, eds. S.M. Schwanauer and D.A. Levitt, pp 83-110. Cambridge, Mass.: MIT Press, 1993.