Automatic Phrase Continuation from Guitar and Bass Guitar Melodies

Srikanth Cherla,* Hendrik Purwins,† and Marco Marchini**

*Music Informatics Research Group
A309 (College Building)
City University London
10 Northampton Square
London EC1V 0HB, United Kingdom
abfb145@city.ac.uk
†Neurotechnology Group
Berlin Institute of Technology
Fakultät IV, MAR 4–3
Marchstraße 23
10587 Berlin, Germany;
Sound and Music Computing Group
Department of Architecture
Design and Media Technology
Aalborg University, Copenhagen,
Denmark
hpurwins@gmail.com
**Music Technology Group
Communication Campus-Poblenou
Universitat Pompeu Fabra
Roc Boronat, 138
08018 Barcelona, Spain
marco.marchini@upf.edu

Abstract: A framework is proposed for generating interesting, musically similar variations of a given monophonic melody. The focus is on pop/rock guitar and bass guitar melodies with the aim of eventual extensions to other instruments and musical styles. It is demonstrated here how learning musical style from segmented audio data can be formulated as an unsupervised learning problem to generate a symbolic representation. A melody is first segmented into a sequence of notes using onset detection and pitch estimation. A set of hierarchical, coarse-to-fine symbolic representations of the melody is generated by clustering pitch values at multiple similarity thresholds. The variance ratio criterion is then used to select the appropriate clustering levels in the hierarchy. Note onsets are aligned with beats, considering the estimated meter of the melody, to create a sequence of symbols that represent the rhythm in terms of onsets/rests and the metrical locations of their occurrence. A joint representation based on the cross-product of the pitch cluster indices and metrical locations is used to train the prediction model, a variable-length Markov chain. The melodies generated by the model were evaluated through a questionnaire by a group of experts, and received an overall positive response.

Introduction

Research in algorithmic music aims to create interesting music using mathematical models, with the aid of computers for its generation and synthesis. The idea is to view music as a deterministic or stochastic process and to program computers to create new music in accordance with that process. Here, the composer is often only involved to the extent of specifying certain rules or an overall structure that the composition is expected to follow. One such scenario, which is the focus of the present work, is generating music according to style. It involves training models on aspects of musical style such as note patterns, rhythm evolution, and overall structure, and using these trained models to generate stylistically similar music. Probably the most popular example where computers are made to imitate musical style is David Cope's system, Experiments in Musical Intelligence (EMI). EMI analyzes the score structure of a MIDI sequence in terms of recurring patterns (a signature), creates a

Computer Music Journal, 37:3, pp. 68–81, Fall 2013
doi:10.1162/COMJ_a_00184

database of the meaningful segments, and learns the style of a composer, given a certain number of pieces (Cope 1996).

What has come to be known as evolutionary music is based on the fundamental idea of how a genetic algorithm works. The generation process starts with some initial musical data (a piece, melody, or loop in audio or symbolic representation), which is initialized either randomly or based on human input. Then, through the repeated application of computational steps analogous to biological selection, recombination, and mutation, the aim is to produce more musical data of the same nature. GenJam (Biles 1994) is one such system developed for composing jazz solos.

A dictionary-based prediction method for automatic composition is discussed in Assayag, Dubnov, and Delerue (1999). Two dictionaries, namely, the *motif dictionary* and the *continuation dictionary*, are used to represent and continue a given melody. A generation algorithm is used to continue a sequence predicted up to the current point in time. A context variable is maintained that determines the maximum length of the previous sequence to consider while making the prediction. The prediction is based on whether the context matches any of the motifs in the motif dictionary. The continuation dictionary gives the probabilities of various continuations and is used to choose the next symbol.

Among different models that have been applied to learning musical style, Markov chains have been very popular in research owing to the fact that they directly incorporate sequential information into music prediction. Since the seminal work by Hiller and Isaacson (1959), resulting in the *Illiac Suite*, there have been others attempting to do the same in a variety of contexts (Ames 1987). In one of the more recent approaches, hidden Markov models (HMMs) are used for harmonizing Bach chorales (Allan 2002). Here, the visible states are melody notes, and the hidden states are a sequence of chords that would suggest possible harmonizations. The model predicts up to three voices at each time step.

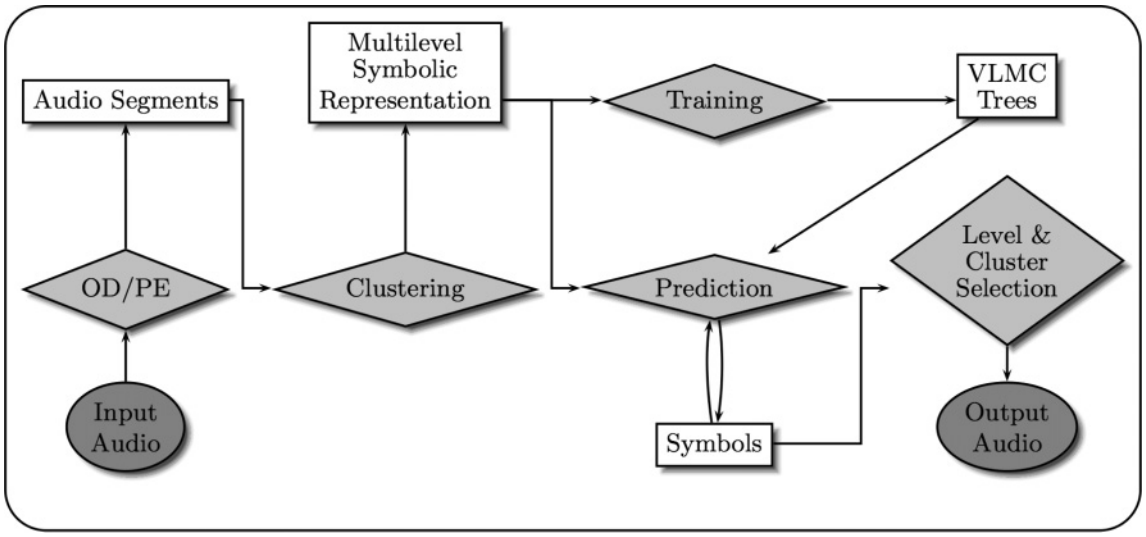
A more elaborate application of HMMs to style-specific music generation can be found in the work of Paiement (2008), where a total of three HMMs are used to different ends. The first one models the underlying rhythm of a MIDI melody, the second the intervallic variations using simplified Narmour features (Schellenberg 1996) given the rhythm, and the third—an Input/Output HMM—predicts pitches that satisfy constraints imposed by an input chord progression and the intervals predicted by the second HMM.

It is well known, however, that training fixed-order Markov chains requires a prohibitive amount of data as order increases. A solution to this problem is used in The Continuator (Pachet 2003). This is a "collaborative musical instrument" that operates mainly on short melodic phrases. A reduction function interprets a given phrase entered using MIDI. Sequences of symbols thus interpreted are parsed using an incremental parsing algorithm to train a variable-length Markov chain (VLMC) that maintains various possible sequences of symbols and their probabilities of occurrence. The system progressively learns new phrases from a musician to eventually develop a more accurate representation of his or her style.

In a similar approach, Marchini and Purwins (2010) developed a system for the analysis of structure and style in a percussive audio sequence with the aim of generating an "arbitrarily long, musically meaningful, and interesting sound sequence with the same stylistic characteristics as the original." By applying several clustering thresholds simultaneously on the values of a regularity measure of the Mel-frequency cepstral coefficients extracted from segmented percussive sounds, a multi-level discrete representation of the sound sequence is obtained. Periodic events are then used to estimate tempo and meter information.

One thing to note about a majority of these approaches is their use of symbolic representations of music such as MIDI or text for reading and representing musical information (Biles 1994; Conklin and Witten 1995; Cope 1996; Allan 2002; Pachet 2003; Paiement 2008). A MIDI-based input is assumed and, hence, the problem of audio segmentation prior to developing a representation is not considered. MIDI is preferred because one may avoid issues related to possible inaccuracy in segmenting audio data and focus solely on the stylistic aspect of music. Moreover, the availability

Figure 1. The generation system first performs onset detection (OD), pitch estimation (PE), and clustering on the melody to create a multi-level symbolic representation of it. These symbols are used to train variable-length Markov chains (VLMCs) at multiple levels, and also as initialization to make predictions with the trained models. This is followed by heuristically selecting the appropriate clustering level and cluster corresponding to each prediction, and output melody generation. The predicted symbols are also reused to predict more of the melody.



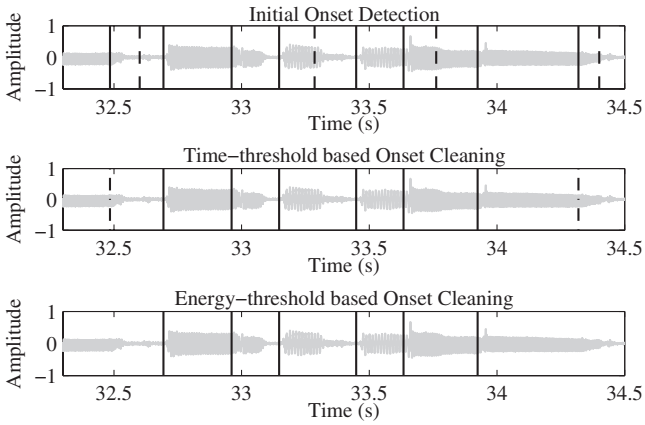
of different MIDI instruments for melody and percussion also makes these approaches feasible. However, it must be noted that, at the same time, the flexibility of such systems is also limited to only these MIDI instruments. There have been a few recent approaches that perform analysis directly on audio data for music generation. The work by Marchini and Purwins (2010) is one such approach for generating rhythmic variations of percussive audio. This system was flexible and robust for audio recorded from several percussive sources such as drums, beat box, etc. In the same spirit, Jehan (2005) used an intermediate minimal data representation directly obtained from the audio signal, based on perceptual listening, for analysis and synthesis. The audio structure analysis system Audio Oracle, also works directly on audio for analysis and generation (Dubnov, Assayag, and Cont 2007).

The framework proposed here (see Figure 1) takes as direct input a monophonic audio signal and segments it into atomic components that serve as a basis for feature extraction. The focus is on pop/rock guitar and bass guitar melodies, with the aim of eventual extensions to other instruments and musical styles. The melody is first segmented into a sequence of notes using onset detection and pitch estimation. Because there is no prior assumption regarding a tuning system, the number and spacing of the scale notes is made here. Hierarchical agglomerative clustering is applied to the estimated pitches that underlie the scale structure, producing a set of multi-level, coarse-to-fine symbolic representations of the melody. This is followed by a novel application of the variance ratio criterion (VRC) (Calinski and Harabasz 1974) to select the appropriate clustering levels. Note onsets are aligned with beats, considering the estimated meter of the melody to create a sequence of symbols that represent the rhythm in terms of onsets and rests and the metrical locations of their occurrence. These symbols are evenly spaced in time (i.e., they are time homogeneous). A joint representation based on the cross-product of the pitch cluster indices and metrical locations is used to train the prediction model: a VLMC. An efficient implementation of this model similar to the one in Pachet (2003) is used here. Finally, a group of experts evaluate the musical output of the system.

Segmentation

The segmentation step converts the input audio data into a sequence of mid-level features that serve as the basis for further analysis. This section describes the steps involved in obtaining an initial description of individual notes that make up the

Figure 2. Different stages involved in onset detection and cleaning for an example segment of an audio signal. The first row shows the unprocessed onset detection output. The onsets denoted by the dashed lines are removed using the time threshold. The second row shows the onset locations after this time-threshold based cleaning. The onsets denoted by the dash-dotted lines are removed using the energy threshold. The third row shows the final onsets, denoted by the solid lines, after filtering.



melody. This is achieved through onset detection and pitch estimation.

Onset Detection

A melody played by an electric guitar or by a bass guitar is first segmented into a sequence of notes or comparably short segments, which form a set of atomic units to be used for generation. A detailed review and comparative study of different algorithms for onset detection has been published by Bello et al. (2005), and some of these algorithms are implemented in the Aubio toolbox (Brossier 2006). In the present work, the Vamp plug-in version of this toolbox is used through the sonic-annotator command-line interface (Cannam 2013). On experimenting with the different methods available for onset detection, it was observed that complex-domain onset detection (Duxbury et al. 2003) performed consistently better than the rest, namely, high-frequency content, phase, and spectral difference methods. Additional post-processing steps are then applied over the detected onsets to filter out false positives produced by Aubio. First, all those onsets that occur in succession with less than 150 msec between them are removed and replaced by the first one in the series. Second, each segment having an average energy less than 40 percent of the entire signal is joined to the segments preceding it. The particular threshold value has been found to work for the instruments used here. These steps are illustrated in Figure 2.

Pitch Estimation

The onset detection step provides possible candidates for notes. As only monophonic melodic are dealt with here, musical pitch is the chosen feature. The YIN pitch estimation algorithm (de Cheveigne and Kawahara 2002) was used for this. Once again, the implementation of this algorithm from the Aubio toolbox was used. Pitch estimation is applied to each segment between two consecutive onsets, yielding one pitch value per segment, using the aubionotes function. Only one pitch is assigned to an entire segment even if the segment contains glissandi. Table 1 lists the parameter values used for onset detection and pitch estimation.

Table 1. Aubionotes Parameters for Note Segmentation

Parameter	*Value*
Pitch type	1 (yinfft)
Step size	512
Block size	2048
Max pitch	12,543 Hz
Min pitch	8 Hz
Onset type	1 (complexdomain)
Peak pick threshold	0.5
Silence threshold	−65
Wrap range	0
Avoid leaps	1

Representation

In this work, melody is encoded using a pitch-and-time representation. First, the sequence of estimated pitch values is quantized using clustering. Second, the sequence of onset and rest times are symbolized by alignment with a beat sequence. Symbol sequences from each of these representations are then combined together to obtain a final sequence that is used to train the VLMC.

Pitch Clustering

In this section, we introduce a new method to generate a pitch representation by unsupervised clustering. Although previous approaches have applied clustering to pitch (Marxer et al. 2008), the novel feature of this work is that the number of clusters (pitches) is determined automatically. Following pitch estimation for each inter-onset segment using the YIN algorithm, a sequence of pitch values (in Hz) are available. These values now need to be grouped together based on similarity. This is required, as any set of frequency values corresponding to the same note are not necessarily identical but are often very close. Moreover, no prior assumption has been made about temperament, instrument tuning, number and pitch of scale notes, or octave information of the given frequency values (in the case of equal temperament, for example, a B4, E5, F♯3, etc.). It is hoped that this freedom from prior assumptions will make our method extensible to any (possibly non-Western) scale intonation system.

First, frequency is transformed into a logarithmic scale (base 2) in order to apply a linear distance measure for clustering pitches. Grouping of similar log-frequency values is realized using agglomerative single-linkage clustering. This yields a dendrogram representing their nested grouping and levels at which groupings change. That is to say, at the bottom of the dendrogram, each leaf node corresponds to an individual log-frequency value. This corresponds to the finest similarity threshold value. At the root node, this value is maximum and all pitches are grouped into a single cluster. This gives a coarse-to-fine (top-to-bottom in the dendrogram) cluster representation of the pitch data. As the log-frequency values are scalar, the absolute value of their difference is used as the distance measure for linkage. Jain, Murty, and Flynn (1999) introduced the clustering algorithm used here and provide a more detailed background on the topic.

Such a clustering method yields clusters at multiple distance-threshold levels. Often these are too many in number, and a subset of these has to be chosen. The task of determining the best number of clusters for a given data distribution is one that has received much attention over the years in a variety of research areas. Milligan and Cooper (1985) reviewed the performance of 30 different criteria for this purpose. One of the most effective criteria in their analysis was the VRC (Calinski and Harabasz 1974), and this was chosen for use in the present case. This method estimates "the best sum-of-squares split" of the dendrogram using the within-group scatter sum (WGSS) and between-group scatter sum (BGSS). The idea is to have clusters that are both well separated and compact (high BGSS and low WGSS, respectively).

An explanation of the VRC for scalar-valued data is as follows. Suppose that there are n log-frequency values $\mathbf{p} = (p_1, p_2, \ldots, p_n)$ corresponding to n segments. Then, the clustering of these values will be given by the partition of $\mathbf{p}$. Without loss of generality, it may be assumed that the mean of the total n pitches is zero. Thus, the total scatter of the n points is given by

$$t = \mathbf{p}^T\mathbf{p} = \sum_{i=1}^{n} p_i^2$$

Now, suppose the n log-frequency values are partitioned into g groups with $n_1, n_2, \ldots, n_g$ values in each group, such that $n = \sum_{i=1}^{g} n_i$. Then, for the k^{th} group, a subset of $\mathbf{p}$, p_{lk} (for $l = 1, \ldots, n_k$) represent the log-frequency values in group $\mathcal{G}_k$. One can now define the scatter for each group $\mathcal{G}_k$ with center of gravity c_k by

$$w_k = \sum_{l=1}^{n_k} (p_{lk} - c_k)^2$$

The pooled WGSS is defined by

$$w = \sum_{k=1}^{g} w_k$$

The BGSS is defined by

$$b = \sum_{k=1}^{g} n_k c_k^2$$

Hence, for each partition (at each clustering level) of the n log-frequency values into g partitions, there exists the identity $t = w + b$ (Friedman and Rubin 1967).

Because the total scatter t is fixed, and a scalar, a natural criterion for grouping is to minimize w. This is equivalent to maximizing b. The VRC uses the values of b and w at each clustering level to find "the best sum of squares split" of the dendrogram by evaluating

$$VRC = \frac{b/(g-1)}{w/(n-g)}$$

The VRC is a decreasing function of the number of clusters which tends to form a local maximum when data points are grouped into natural clusters, with small variation within clusters. It is suggested that those numbers of clusters g (at certain clustering levels) be chosen for which the VRC has an absolute or local maximum, or at least a comparatively rapid increase (Calinski and Harabasz 1974). Depending on the case, either situation might occur. Also, if there are several local maxima, the most economical choice would correspond to the smallest number of clusters g. Following this suggestion, in the present case, local maxima are given first preference, starting with the level containing the lowest number of clusters. If a sufficient number of local maxima do not occur, slope is used.

The cluster levels that are the VRC maxima are sorted in increasing order of the number of clusters (or slope, if that be the case) and the top C levels are chosen for training C VLMCs. In the experiments, a value of $C = 4$ was used. This is done first, as there is no a priori knowledge of the correct number of pitch clusters that actually occur in the melody and also because the VRC only provides an estimate of the best clustering levels. Second, selecting multiple cluster levels also provides us with more patterns at different (coarse-to-fine) levels to learn from the data. It is not necessarily the case that each obtained cluster contains frequencies corresponding to a single note. The similarity threshold corresponding to each selected level determines the frequency precision of clustering. For instance, among the selected levels, the one that contains the least number of clusters is more likely to have grouped a wider frequency range together (that may even correspond to different notes) into the same cluster. Similarly to how adjacent frequencies are quantized into "notes" in different musical traditions, even if the note instances slightly deviate from the ideal note frequency, clustering provides one such quantized representation at each level (henceforth referred to as *note unit*). The number of note units, depending on the melody, typically varied between 3 at the coarsest level up to 30 in the finest level.

Metrical Analysis

Every melody has an underlying rhythm that gives it a certain structure according to a beat or meter. Some of the prior approaches model the rhythmic structure of a melody explicitly to use the information in generating continuations (Pachet 2003; Paiement 2008). In the present work, a time-homogeneous sequence of symbols (symbols that occur at regular intervals of time), which serves as a representation of rhythm, is derived from the set of detected onsets. Each symbol is a beat in a beat sequence and is represented as the cross-product of metrical locations (for example, from the set $\mathcal{M} = \{1, 2, 3, 4\}$, for a metrical unit with 4 equal-length subdivisions) and $\{0, 1\}$ indicating whether an onset exists (1) or not (0) (cf. Table 2). Such a time-homogeneous representation ensures that only onsets with the same metrical locations are treated as equivalent in the Markov chain. The system does not have to detect the beginning of a measure (the "1" in $\mathcal{M}$) correctly, as long as metrical locations are assigned consistently throughout the piece. The method for metrical analysis presented here was adapted to melodies from work by Marchini and Purwins (2010), where this approach was first introduced for drum sounds.

The task at hand is to assign appropriate metrical locations to each of the onsets and rests in the lower row of Table 2. Using a beat-detection algorithm, an initial beat sequence (at an arbitrary metrical level) can be obtained. Some of the onsets would coincide with the beats in this sequence, which corresponds to the metrical location "1". The procedure from here on is to progressively halve the inter-beat interval until at least 90 percent of onsets coincide with beats. In each iteration, the metrical location of the onset that coincides with a beat is noted. A rest (or equivalently, the continuation of a sustained

Table 2. Rhythmic Structure of an Arbitrary Melody

1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
0	X	0	0	0	0	0	X	X	0	0	0	0	X	0	0	X	0	X	X	0	X	0	0

An example of a beat sequence used for determining the underlying rhythmic structure. The Xs and 0s indicate where note onsets are present and absent, respectively. The numbers in the top row are metrical weights.

note) is assumed at every beat where there is no coincidence with an onset. In this representation, a symbol is generated corresponding to every beat (at the final metrical level where at least 90 percent of the onsets match with beats) irrespective of whether an onset exists there or not.

At a homogeneous time instant i, given a metrical location $m_i \in \mathcal{M}$, and a onset-type symbol value $v_i \in \mathcal{V}$, the cross-product rhythm symbol r_i of these two can be written as the ordered pair

$$r_i = (m_i, v_i)$$

This symbolization scheme is implemented at multiple levels of meter length, namely, 1, 2, and 4. At each level, the number of symbols for representing the rhythm is twice as many as the meter length itself, in order to be able to represent either an onset or a rest at each metrical location.

Combined Representation

The final representation is also a cross product between the symbols generated for pitches and those generated for the homogeneous rhythm representation described in the previous section. Consider a set of symbols $\mathcal{R} = \{r_1, \ldots, r_R\}$ that represent R homogeneous metrical symbols. And let $\mathcal{N} = \{n_1, \ldots, n_N\}$, a set of N elements that represents the note units. The combined representation essentially involves generating the set of levels $\mathcal{S} = \{s_1, \ldots, s_S\}$ such that $\mathcal{S} = \mathcal{N} \times \mathcal{R}$. This would result in a total of $S = N \cdot R$ symbols.

Statistical Modeling

A VLMC defines a probability for a symbol $s_j \in \mathcal{S}$, following a symbol sequence $s_{j-1}, \ldots, s_{j-k+1}$ (the *context*), given by the probability distribution $p(s_j | s_{(j-k+1)}, \ldots, s_{j-1})$ where k is the maximal context length that can influence the prediction of s_j. Given a symbol sequence, probabilities of symbol subsequences (of varying lengths) are estimated by their respective frequency counts. These counts are stored in a suffix tree. Starting from a random segment of the original sequence, segments can be iteratively appended by random sampling from the suffix tree, selecting a segment assigned to symbol s_j, following the defined context. Several VLMCs are used in parallel for the statistical analysis of the $C = 4$ combined representation symbol sequences. Ron, Singer, and Tishby (1996) as well as Bühlmann and Wyner (1999) describe a general method for inferring long sequences. Pachet (2003) presents a simplified implementation for faster computation. A suffix tree can be generated in real time for each level of the combined representation. Each node of the tree represents a specific context that has occurred in the past. In addition, each node carries a list of continuation indexes corresponding to segment indexes matching the context. In every iteration of incremental parsing (IP) (Assayag, Dubnov, and Delerue 1999), the minimum-length symbol sequence that has not appeared so far generates only one new node. In contrast, in the VLMC, for all symbol sequences at least as long as this minimal length and up to its maximal length, a new node including a frequency count is generated. Therefore, in the VLMC, long contexts are not automatically split up into substrings as done in IP, but taken as they are. In the VLMC, the generation respects contexts of that proper length. In the beginning, the VLMC builds up a larger tree and therefore provides more detailed information to generate sequences from a short music excerpt.

For audio, a different approach has been applied by Dubnov, Assayag, and Cont (2007). This method does

not require an event-wise symbolic representation, because it uses the factor oracle algorithm. To the best of our knowledge, VLMCs have, to date, only been applied for audio data by Marchini and Purwins (2010), owing to the absence of an event-wise symbolic representation prior to that work.

If a particular level is fixed, the continuation indexes are drawn according to a posterior probability distribution determined by the longest context found. But the question arises as to which level has to be chosen. A trade-off exists between the level and the number of choices that it makes available. Lower levels tend to present fewer continuation choices, resulting in a replication of subsequences in the original melody. On the other hand, higher levels tend to result in random generations owing to an excessive number of choices. Selecting a lower level at which a context of at least $\hat{l}$ exists (for a predetermined fixed $\hat{l}$, usually equal to 6 or 8) works quite well for the examples. But in some cases a context of that length does not exist and the system often reaches the higher level where too many symbols are provided, inducing generations that are too random.

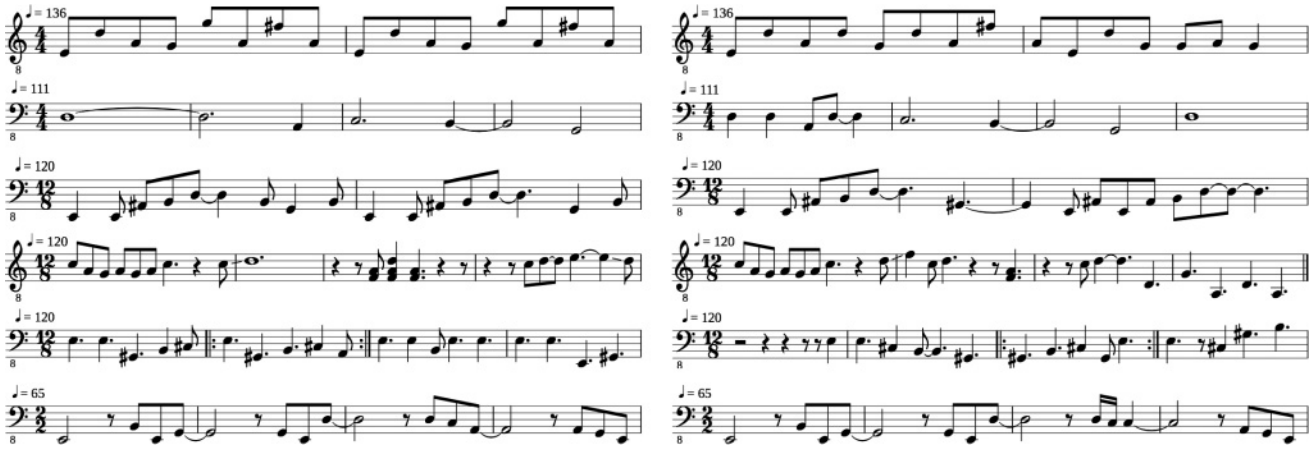
In order to increase recombination of symbols and still provide good continuation, some heuristics are used taking into account the multiple levels available for the prediction. A recombination value p in the range $[0, 1]$ is also set. The following heuristics are used to generate the continuation at each step:

1. Set a maximal context length $\hat{l}$ and compute the list of indexes for each level using the appropriate suffix tree. Store the achieved length of the context for each level.
2. Count the number of indexes provided by each level. Select only the levels that provide less than 75 percent of the total number of symbols, in order to guarantee a minimum of prediction specificity.
3. From these level candidates, select only those that have the longest context.
4. Merge all the continuation indexes across the selected levels and remove the trivial continuation (the next segment).
5. In the case where there is no level providing such a context and the current block is not the last, use the next segment as a continuation.
6. Otherwise, decide randomly, with probability p, whether to select the next segment or, instead, to generate the actual continuation by selecting randomly between the available indexes.

Evaluation

Evaluation of computational musical creativity is an issue that has, in the past, been dealt with in a variety of ways (Conklin 2003). A possible reason for this is that each system realized for music generation highlights one among many views of what may be considered "creative" or "good." Moreover, the existence of a plethora of musical styles, often with a subset that is handled by different approaches, has made it difficult to establish a standard to quantitatively compare these approaches. An evaluation based purely on "accuracy" (e.g., Paiement 2008), favors exact repetition. Such an evaluation measure is not suitable here, because instead of merely reproducing the original, we aim instead at creating variations of the original. Allan (2002) and Dubnov, Assayag, and Cont (2007) evaluate a generative style model using the model itself as a classifier: Trained for a particular style, each generative model assigns a generation probability to a given music excerpt. The style is then determined by the model with maximal probability. The aforementioned measures are objective criteria that measure the stylistic coherence or musical characteristics of the outcome. However, it is not clear to what extent they measure the aesthetic quality of the generated music. For the assessment, e.g., of musical interest, a subjective measure is more suitable. A quasi-Turing test can be utilized that determines to what extent a listener can be led to believe that a melody generated by the system was composed or played by a human (or vice versa). This approach was used in evaluating The Continuator (Pachet 2003). Because sound synthesis is not currently the focus of this work, however, the generated melodies often contain certain artifacts that would make the answer to

Figure 3. Transcribed excerpts from the six input melodies on the left and the corresponding generated excerpts on the right. The excerpts notated in treble clef are for solo guitar, the rest are for bass guitar. (Due to copyright issues, only the guitar melodies are available at soundcloud.com/freakanth/sets/melody-prediction-examples-1.)



this question too obvious. Due to the absence of a reference or benchmark dataset for music generation which would make the performance of different generation systems quantitatively comparable, an alternate evaluation, based on the feedback of a group of musical experts was conducted here. It was decided that those aspects of the system that required evaluation could be focused on through a questionnaire. Experts—individuals with an educational background in music or extensive music performance experience—were approached and asked to provide feedback on the quality of the generated melodies. We consulted a group of ten experts (six male, four female, between the ages of 21 and 42 years), each of whom is a composer or session musician and/or holds a diploma in classical music or jazz.

Database

The analyzed database consists of a variety of recorded pop/rock excerpts from the authors' personal collections. The main focus was on bass lines, with some consideration also given to guitar melodies. Bass lines usually had a riff-like structure in which the same musical phrase is repeated over a short period of time (typically 2 to 5 sec) with minor (if any) variations in each repetition. Two solo guitar melodies and four accompanied bass guitar excerpts were considered. Short excerpts from each melody in the database are shown on the left-hand side of Figure 3.

Onset Detection

A method based on precision recall was used to determine the performance of onset detection. This measures how accurate, and at the same time, how exhaustive a certain retrieval operation was (Brossier 2006). In this context, the data to be retrieved is the set of onsets of the original melody. Informally, a high recall would mean that nothing has been missed but there may be a lot of irrelevant results to sift through (which would imply low precision). High precision means that everything returned was a relevant result, but all the relevant items may not have been found (which would imply low recall). The F-measure is defined as the harmonic mean of the precision and recall.

A detection nearest to a ground-truth onset (i.e., to within 150 msec) was considered to be a match. Table 3 shows the evaluation of onset detection according to the described measures, with an overall precision of 91.25 percent, recall of 87.45 percent, and F-measure of 88.91 percent.

Expert Evaluation

A questionnaire (cf. Figure 4) was prepared and presented to each of the experts as a part of an

Table 3. Performance of Onset Detection

Sequence number	*1*	*2*	*3*	*4*	*5*	*6*	*Overall*
F-measure (before)	0.755	0.899	0.749	0.716	0.731	0.606	0.742
F-measure (after)	0.951	0.954	0.894	0.905	0.776	0.851	0.889
Onset count	60	31	54	65	111	20	331

F-measures are displayed for the six examples, both before and after post-processing. The final column indicates performance for the entire data set.

evaluation package. The evaluation package contained six folders, each with a copy of the questionnaire, an audio file of the original music excerpt, and another file that was generated from the original. In the case of multi-track excerpts, both the original and the generated versions (bass lines, in all cases) were overlaid on other tracks of the original recording. In each case, it was specified in the instructions that the bass line was to be focused on. The solo guitar excerpts were played in isolation without accompaniment. The experts were first asked to listen to the original excerpt any number of times until they developed a fair idea of the melody in terms of tempo, melodic and rhythmic patterns, structure, etc. Following this, they were asked to listen to the generated melody and answer the questionnaire.

Responses to Questions 2 through 6 of the questionnaire have been summarized in Figure 5. Question 1 is not included here as its result was the same ("Yes") across all experts and melodies. On the whole, the responses indicated that the generated music was fairly interesting and coherent with the original in terms of style, motivic material, tempo, and metrical structure. In general, there was agreement in responses from different experts.

One response that all the experts consistently gave for all excerpts was the observability of short note patterns in the generations that also occurred in the original (Question 1). This is encouraging and demonstrates the efficacy of the system in effectively capturing recurring recognizable melodic segments or motifs. When it came to the occurrence of original patterns in the generated melody (Question 2), however, it was pointed out that note accents changed on some occasions, for example in excerpts 1 and 5. That is, certain notes that originally occurred on a strong beat shifted to a weak beat and vice versa. This can be explained as follows. The beat-detection algorithm generated a sequence of beat locations that occurred between approximately equal time intervals. Metrical levels corresponding to these beats at higher resolutions were generated by linear interpolation of these detected beat locations which also occurred between approximately equal time intervals. Because, at a higher resolution, the interpolated beats were very close together, on certain occasions a note onset was matched wrongly with a generated beat adjacent to the actual beat to which it corresponded. Accordingly, during generation these beats were reproduced at what were considered by some experts to be inappropriate metrical locations.

In the case of the tempi of the generated melodies (Question 3), nearly three-quarters of all responses agreed that they were the same as those of their respective originals. It is interesting to see that even in those cases where the tempo of generated music was considered not to be the same as that of the original, the tempo was either "something else" (regular, but not the same) or "not determinable" (irregular). This happened with the third and fifth excerpts. Also, in those cases where the tempo of the generated music and the original were not considered to be the same, the experts pointed out that melodic patterns were occurring in inappropriate metrical locations (or broken riffs) in the generation. As pointed out earlier, one of the reasons for this could be the shift in note accents. Also, some of the generated melodies suffered from a drawback that note onsets differed from the accompaniment by a slight temporal offset. This is, once again, due to the approximate regularity of the beat-detection algorithm. It may, have been the case that, owing to the delayed or premature reproduction of certain onsets, keeping track of the tempo in case of some of the melodies was not straightforward. This was

Figure 4. The questionnaire presented to the experts for their feedback. This questionnaire was to be answered for each of the pairs of original and generated melodies. The text of the second question shown here was for melodies that were guitar or bass solos; in the case of riff-like melodies, we asked if the melodic patterns occurred in such a way that a riff structure was evident. The sixth question was only applicable to those songs with accompaniment.

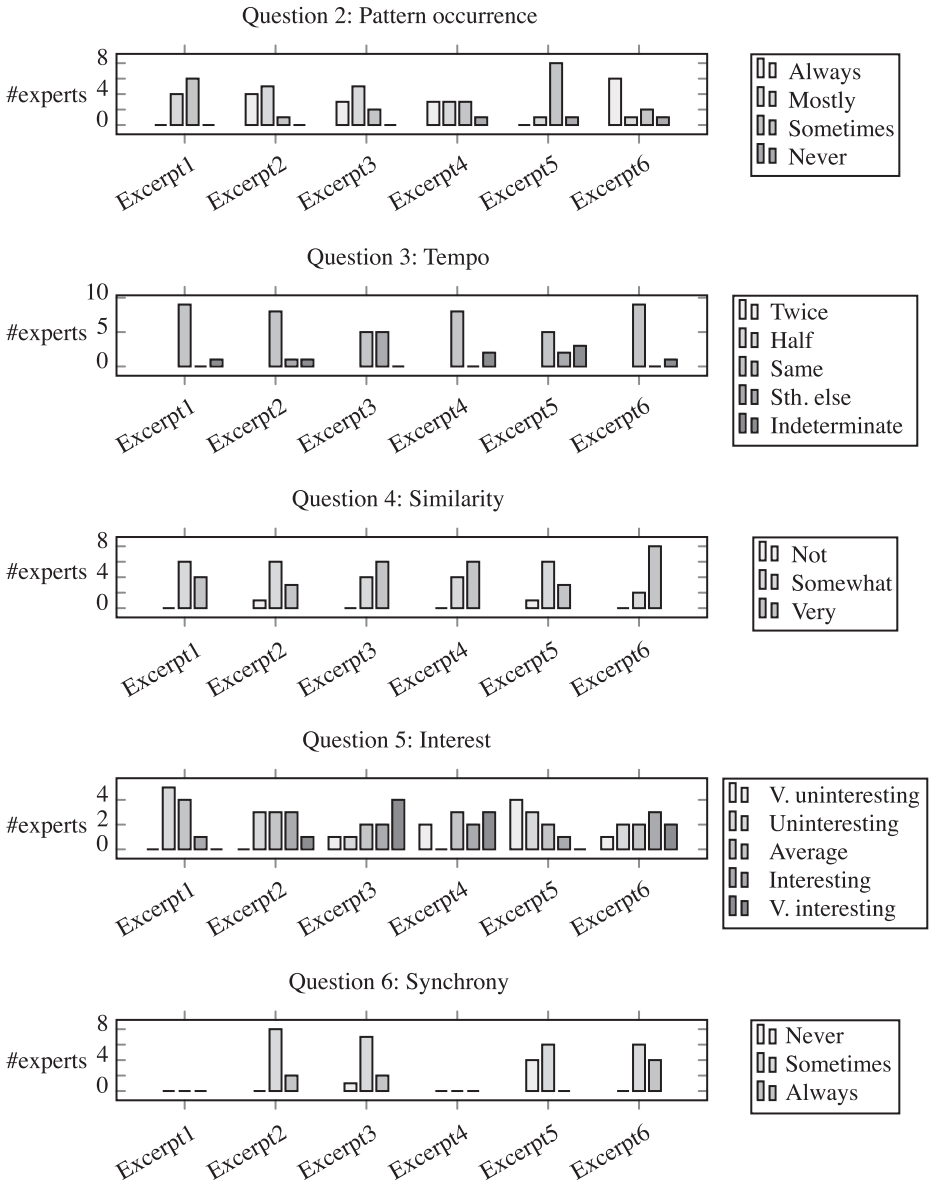
EVALUATION QUESTIONNAIRE

1. Patterns: The generation contains recognizable melodic patterns from the original melody.
   (a) Yes.
   (b) No.
2. Pattern Occurrence: If you answered "Yes" to question 1, are these melodic patterns occurring in metrically meaningful locations?
   (a) Always.
   (b) Sometimes, and more often appropriately.
   (c) Sometimes, but more often inappropriately.
   (d) Never.
3. Tempo: The tempo of the generated melody, on an average, when compared to the original is nearly
   (a) Twice as fast.
   (b) Half as fast.
   (c) The same.
   (d) Something else.
   (e) Cannot be determined.
4. Similarity: Does the generation sound similar in (melodic) style to the original?
   (a) Not similar.
   (b) Somewhat similar.
   (c) Very similar.
5. Interesting: How interesting is the generation (in terms of generating new melodic/rhythmic patterns)? Please rate on a scale of 1 (very uninteresting) to 5 (very interesting).
   (1) (2) (3) (4) (5)
6. Synchrony: Is the generated melody synchronized with the accompaniment?
   (a) Always.
   (b) Sometimes.
   (c) Never.
7. Do you have any additional comments? Please write them down in the space below. It would be particularly interesting to know what type of similarities/differences you noticed between the original and the generation.

the case particularly with the generated versions of the third and fifth excerpts, both of which were considered to lack synchrony with their respective accompaniments (Question 6).

The question on overall similarity between the generation and the original received a fairly good feedback with all responses being either "very similar" or "somewhat similar." It must be noted, however, that in certain cases (excerpts 2 and 6), the generated versions were found to be too similar to the original. On revisiting generation parameters for these melodies, we found that the generated

Figure 5. Summary of the ten experts' responses to the evaluation questionnaire (Figure 4). Question 1 has been left out as all the responses to it were the same (Yes).



version of excerpt 2 used a very long context (32 symbols) as a result of which there were few note choices left following such a long context. Despite the long context, some degree of variation still occurred. In the case of excerpt 6, the riff itself was composed of few notes and with minimal variations. Moreover, the riff structure and rhythmic evolution of the generation in this case are almost identical to that of the original, and when played with the accompaniment could have created such an impression.

The generation process used here, namely, the VLMC, is more suitable in the case of short melodic segments than for long solos. The main problem with longer melodies is that elements reflecting long-term coherence, such as repetition of segments

from time to time and logical transitions between shorter segments of a melody, are not evident. This was highlighted in some of the comments received for the fourth and fifth excerpts. One contained a minute-long guitar solo, and the other was a walking bass line without obvious repetitive structure. As the temporal scope considered by the system was limited to a relatively short duration, it does not take into account the long-term evolution and structure of these melodies.

In general, the experts found those generations interesting that had a regular rhythm and noticeable pitch variations from the original. It was often the case that the same variation was considered to be inappropriate or abrupt by one expert, and interesting by another. The generated versions of excerpts 3 and 4 received better feedback and ratings than the others, consistently from all the experts. It would help to use these excerpts as a reference in the future. It was also appreciated that the system was able to reproduce even silent pauses from the original melody on several occasions. The average interest rating for excerpts 1 through 6 were 2.3, 3.2, 3.7, 3.4, 2.0, and 3.3 (out of 5.0), respectively. All of the experts found at least two generations out of the six genuinely interesting and expressed the opinion that, with some minor improvements in synthesis and overall structure of the generated version, the others could also sound much better.

Conclusions and Future Work

The present work addresses the much ignored problem of generating stylistically similar melodies directly from audio, instead of from symbolic data (MIDI, MusicXML, etc.). This is, in general, a more difficult problem due to the occurrence of segmentation errors that tend to propagate into any symbolic representation that is in use for generating music. A multi-level representation, similar in spirit to that used by Marchini and Purwins (2010), was used here for the symbolic representation of notes in the melody and their metrical locations.

In particular, the representation adapts to the style of the melodic excerpt. A particular scale (e.g., C major) is not assumed, and neither is a particular tuning (e.g., equal temperament) or the use of a particular subset of the diatonic scale of seven notes. The utilized numbers of note units are chosen automatically using the clustering level selection method by Calinski and Harabasz (1974). Unlike a music generation system based on audio-to-MIDI transcription as the first stage, this system could be made to meaningfully adapt to non-equal temperaments such as in Greek, Ottoman, Arabic, or African music. Moreover, the multi-level representation provides a coarse-to-fine abstraction of the melody and enables different levels of precision while choosing a continuation pitch. It allows the system to simplify the melodic structure of the example and to generate musical continuation even from short melodic examples.

Although a pitch-based representation for symbolizing audio data is used here, others, like those based on intervals or melodic contours, may be incorporated, possibly generating different musical output. It is worth noting that the present method for symbolizing rhythm does not take into account anything other than binary (e.g. triple, quintuple) and compound meters, and changing tempo (accelerando and ritardando). Introducing other than binary metrical divisions and applying the method locally would yield a more general method to handle these cases. It was also observed during evaluation that there was no single value for context length that would suit all the examples. A possible cue, it seemed, that could be explored to automatically estimate its value, is the tempo of the melody. Our method, which seemed more suitable for short, riff-like melody generation, may be improved to handle longer solistic melodies. One improvement would be to incorporate methods that segment the melody into regions based on higher-level similarity to help reproduce its global evolution (Mozer 1994), and use what could be short motifs that are generated by the VLMC locally. Another improvement over the current system would be the possibility of automatically determining the number of selected clustering levels dynamically, depending on the melody. With these improvements, the present approach can also eventually be extended to melodies of other musical styles played by different instruments. The present generative model for bass riffs and guitar

melodies sheds light into some aspects of musical creativity, but it is only a small step towards the goal of a comprehensive understanding of human musicianship.

Acknowledgments

Hendrik Purwins was supported, in part, by the German Bundesministerium für Bildung und Forschung (BMBF), grant BFNT 01GQ0850.

References

Allan, M. 2002. "Harmonizing Chorales in the Style of Johann Sebastian Bach." Master's Thesis, School of Informatics, University of Edinburgh.

Ames, C. 1987. "Automated Composition in Retrospect: 1956–1986." *Leonardo Music Journal* 20(2):169–185.

Assayag, G., S. Dubnov, and O. Delerue. 1999. "Guessing the Composer's Mind: Applying Universal Prediction to Musical Style." In *Proceedings of the International Computer Music Conference*, pp. 496–499.

Bello, J. P., et al. 2005. "A Tutorial on Onset Detection in Musical Signals." *IEEE Transactions on Speech and Audio Processing* 13(5):1035–1047.

Biles, J. 1994. "GenJam: A Genetic Algorithm for Generating Jazz Solos." In *Proceedings of the International Computer Music Conference*, pp. 131–137.

Brossier, P. 2006. "Automatic Annotation of Musical Audio for Interactive Applications." PhD thesis, Centre for Digital Music, Queen Mary, University of London.

Bühlmann, P., and A. J. Wyner. 1999. "Variable Length Markov Chains." *The Annals of Statistics* 27(2):480–513.

Calinski, T., and J. Harabasz. 1974. "A Dendrite Method for Cluster Analysis." *Communications in Statistics—Theory and Methods* 3:1–27.

Cannam, C. 2013. "Sonic-Annotator." Available online at omras2.org/SonicAnnotator. Accessed April 2013.

Conklin, D. 2003. "Music Generation from Statistical Models." In *Proceedings of the AISB Symposium on Artificial Intelligence and Creativity in the Arts and Sciences*, pp. 30–35.

Conklin, D., and I. H. Witten. 1995. "Multiple Viewpoint Systems for Music Prediction." *Journal of New Music Research* 24(1):51–73.

Cope, D. 1996. *Experiments in Musical Intelligence*. Madison, Wisconsin: A-R Editions.

de Cheveigne, A., and H. Kawahara. 2002. "YIN, a Fundamental Frequency Estimator for Speech and Music." *Journal of the Acoustic Society of America* 11(4):1917–1930.

Dubnov, S., G. Assayag, and A. Cont. 2007. "Audio Oracle: A New Algorithm for Fast Learning of Audio Structures." In *Proceedings of the International Computer Music Conference*, pp. 224–228.

Duxbury, C., et al. 2003. "Complex Domain Onset Detection for Musical Signals." In *Proceedings of the Conference on Digital Audio Effects*, pp. 90–93.

Friedman, H. P., and J. Rubin. 1967. "On Some Invariant Criteria for Grouping Data." *Journal of the American Statistical Association* 62(320):1159–1178.

Hiller, L. A., and L. M. Isaacson. 1959. *Experimental Music: Composition with an Electronic Computer*. New York: McGraw-Hill.

Jain, A. K., M. N. Murty, and P. J. Flynn. 1999. "Data Clustering: A Survey." *ACM Computing Surveys* 31(3):264–323.

Jehan, T. 2005. "Creating Music by Listening." PhD thesis, MIT Media Lab, Massachusetts Institute of Technology.

Marchini, M., and H. Purwins. 2010. "Unsupervised Generation of Percussion Sound Sequences from a Sound Example." In *Proceedings of the Sound and Music Computing Conference*, pp. 477–484.

Marxer, R., et al. 2008. "Dynamical Hierarchical Self-organization of Harmonic and Motivic Musical Categories." *The Journal of the Acoustical Society of America* 123(5):3800. Available online at http://asadl.org/jasa/resource/1/jasman/v123/i5/p3800.s4. Accessed July 2013.

Milligan, G. W., and M. C. Cooper. 1985. "An Examination of Procedures for Determining the Number of Clusters in a Data Set." *Psychometrika* 50(2):159–179.

Mozer, M. C. 1994. "Neural Network Music Composition by Prediction: Exploring the Benefits of Psychoacoustic Constraints and Multi-Scale Processing." *Connection Science* 6(2–3):247–280.

Pachet, F. 2003. "The Continuator: Musical Interaction with Style." *Journal of New Music Research* 32(3):333–341.

Paiement, J.-F. 2008. "Probabilistic Models for Music." PhD thesis, Idiap Laboratory, École Polytechnique Fédérale de Lausanne.

Ron, D., Y. Singer, and N. Tishby. 1996. "The Power of Amnesia: Learning Probabilistic Automata with Variable Memory Length." *Machine Learning* 25(2–3):117–149.

Schellenberg, E. G. 1996. "Expectancy in Melody: Tests of the Implication–Realization Model." *Cognition* 58:75–125.