Journal of Information Processing Vol.27 693–700 (Nov. 2019)

[DOI: 10.2197/ipsjjip.27.693]

Regular Paper

Automatic Music Completion Based on Joint Optimization of Harmony Progression and Voicing

Christoph M. Wilk[1,a)] Shigeki Sagayama[1,b)]

Received: January 31, 2019, Accepted: September 11, 2019

Abstract: This paper proposes automatic music completion - the automatic generation of music pieces from any incomplete fragments of music - as a new class of music composition assistance tasks. This is a generalization of conventional music information problems such as automatic melody generation and harmonization. The goal is to turn musical ideas of a user into music pieces, allowing users to quickly explore new ideas and enabling inexperienced users to create their own music. This principle is applicable to a wide variety of music, and as a first step, we present a system that automatically fills in missing parts of a four-part chorale, as well as the underlying harmony progression. The user can input any combination of melody fragments, and freely constrain the harmony. Our system searches for harmonies and melodies that adhere to music-theoretical principles, which requires extensive knowledge and practice for human composers. Accounting for the mutual influence of melodic and harmonic development in music composition, the system is based on a joint model of harmony and voicing. The system was evaluated by analyzing generated music with respect to music theory, in addition to a subjective evaluation experiment. The readers are invited to experiment with our system at http://160.16.202.131/music_completion.

Keywords: automatic music completion, probabilistic model, harmony progression, polyphonic voicing

1. Introduction

Artificial intelligence is becoming increasingly popular in creative art. In particular, algorithmic music composition has been a topic of research for several decades. However, instead of autonomously generating music and replacing a human composer, our motivation is to react to user input, with the aim of supporting human creativity and realizing fruitful collaboration between human and computer. In previous research, multiple automatic music generation systems were designed to process some sort of input. In most cases, however, the type of information that a user can input is very limited. For example, a typical problem is melody harmonization [1], [2]: Users insert a melody and the computer returns melodies of the accompaniment. However, what if the user has ideas for two melodies and wants the system to add more, or what if the user would like to hear their favourite harmony at a specific position? Furthermore, harmonization algorithms usually cannot handle incomplete input melodies.

Therefore, we want to proceed in a new direction in algorithmic music composition, which we call "automatic music completion," emphasizing the freedom of user input. Our goal is a system that is able to generate music based on a wide variety of musical ideas of the user, such as interesting melodic motifs, rhythmic patterns, incomplete melodies of multiple instruments, partial harmony progressions, and any combination thereof. Music completion is the problem of filling in all missing elements, be it completing one or multiple melodies, deriving a harmony progression, adding additional melodies, or everything at once. This is a generalization of several music information tasks, e.g., an automatic music completion system could solve the problems of melody generation and harmonization. However, it goes even further by handling incomplete input in multiple domains and solving several sub-problems simultaneously. Such a flexible system can make music composition more accessible to users with limited music-theoretical knowledge, and enables composers to quickly explore new musical ideas, but it requires working towards a complete model of music that accounts for a multitude of musical aspects such as melody, harmony, rhythm and voicing.

As a first application of the principle of automatic music completion, we discuss the composition of four-part chorales, which is a popular discipline of classical music. These chorales are polyphonic music pieces for choirs or instrumental ensembles with a strong focus on the relationship between the typically four parts soprano, alto, tenor and bass. A multitude of music-theoretical rules make chorale composition a complex task, and a typical problem for music students to solve. An exemplary result of our music completion system is shown in **Fig. 1**.

2. Related Research

Automatic music composition has been a topic of research since the automatic generation of the Illiac suite [3] in the 1950s. In the following years, a wide variety of methods have been applied in this field of research. For an overview we refer to a survey of several hundred publications by Fernández and Vico [4].

Many of these publications present autonomous music composition algorithms, which generate music largely independent of

[1] Graduate School of Advanced Mathematical Sciences, Meiji University, Nakano, Tokyo 164–8525, Japan
[a)] wilk@meiji.ac.jp
[b)] sagayama@meiji.ac.jp

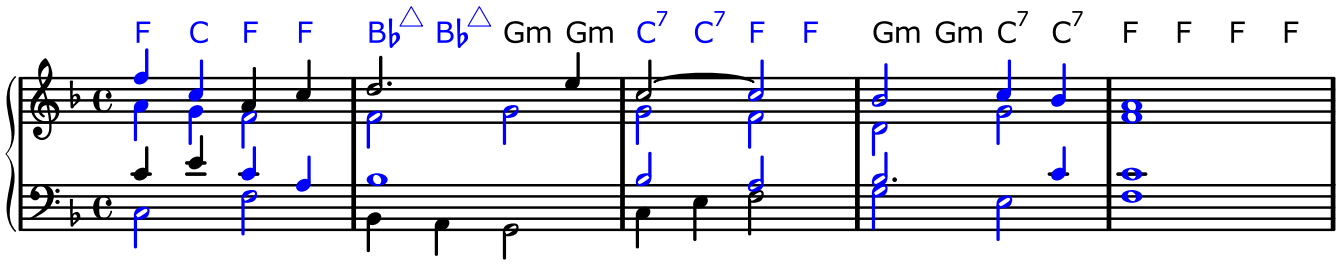

Fig. 1 A result of the automatic music completion system. Notes and harmonies input by the user are shown in black, and the output of the system in blue. While not illustrated, it is possible to specify multiple harmony candidates for each beat, of which the algorithm will chose the optimal one.

the user input. Examples are David Cope's Experiments in Music Intelligence [5], the Melomics music database [6], which was generated using a genetic algorithm, Kulitta [7], a composition framework based on a formal grammar, and Google Magenta's recurrent neural network which generates human-like piano performances [8]. While some autonomous automatic composition systems are quite popular, they aim at completely replacing human creativity.

On the other hand, the goal of automatic composition assistance systems is to support human creativity. They generally allow a user to input some kind of musical idea, and generate results based on this input. An example is the composition system Orpheus [9], which uses an hidden Markov model to generate melodies based on user input, and provides a web-based user interface that has been accessed by several thousand users. Researchers have published many more algorithms for user-driven automatic melody generation, some constrained by abstract parameters [10], others by input melodies in order to generate variations of it [11], to interpolate missing parts [12], or to improvise a fitting countermelody [13].

Complementary to constrained melody generation, melody harmonization describes the problem of adding harmonies to input melodies. One type of harmonization algorithm computes abstract harmonies, resulting in lead sheets, which consist of a single melody and an accompanying harmony progression. Methods to obtain such lead sheets include genetic algorithms [14] and Markov models [15]. The other type of harmonization algorithm computes voicings, i.e., individual notes for additional voices and instruments, which are usually subject to music-theoretical rules. For an overview over approaches to automatic harmonization, we refer to a survey by Pachet and Roy [1].

However, the goal of our research is to not only compute harmonies for melodies or vice versa, but to allow the user to input any combination of partial harmonies, melodies and voicings. Similar to our approach in that respect is a system called FlowComposer [16], that assists a user in generating lead sheets. The system allows users to constrain both melody and harmony, and generates results using a Markov model.

The two approaches closest to ours are the four-part chorale generation systems DeepBach [17], which uses recurrent neural networks, and the similar Coconet [18], which is based on a convolutional neural network, and was developed in collaboration with Google's Magenta project. The aim of both systems is to imitate the composer J. S. Bach, which is achieved by random sampling. While not their focus, both systems can process relatively free input for four voices, but they do not allow users to constrain harmony and are limited to Bach-like music pieces.

3. Automatic Music Completion

3.1 New Direction: Free User Input

Our motivation is to provide automatic music composition assistance with as much freedom to input musical ideas as possible, which is mainly influenced by two factors.

(1) Allowing input of any size: No input at all, an almost complete melody with a short section missing, or multiple melody fragments.

(2) Providing multiple modes of input, e.g., pitch, harmony, rhythm, structure, or tuning parameters that allow a user to intuitively influence the automatic generation process.

Therefore, an ideal system would be able to compute solutions without input, but also when subject to complex constraints in multiple domains, including input that might be rare or unseen in music data sets. This requires a powerful and flexible model covering various aspects of music composition.

In this paper, we present a model that accounts for multiple factors that influence the composition of four-part chorales. Instead of imitating a specific composer, such as J. S. Bach in related research [17], [18], our aim is to devise a model that does not extract composer-specific characteristics, but music theoretic principles that are valid for a wider range of music. While this paper focuses on chorales, our approach can be applied to various musical styles. By training individual parts of the model on different data sets, we could, in principle, combine classical polyphony with jazz harmonies, for example.

The flexibility of our approach enables the system to target a wide range of users. Our user interface was designed such that even beginners with very little music knowledge can intuitively input melodies in a piano roll format and then let the system generate the underlying harmony progression and all missing notes of the four voices. However, the type of user that can benefit most from our system would have basic knowledge of chords in order to use them as constraints, as well as enough musical intuition to insert notes in more than one voice (e.g., a countermelody). For this type of user, our system can display its full potential by simultaneously responding to various different user input constraints.

3.2 The Complexity of Polyphonic Music

Musicians have derived a multitude of rules for composing

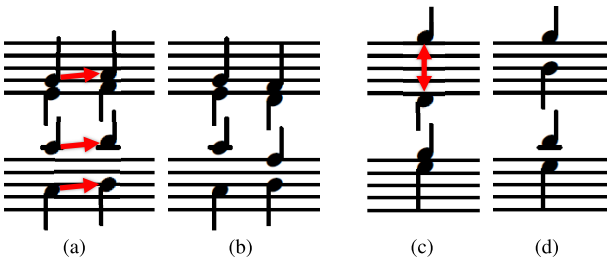

Fig. 2 Two exemplary voicing rules. Parallel fifths and octaves, i.e., the same motion of two voices into a perfect fifth or octave as shown in red in (a) should be avoided (as in (b)). Large distances between voices as shown in red in (c) should also be avoided (as in (d)).

polyphonic music [19]. Examples of such rules are illustrated in **Fig. 2**. However, most of these rules are not absolute, and sometimes broken by composers. The rules can also conflict with each other, e.g., avoiding parallel fifths and octaves (Fig. 2 (a)) can result in large intervals between two voices (Fig. 2 (c)). The problem becomes more complicated when voicing is constrained by user input. This is especially important for freedom of user input, which significantly increases when allowing users to break rules. Instead of simply avoiding rule violation, we want to be able to determine, for example, how to best complete a voicing in which the user manually inserted a parallel fifth. In a sense, the problem of polyphonic voicing is similar to solving a puzzle. While this requires a lot of knowledge and practice for humans, computers are well-suited for exploring solution spaces. Therefore, we apply an optimization algorithm to a probabilistic model designed to balance various aspects of music theory.

Another difficulty is the modeling of nonharmonic tones. These are melody notes that do not belong to the current chord, complicating the relation between voicing and underlying harmony. A voicing can result from different harmonies depending on which of its notes are interpreted as nonharmonic. This is a common problem in harmony analysis [20], and also in our inverse task of automatically generating voicings from harmonies. Ignoring the possibility of nonharmonic tones would limit user input [21], while simply assuming that harmonies produce nonharmonic tones with a certain probability can result in dissonance [22]. The algorithm presented here approaches this problem by jointly optimizing voicing and harmony, thus comparing different harmonic interpretations of notes depending on harmonic and melodic context. To avoid dissonance, the algorithm considers how different classes of nonharmonic tones are resolved according to music theory [23] (see Section 4.6).

4. Four-Part Chorale Model

In order to account for the inter-dependency of harmonic and melodic development [19], we implemented a model of the joint probability $P(H, V)$ of harmony sequence H and voicing sequence V. Before explaining our mathematical model, we discuss the challenges of computational cost and data sparsity in the context of polyphonic music.

4.1 Computational Cost

The number of possible harmony and note combinations is very large. We denote the range of a voice (number of notes a particular voice can sing) as r and the number of harmony candidates as c. A brute force algorithm exploring all possibilities of a four-part chorale with n time steps would have a computational cost of $O((c\,r^4)^n)$. We implemented an efficient beam search algorithm, which reduces the complexity to $O(c\,r^4\,n\,w)$ for beam width w. The ideal value for w strongly depends on the user input. Without user input, $w = 1$ generally suffices. However, to properly account for user input constraints, a wider range of possibilities has to be explored. For example, when the user inserts a low note at the first bar of the piece and a high note at the nth bar, the state at bar $n-1$ with the highest probability (i.e., retained with $w = 1$) likely also contains a low note, since small step movement is highly probable. This would result in a large and highly improbable jump from this low note to the high user input note at bar n. With sufficiently high beam width w, the algorithm can consider states at bar $n-1$ that contain higher notes, which would eliminate the jump and thus lead to overall more likely results. In our experiments, we allowed users to tune the beam width, and most users chose values between 100 and 1000. In our music theoretic evaluation (Section 5.1) $w = 100$ was used.

However, considering that typical values are $c > 100$ and $r \approx 40$ (for Bach chorales), the cost is generally still too high, since our system is meant for interactive composition assistance, where computation should finish within several seconds. Simply reducing the search space according to strict rules would strongly limit the freedom of user input. Instead, we allow the user to input any note within the range of a voice, and only heuristically restrict which note candidates the algorithm considers for filling in the gaps:

- Voice crossings: The algorithm does not insert notes that are lower than that of a voice below and vice versa. Motivation: Would sound very similar when interchanging notes. Restriction on user input: Weak; e.g., will not yield solutions when a tenor note is higher than a soprano note in the same voicing, which is possible, but very unusual.
- Large melody intervals: Only intervals up to octaves are considered. Motivation: Large jumps would create the impression of two separate melodies. Restriction on user input: Slight; will not yield solutions when two notes with only one time step in between are more than two octaves apart, which is very uncommon and very difficult for singing voices.
- Large inter-voice intervals: Only intervals up to tenths are considered. Motivation: Voices would sound disconnected. Restriction on user input: Slight; similar reason as above.
- Repeated note doubling: The algorithm inserts at most two notes of one pitch class into a single voicing. Motivation: Further doubling would result in incomplete and empty sounding chords. Restriction on user input: None; at least three pitch classes are available to choose four notes from; however, slightly restrictive in combination with voice crossing avoidance, e.g., will not yield solutions when a soprano note is identical to a tenor note, which is slightly stricter than the restriction when only avoiding voice crossings.
- Many nonharmonic tones: At most one nonharmonic tone of the current key's scale is inserted per voicing. Motivation: A high number of nonharmonic tones implies that another

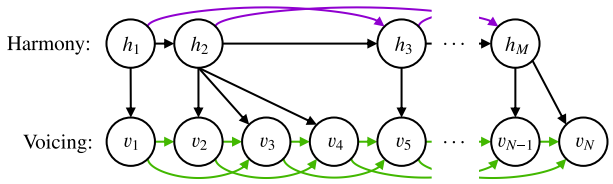

Fig. 3 A graphical representation of the model used in our system. Harmonies h_i are treated as hidden states of observable voicings v_j. Harmonies can be of variable length to account for harmonic rhythm. In addition to dependencies of a conventional hidden Markov model (black), we consider larger harmonic contexts (purple) and dependencies between consecutive voicings (green) in order to capture properties of the voices' melodies.

harmony is likely to be of significantly higher probability.

Restriction on user input: None; same reason as above.

These heuristics generally avoid uncommon musical patterns and thus in most cases do not affect optimality of search results. They also coincide with principles of music theory.

4.2 Data Sparsity

Compared to the large amount of possible note combinations, only a very small subset is contained in training data. This would be less of a problem without user input, since the search space could be limited to common musical patterns more easily. However, we want to allow the user to input ideas that do not occur in data, and still be able to react meaningfully to it. On the other hand, if the model is too simple, the generated music will be equally simple. Imagine the same harmony continuing throughout the whole piece. Thus, we need to efficiently extract relevant information from small amounts of data.

Still, the size of data sets of four-part voicings and harmony progressions is quite limited. In particular, the largest data set available to us that contains information on both harmony and voicing comprises only about 40 four-part chorales with harmony annotation [24]. We also had access to around 400 four-part chorales without harmony annotation [25], and harmony annotation of around 80 classical pieces, but without four-part voicing [26]. Since the amounts of data are small, our model was designed to be able to extract information from different data sets.

4.3 Joint Model and Optimization Objective

The model follows the concept that harmonies are hidden states of observable voicings [27], [28], which are arrangements of multiple notes according to the underlying harmony. This means that an individual harmony can result in many different note combinations, including nonharmonic tones. A graphical representation of the model is shown in **Fig. 3**. In addition to dependencies of a conventional hidden Markov model, we account for extended harmony context, considering the trade-off between combinatorial complexity and the ability to capture characteristic chord sequences, and allow harmonies to be of different lengths in order to model the harmonic rhythm. Furthermore, the model captures dependencies between consecutive voicings in order to model melodic development, which allows accounting for principles of music composition concerning melodies. However, this considerably increases the complexity of the model, which is why we discuss how to capture important contextual information while reducing the problem of data sparsity in the following sections.

Denoting a chorale's harmony progression as $H \equiv (h_1, \ldots, h_M)$ and its sequence of voicings as $V \equiv (v_1, \ldots, v_N)$, the objective of obtaining the optimal sequences H^* and V^* can be formulated as follows.

$$H^*, V^* = \arg\max_{H,V} \prod_i \left(P(h_i \,|\, h_{i-1}, h_{i-2}) \prod_j P(v_j \,|\, v_{j-1}, v_{j-2}, h_i) \right) \tag{1}$$

We apply modified Kneser-Ney smoothing [29] to all conditional probabilities of our model in order to compute probabilities for candidates that do not appear in the training data. This increases the freedom of user input. For details on the smoothing implementation, we refer to our previous paper [21].

4.4 Harmonic Rhythm

We want to model harmonic rhythm in order to reduce the probability of exceedingly long harmonies or unrhythmic changes. However, this restricts us to training the model on data with suitable harmony annotation, which is only available in relatively small amounts. Therefore, we approximate trigrams of harmonies $h_i \equiv (c_i, r_i)$ by assuming the approximate independence of harmonic content c_i (root degree and chord quality) and harmonic rhythm r_i (onset beat and duration).

$$\begin{aligned} P(h_i \,|\, h_{i-1}, h_{i-2}) &= P(c_i, r_i \,|\, c_{i-1}, r_{i-1}, c_{i-2}, r_{i-2}) \\ &\approx P(c_i \,|\, c_{i-1}, c_{i-2})\ P(r_i \,|\, r_{i-1}, r_{i-2}) \end{aligned} \tag{2}$$

An exemplary value for r_i is [onset = first beat of a bar; duration = 2 beats]. In a trigram, the information of two of the three onsets is redundant given the duration of all harmonies, since after the exemplary r_i, the onset of r_{i+1} has to be the third beat of a bar. However, not using onset information would ignore, that, for example, a trigram with the durations [2 bars, 2 bars, 4 bars] should be more likely to start on a first beat of a bar than on a second beat, because the latter would result in an unusual syncopatic harmonic rhythm where harmonies change on weak beats and cross bar lines. If we do not use rhythmic information at all, the continuation of the same harmony becomes the most likely, i.e., if a user, for example, inserts no input in three consecutive bars, the system would most likely output one single harmony spanning all three bars, which we want to avoid. Not using Eq. (2) to approximate independence between harmonic content and rhythm results in frequent fallback to bigrams or even unigrams due to data sparsity, which can cause less smooth harmony progressions. In addition to reducing data sparsity, Eq. (2) allows us to train harmonic content and rhythm on different data sets. We can use rhythm information from chorales in combination with harmony information of a larger number of classical pieces. The harmony information of the latter is tonality-independent, and thus more relevant for harmony progressions [30].

4.5 Voicing Structure

In a four-part chorale, each voicing v_j comprises the four notes n_j^S, n_j^A, n_j^T and n_j^B of soprano, alto, tenor and bass, respectively.

We process notes as MIDI note numbers, i.e., the middle C equals the number 60, and the C# or D♭ above equals 61. We denote intervals between notes as follows.

$$I^S_{j-1\to j} \equiv n^S_j - n^S_{j-1} \tag{3}$$

where $I^S_{j-1\to j}$ is the melodic interval from beat $j-1$ to beat j.

To reduce the problem of data sparsity, we apply multiple heuristic approximations, considering how a human might approach music composition. First, we define a voicing transition probability $P_T(v_j \mid v_{j-1})$, and discuss how to account for the additional context v_{j-2} and h_i in Section 4.6. Despite the reduced context, the combinatorial complexity of note combinations is of $O(r^8)$, of which only very small subset appears in the training data. Therefore, we further approximate the probability of voicing transitions by considering inter-voice distance and relative melody motion separately as follows.

$$P_T(v_j \mid v_{j-1}) \approx \frac{P(v_j)\, P(I^v_{j-1\to j})}{\sum_{v_{j-1}} P(v_j)\, P(I^v_{j-1\to j})} \tag{4}$$

where $P(I^v_{j-1\to j}) \equiv P(I^S_{j-1\to j}, I^A_{j-1\to j}, I^T_{j-1\to j}, I^B_{j-1\to j})$ denotes the probability of the melody intervals of the four voices between voicings v_{j-1} and v_j. $P(v_j) \equiv P(n^S_j, n^A_j, n^T_j, n^B_j)$ captures inter-voice distance, which is important because the sound of a chord significantly varies depending on the relative position of its notes, and music theory encourages keeping the distance between these notes within certain bounds. On the other hand, $P(I^v_{j-1\to j})$ captures the relative motion of voices and thus the relation between similar (same direction) and contrary motion, where the latter is encouraged by music theory. Our approximation allows learning how to balance these aspects by learning from training data, while reducing the combinatorial complexity from $O(r^8)$ to a maximum of $O(r^4)$. The normalization factor in the denominator is required, because v_j affects both $P(v_j)$ and $P(I^v_{j-1\to j})$. Its computation takes a lot of time, but has to be done only once.

In addition, $P(v_j)$ and $P(I^v_{j-1\to j})$ are factorized into conditional probabilities as follows in order to benefit from smoothing techniques ($P(I^v_{j-1\to j})$ in analogy to $P(v_j)$).

$$P(v_j) = P(n^S_j)\, P(n^A_j|n^S_j)\, P(n^T_j|n^A_j, n^S_j)\, P(n^B_j|n^T_j, n^A_j, n^S_j) \tag{5}$$

$$P(I^v_{j-1\to j}) = P(I^S_{j-1\to j}) \ldots P(I^B_{j-1\to j}|I^T_{j-1\to j}, I^A_{j-1\to j}, I^S_{j-1\to j}) \tag{6}$$

The smoothing is applied to account for unseen candidates. For example, a tenor note n^T_j might not appear in the training data in the context two specific notes n^S_j and n^A_j. However, thanks to applied smoothing, the context of only the neighboring n^A_j is considered as well. Since this is more likely to appear in training data, we can obtain more meaningful probabilities for some user inputs, where the non-factorized probability $P(v_j)$ would be 0. When we furthermore only consider relative voice distance instead of absolute pitch in $P(v_j)$, the system becomes more flexible towards user input with uncommonly high or low notes.

Lastly, in order to identify parallel fifths and octaves, information about both distance (perfect fifth or octave) and motion (parallelity) is required, which is not captured by the approximated probability. Therefore, we multiply $P_T(v_j \mid v_{j-1})$ with a heuristic penalty factor $H_p(v_i, v_{i-1})$.

$$H_p(v_i, v_{i-1}) = \alpha_{\mathrm{p}}^{N_{\mathrm{p}}(v_i, v_{i-1})}\ \alpha_{\mathrm{hp}}^{N_{\mathrm{hp}}(v_i, v_{i-1})} \tag{7}$$

where $N_{\mathrm{p}}(v_j, v_{j-1})$ and $N_{\mathrm{hp}}(v_j, v_{j-1})$ are the numbers of parallel and hidden parallel fifths and octaves between the voicings v_j and v_{j-1}, respectively. Hidden parallel motion describes the state where two voices move towards a perfect fifth or octave in the same direction but are not perfectly parallel. This motion should also be avoided according to music theory, but the rule is not as strict as for parallel motion, thus $\alpha_{\mathrm{p}} < \alpha_{\mathrm{hp}} < 1$.

4.6 Harmonic and Melodic Context

The amount of data containing both four-part voicing and harmony annotation is very small compared to the complexity of the problem. Therefore, we implemented a harmonic dependency heuristic $H_h(v_j, h_i)$ that reduces the context information to the number of occurrence of certain pitch classes in a voicing. In case of a triad harmony, the heuristic is defined as follows.

$$H_h(v_j, h_i) = P(N_{\mathrm{root}}, N_{\mathrm{third}}, N_{\mathrm{fifth}}, N_{\mathrm{nonharmonic}}) \tag{8}$$

where N_{root} is the number of root notes of the harmony h_i in the voicing v_j. N_{third}, N_{fifth}, and $N_{\mathrm{nonharmonic}}$ are the numbers of thirds, fifths and nonharmonic tones, respectively. Separate probabilities are computed for seventh chords. The heuristic does not capture the order or distance of these notes, and thus there is little information overlap with the voicing structure probability $P_T(v_j \mid v_{j-1})$. The combinatorial complexity is low enough to make training on a small data set feasible. It allows the algorithm to not only naturally penalize nonharmonic tones, but also capture note doubling. To handle unseen user input, additive smoothing is applied.

However, simply penalizing nonharmonic tones does not always result in a favorable outcome [22]. According to music theory, there are several classes of nonharmonic tones, such as passing tones and suspensions, which are defined by the intervals surrounding the nonharmonic tone [23]. Our algorithm tracks whether nonharmonic tones are properly resolved and if not, penalizes the corresponding voicing's probability. In our experiments, we considered the major nonharmonic tone classes found in Bach's chorales: Passing tones, neighboring tones and suspensions. Additionally, nonharmonic tones that do not belong to the current musical key's scale are penalized.

Lastly, $P_T(v_j \mid v_{j-1})$ considers only one previous voicing and is therefore unable to capture music-theoretical rules concerning consecutive melody intervals. For example, consecutive large jumps in a melody should be avoided, and instead jumps should be followed by small movements in the other direction [19]. Adding an additional voicing v_{j-2} to the context of P_T would be infeasible considering the combinatorial complexity. However, from a composers standpoint, one could view longer melodic dependencies separately from the immediate voicing transition. Furthermore, most rules for melody can be formulated by grouping intervals $I_{j-1\to j}$ into movement types $M_{j-1\to j}$: Minor seconds (one semitone) and major seconds (two semitones) are both so-called step intervals, and any larger interval is a skip. Many melodic rules concern the relation between consecutive steps and skips. We implement the melodic probability as follows, where C denotes all the context information already considered.

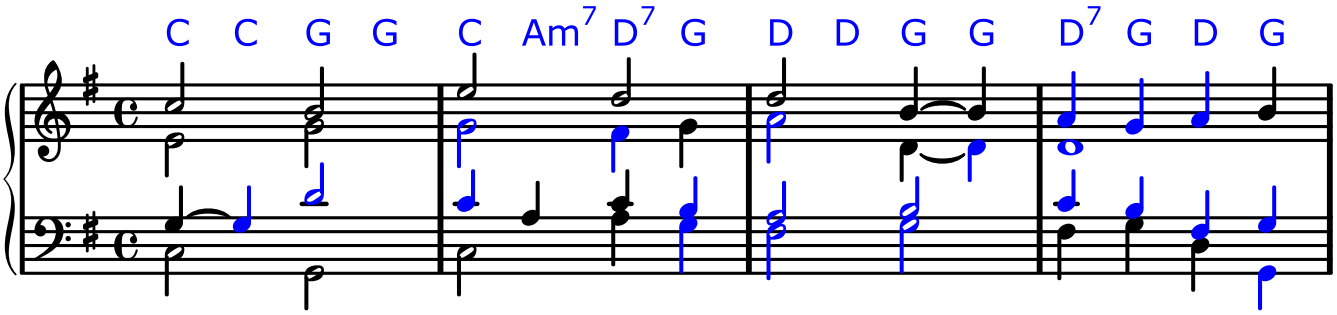

Fig. 4 A result of the automatic music completion system. Notes and harmonies input by the user are shown in black, and the output of the system in blue. While not illustrated, it is possible to specify multiple harmony candidates for each beat, of which the algorithm will chose the optimal one.

$$P(M_{j-1\to j}\,|\,C, M_{j-2\to j-1}) \approx P(M_{j-1\to j}\,|\,C)\; P(M_{j-1\to j}\,|\,M_{j-2\to j-1})\; P(M_{j-1\to j})^{-1} \tag{9}$$

which results from Bayes' rule if assuming approximate independence between immediate context C (voicing transition) and melodic context $M_{j-2\to j-1}$. Since $P(M_{j-1\to j}\,|\,C)$ is already contained in $P_T(v_j\,|\,v_{j-1})$, and the number of movement types is low compared to the number of intervals, the added context does not increase computational complexity very much. When considering the melodic context separately for each voice, the extension of the context to several melody intervals is quite computationally cheap. Thus, we combine detailed context in the immediate vicinity of a voicing with efficiently reduced but very relevant context information for longer distance dependencies.

In summary, the probability of a voicing in its context is obtained as follows.

$$P(v_j\,|\,v_{j-1}, v_{j-2}, h_i) \propto P_T(v_j|v_{j-1})\, H_p(v_i, v_{i-1})\, H_h(v_j, h_i)\, \frac{P(M_{j-1\to j}\,|\,M_{j-2\to j-1})}{P(M_{j-1\to j})} \tag{10}$$

Using this probability in combination with $P(h_i\,|\,h_{i-1}, h_{i-2})$, our algorithm explores harmonies and voicings for filling in gaps left open by the user and searches for the most probable solution.

5. Evaluation

5.1 Music-Theoretical Evaluation Experiment

Our algorithm was designed to find solutions that best adhere to various principles of music theory. Therefore, we statistically analyzed how often music-theoretical rules are violated by the algorithm. However, following music theory is easy without input constraints. To generate a variety of randomized inputs, we randomly chose 100 excerpts of 4 bars length from a test set containing 10% of the 380 four-part chorales obtained from the Classical Archives [25]. We randomly removed 50% percent of the notes and shifted some pitches. The musical key was estimated by minimizing the number of notes not contained in the corresponding scale. The joint optimization of harmony and voicing allowed the algorithm to find a solution for every input, which is an improvement over our previous results of a 90% success rate [21]. An exemplary output is shown in **Fig. 4**.

A selection of results of our analysis is displayed in **Fig. 5**. Parallel fifths and octaves were almost completely suppressed by

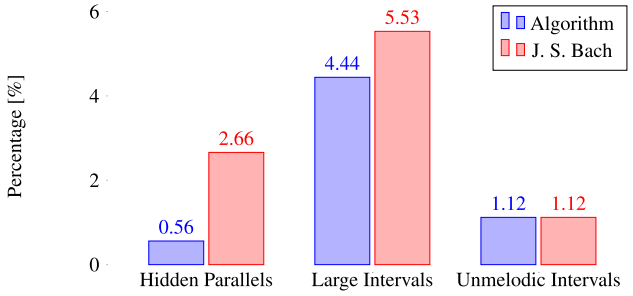

Fig. 5 Exemplary results of music-theoretical analysis. Since rules of music theory are not absolute, statistics from Bach's chorales are shown for comparison. Occurrence of hidden parallels is compared, because true parallel fifths and octaves were almost completely suppressed. Intervals between soprano, alto and tenor exceeding octaves are classified as large. Melody intervals that should be avoided according to music theory (such as tritones) are classified as unmelodic.

our algorithm and hidden parallels were fewer than in Bach's chorales. Likewise, intervals between voices exceeding a tenth were almost completely suppressed and intervals exceeding octaves were avoided to a similar extent as by Bach. Melody intervals that should be avoided according to music theory were equally few in generated music and Bach's chorales. However, the amount of step intervals (one or two semitones) in melodies is significantly higher in Bach's chorales (76%) than in the automatically generated results (61%), resulting in less smooth melodies. The balance between similar and contrary voice motion (18.5% and 19.1%) differs slightly but not greatly from Bach (16.4% and 15.7%). These results indicate that the algorithm succeeds at computing valid voicings for various inputs, but for evaluating the actual sound we conducted the next experiment.

5.2 Subjective Evaluation Experiment

Evaluating music is inherently difficult due to the involvement of a listener's personal taste. In our case, copying any specific composer is not the goal of the algorithm, so there are no original pieces that would allow users to compare their generated music for making an evaluation. Furthermore, an important factor of our system is the response to individual user input, which is why preparing music samples prior to an experiment does not make much sense, and makes comparative evaluation even more difficult. Therefore, we instead implemented a graphical user interface, that can be accessed online. We conducted an experiment, in which 48 participants were asked to use the system with their individual input and evaluate the generated results.

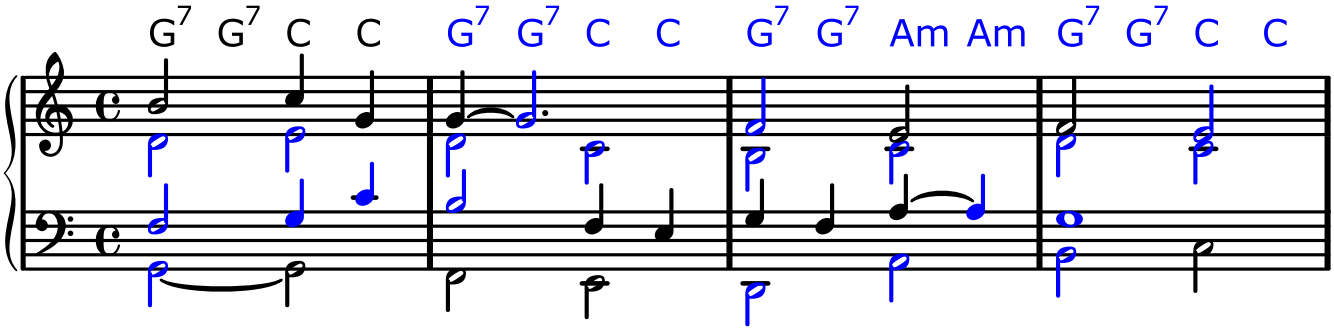

Fig. 6 An example of a problematic solution (input notes and harmonies in black, generated ones in blue). This example contains hidden parallel motion (beat 7 and 11) and a problematic nonharmonic tone: The F in the tenor at beat 7 is treated as a nonharmonic tone of a C major chord despite being accented (i.e. the harmony changes) and on a heavy beat.

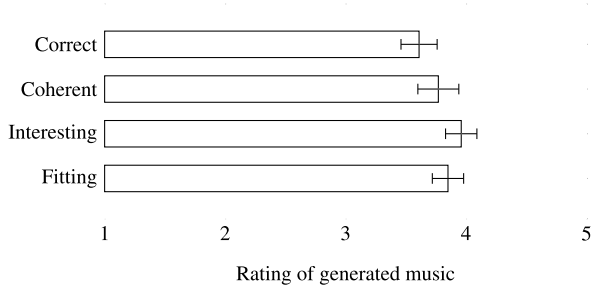

Fig. 7 The results of our subjective evaluation experiment, averaged over 48 participants. Generated music was rated with respect to the shown criteria, where 1 is the worst and 5 the best rating.

In the experiment, the system was rated according to the four criteria shown in **Fig. 7**. The participants were first asked to judge whether the generated music adheres to classical music theory, especially with regard to rules for polyphonic voicing (such as shown in Fig. 2). However, we also received some feedback regarding the harmony. To interpret the participants' answers to this question, they were also asked to rate their music-theoretical knowledge on a scale from 1 to 5. Answers from participants without knowledge in music theory (rating 1) were ignored, and the other answers linearly weighted. In other words, the answers of a participants with extensive knowledge (rating 5) were weighted 4 times higher than that of a participants with little knowledge (rating 2). The received feedback was mostly positive: Voicing rule violations were rare (occasionally, hidden parallel or consecutive seventh movement was observed) and also the chord progressions were generally perceived as smooth, albeit quite conservative. Some users pointed out a few problematic nonharmonic tones which were ambiguous and lead to incorrect interpretation of a harmony, i.e., the voicing sounded different from the annotated chord. An example for such a nonharmonic tone is the tenor note on the 7th beat in **Fig. 6**. To account for participants without music-theoretical knowledge, we also asked for an evaluation of coherence, namely the subjective musicality or validity of the generated music. The results (not weighted according to knowledge) were slightly better than for correctness, but similar problems with the harmony were pointed out, mainly occasional occurrence of unexpected harmonies.

To evaluate how well our system fulfills its goal of supporting music creation, we also asked participants to rate how interesting the generated music was, and how well it fit their input. Interestingness was rated highest, but might be influenced by the fact that having their musical idea completed by a computer was a new experience for the participants. Most users reacted positively to this experience and reported that their musical idea was realized in the generated piece. Critique included harmony progressions being too conservative and the wish to choose a style for the music to be generated. However, the ratings varied significantly due to personal tastes involved. Therefore, we invite the reader to experiment with the system themselves at http://160.16.202.131/music_completion.

6. Conclusion

In this paper, we propose automatic music completion as a generalization of several tasks such as melody generation and harmonization, which were usually treated separately in previous research. We present a system that allows a user to freely insert any combination of melody fragments and harmony constraints. It computes a complete four-part chorale by using an optimization algorithm based on a probabilistic model of harmony and voicing. Statistical analysis of generated music confirmed that the algorithm follows multiple music-theoretical principles. We also received positive feedback in a subjective evaluation experiment.

Our current system could be improved by implementing a sophisticated model of harmonic and melodic rhythm. Furthermore, providing tuning parameters that allow users to intentionally influence the automatic generation process, e.g., by encouraging interesting harmonies or rhythms, could improve both the user interaction and the generated music.

Our main goal is to apply the principle of automatic music completion to a wider variety of music. Since the idea is not constrained to chorales, future research in this direction could realize systems that allow users to create their own jazz or pop songs, or even orchestral pieces. The focus of such systems is to realize the musical ideas of their users, making music composition more accessible to inexperienced users, and providing an entertaining collaboration between human and computer.

Acknowledgments This work was in part supported by JSPS KAKENHI Grant Number 17H00749.

References

[1] Pachet, F. and Roy, P.: Musical harmonization with constraints: A survey, *Constraints*, Vol.6, No.1, pp.7–19 (2001).

[2] Hild, H., Feulner, J. and Menzel, W.: HARMONET: A neural net for harmonizing chorales in the style of JS Bach, *Advances in Neural Information Processing Systems*, pp.267–274 (1992).
[3] Hiller, L.A. and Isaacson, L.M.: Experimental music: Composition with an electronic computer (1959).
[4] Fernández, J.D. and Vico, F.: AI methods in algorithmic composition: A comprehensive survey, *Journal of Artificial Intelligence Research*, Vol.48, pp.513–582 (2013).
[5] Cope, D.: Experiments in musical intelligence, *Proc. International Computer Music Conference, San Francisco* (1987).
[6] Quintana, C.S., Arcas, F.M., Molina, D.A., Rodríguez, J.D.F. and Vico, F.J.: Melomics: A case-study of AI in Spain, *AI Magazine*, Vol.34, No.3, pp.99–103 (2013).
[7] Quick, D.: *Kulitta: A framework for automated music composition*, Yale University (2014).
[8] Oore, S., Simon, I., Sander, D. and Eck, D.: Learning to Create Piano Performances, *NIPS Workshop on Machine Learning and Creativity* (2017).
[9] Fukayama, S., Nakatsuma, K., Sako, S., Nishimoto, T. and Sagayama, S.: Automatic song composition from the lyrics exploiting prosody of the Japanese language, *Proc. 7th Sound and Music Computing Conference* (*SMC*), pp.299–302 (2010).
[10] Roig, C., Tardón, L.J., Barbancho, I. and Barbancho, A.M.: Automatic melody composition based on a probabilistic model of music style and harmonic rules, *Knowledge-Based Systems*, Vol.71, pp.419–434 (2014).
[11] Roberts, A., Engel, J., Raffel, C., Hawthorne, C. and Eck, D.: A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music, *CoRR*, Vol.abs/1803.05428 (2018) (online), available from ⟨http://arxiv.org/abs/1803.05428⟩.
[12] Hirai, T. and Sawada, S.: Melody2Vec: Distributed Representations of Melodic Phrases based on Melody Segmentation, *Journal of Information Processing*, Vol.27, pp.278–286 (2019).
[13] Biles, J.A. et al.: GenJam: A genetic algorithm for generating jazz solos, *ICMC*, Vol.94, pp.131–137 (1994).
[14] Freitas, A. and Guimaraes, F.: Melody harmonization in evolutionary music using multiobjective genetic algorithms, *Proc. Sound and Music Computing Conference* (2011).
[15] Raczyński, S.A., Fukayama, S. and Vincent, E.: Melody harmonization with interpolated probabilistic models, *Journal of New Music Research*, Vol.42, No.3, pp.223–235 (2013).
[16] Papadopoulos, A., Roy, P. and Pachet, F.: Assisted lead sheet composition using flowcomposer, *International Conference on Principles and Practice of Constraint Programming*, pp.769–785, Springer (2016).
[17] Hadjeres, G., Pachet, F. and Nielsen, F.: DeepBach: A Steerable Model for Bach chorales generation, arXiv preprint arXiv:1612.01010 (2016).
[18] Huang, C.-Z.A., Cooijmans, T., Roberts, A., Courville, A. and Eck, D.: Counterpoint by convolution, *ISMIR* (2017).
[19] Marx, A.B.: *Theory and practice of musical composition*, Gordon (1866).
[20] Lee, K.: Automatic Chord Recognition from Audio Using Enhanced Pitch Class Profile, *ICMC* (2006).
[21] Wilk, C.M. and Sagayama, S.: Harmony and voicing interpolation for automatic music composition assistance, *APSIPA* (2018).
[22] Wilk, C.M., Ito, S. and Sagayama, S.: Music interpolation considering nonharmonic tones, *SIGMUS* (2018).
[23] musictheory.net, available from ⟨https://www.musictheory.net/lessons/53⟩ (accessed 2019-01-15).
[24] Radicioni, D.P. and Esposito, R.: BREVE: an HMPerceptron-based chord recognition system, *Advances in Music Information Retrieval*, pp.143–164, Springer (2010).
[25] Classical Archives LLC, available from ⟨https://www.classicalarchives.com⟩ (accessed 2018-05-23).
[26] Kaneko, H., Kawakami, D. and Sagayama, S.: Functional harmony annotation data-base for statistical music analysis, *ISMIR* (2010).
[27] Papadopoulos, H. and Peeters, G.: Large-Scale Study of Chord Estimation Algorithms Based on Chroma Representation and HMM, *International Workshop on Content-Based Multimedia Indexing, CBMI'07*, pp.53–60 (2007).
[28] Ueda, Y., Uchiyama, Y., Nishimoto, T., Ono, N. and Sagayama, S.: HMM-based approach for automatic chord detection using refined acoustic features, *2010 IEEE International Conference on Acoustics Speech and Signal Processing* (*ICASSP*), pp.5518–5521, IEEE (2010).
[29] Chen, S.F. and Goodman, J.: An empirical study of smoothing techniques for language modeling, *Proc. 34th Annual Meeting on Association for Computational Linguistics*, pp.310–318, Association for Computational Linguistics (1996).
[30] Scholz, R., Vincent, E. and Bimbot, F.: Robust modeling of musical chord sequences using probabilistic N-grams, *ICASSP*, Vol.53–56 (2009).

Christoph M. Wilk was born in 1992. He received his M.S. degree from Heidelberg University, Germany in 2017. He is currently a Ph.D. student at Meiji University. His research interest is mathematical modeling of musical concepts.

Shigeki Sagayama was born in 1948. He received his B.E., M.S. and Ph.D. degrees from the University of Tokyo, Tokyo, Japan, in 1972, 1974, and 1998, respectively, all in mathematical engineering and information physics. After spending 24 years with NTT Laboratories in Tokyo and Yokosuka, Japan, and ATR Interpreting Telephony Laboratories, Kyoto, Japan, he became a Professor of Japan Advanced Institute of Science and Technology (JAIST), Ishikawa, Japan. In 2000, he was appointed Professor at the University of Tokyo, Tokyo, Japan. Since 2014, he was a Professor of Meiji University for 5 years. His major research interests include processing, recognition and synthesis of speech, music, acoustic signals, handwriting, and images. Prof. Sagayama received the National Invention Award from the Institute of Invention of Japan in 1991, the Director General's Award from the Science and Technology Agency of Japan in 1996, and other academic awards. He is a fellow of IEICEJ, a life member of IEEE and a member of the ASJ (Acoustical Society of Japan) and IPSJ.