Style-Specific Harmonization by Multi-Objective Optimization without Training

Stephanie Y. H. Lew[1], Wee Kheng Leow[1], and Kat R. Agres[2]

[1]Department of Computer Science, National University of Singapore
[2]Centre of Music and Health, National University of Singapore
slewyh@comp.nus.edu.sg, leowwk@comp.nus.edu.sg, katagres@nus.edu.sg

Abstract. Style-specific harmonization aims at generating a chord sequence that matches a given melody in a specific style. Existing methods for generating chord sequence have some shortcomings. Most of them are trained on a large training set of a specific genre such as pop or jazz. To accommodate to different styles, they need to train a different version of the method on a different style-specific training set, increasing the total training time. If they are trained on a mixed-style set that contains examples of various styles, they will mix up the styles and produce results that are diluted in style. Moreover, existing methods cannot cater to cadence and voice leading. Many recent deep neural networks (DNNs) for harmonization are tested on 8-bar phrases instead of complete music pieces. While this makes testing easier, they cannot handle the global structure of music which lead to better overall harmonic coherence. This paper proposes a style-specific harmonization method called FlexChord that overcomes the shortcomings of existing methods. FlexChord achieves style specificity by producing chord sequence that matches the style of a selected set of example music, which can be as little as a single example music piece. It applies direct optimization during chord generation that does not require training. Thus, it can easily produce chord sequence of a specific style or a custom style. The requirement of cleaning, annotating and training on large datasets is avoided. In addition, FlexChord can handle cadence and voice leading effectively. It can generate chords that match full-length melodies, complete with global music structure. Test results show that FlexChord's harmonization is better than those of comparable DNNs and is close to that of human harmonization.

Keywords. Chord generation, harmonization, multi-objective optimization, style-specificity.

1 Introduction

Harmonization is the process of generating a chord progression to accompany a melody. There are two forms of harmonization [1]: chord generation and chorale generation. The first form generates a sequence of chords at strong beats to match the melody. The second form distributes chord notes into three to four voices to support the melody. This paper focuses on chord generation.

Music style is influenced by the relationships between melody, chords, and chord progressions. It can be understood at multiple levels. At the coarsest level, it refers to specific genres such as pop, jazz and classical. Style can also refer to music in specific historical periods such as Baroque, Classical, Romantic, etc. At a finer level, it reflects

the style of individual composers like Beethoven and Chopin. Even within a composer's work, different pieces can be stylistically different such as Chopin's Mazurkas and Nocturnes. At the finest level, each piece of music can have a different style. For example, Chopin's Mazurkas No. 1 and No. 2 are stylistically distinct even though they are both Mazurkas.

Past works on harmonization [2], [3], [4], [5], [6], [7], [8] use well-defined rules and grammar to encode genre-specific harmony. It is difficult to manually create rules and grammars for various styles. Evolutionary algorithms [9], [10], [11], [12], [13] offer flexibility by maintaining a population of chord sequences and selecting them based on fitness function. Unfortunately, defining style-specific fitness function is difficult, if not impossible, for some styles.

Recent methods train hidden Markov models (HMMs) [14], [15], [16] and deep neural networks (DNNs) [17], [18], [19], [20], [21], [22], [23], to generate chords sequences. These DNNs are trained on a large training set specific to a particular genre, such as pop or jazz, whereas the HMMs are trained on smaller datasets. An exception is the HMM of [16] that requires the chord sequence of only one reference song for generating the chord sequence of a different target song. To accommodate to different styles, both HMMs and DNNs need to train a different version of the method on a different style-specific training set, increasing the total training time. If they are trained on a mixed-style set that contains examples of various styles, they will mix up the styles and produce results that are diluted in style. Except for the HMM of [24] that handle cadence, the other HMMs and DNNs cited above do not cater to cadence and voice leading. Moreover, they are tested on 8-bar to 12-bar phrases instead of full-length melodies.

This paper proposes a harmonization method called **FlexChord** that overcomes the shortcomings of existing methods, and contributes to the following:

- FlexChord can generate harmonization of the required style at various levels such as pop, jazz, a particular Greek composer and high-tension harmonization based on Hitchcock's *Psycho*.
- As for [16], FlexChord achieves style-specificity using as few as a single reference music piece. Unlike [16], FlexChord is fully automated and does not require the user to provide the start and end chords of a melody section.
- Generate harmonizations of custom styles using different reference music.
- FlexChord belongs to the category of direct optimization method (Section 2) that **does not** require training, thus eliminating the need to clean and annotate large amounts of training data.
- Handle cadence and voice-leading that are absent in existing work.
- Generate harmonization for full-length melodies.

Audio files of test results are available in https://tinyurl.com/flexchord.

2 Related Work

Chord generation methods can be grouped into five categories, namely rule-based methods, evolutionary algorithms, neural networks, Markov models and direct optimization. The first two categories are reviewed in Section 1 and [1].

Neural networks [25], [26] and deep neural networks (DNNs) [17], [18], [19], [20], [21], [22], [23], are trained to generate chords that fit the melodies and chord sequences of training examples. Training of these methods require large training sets that can mix up the various styles of the training samples, diluting the resulting harmonization style. They cannot harmonize a melody with various music styles without re-training [27]. The DNNs in [20], [21] support only simple triads and a few seventh chords. Consequently, their results are stylistic limited.

Existing Markov models for harmonization consist of two main types, namely Markov chains and hidden Markov models. The Markov chains of [28], [29] generate chords for melody using state transition probabilities. [28] derives state transition probabilities from the frequency counts of chord transitions in a small example set during data preparation. [29] defines chord transition probabilities based on neo-Riemannian principles. Thus, these methods do not require training. However, it is difficult to adapt [29] to various styles.

The hidden Markov models (HMMs) of [14], [15], [16], [24], [30], [31], [32] model the relationship between hidden states and observable melody from the training data, and capture both state transition and output probabilities. The HMMs of [14], [15], [31], [32], [33] apply the Viterbi algorithm, a dynamic programming method, to determine the optimal chord sequence for the given melody. HMMs can be trained on a small training set [34]. An exception is the HMM of [16] which requires only the chord sequence of a single reference song for training.

The DNNs and HMMs cited above are tested only on 8-bar to 12-bar phrases. To handle full-length melodies, they need to train on examples of various lengths, increasing the complexity and training time of these methods significantly. Moreover, the HMM of [16] is not fully automated because they require the user to specify the start and end chords of a melody section.

Direct optimization methods [27], [35], [36] generate chord sequences by directly optimizing an objective function using optimization algorithms without the need for training. In principle, they can incorporate reference inputs or constraints in the optimization process to achieve style specificity. Unfortunately, these methods do not adopt any approach that is style-specific.

All the above methods optimize a weighted-sum of loss functions. This is a common technique for solving multi-objective optimization but it can miss good solutions (Section 4.1). It is very difficult for them to handle cadence, which is a hard constraint at phrase and section endings, unless the ending chords are specified by the user as for [15], [16]. Incorporating cadence into the weighted loss function turns cadence into a soft constraint. This technique encourages the generated chords to follow cadence but cannot enforce it. Moreover, these methods do not consider chord inversion, which can improve voice leading.

3 Style-specific Harmonization

FlexChord is guided by three objectives: (1) good chord-melody matching, (2) good chord progression and (3) good voice leading. It generates a sequence of chords that simultaneously optimizes the three objectives, given a melody.

3.1 Representation of Melody and Chords

Melody is represented as a step function with a sequence of m pitches over integer time index $i = 0, \ldots, m - 1$. Chords are generated at each **strong beat** to match the pitch sequence and structure of the melody. Let j denote strong beat index, for $j = 0, \ldots, n - 1$. It is related to time index i by

$$j = int(i/\alpha\beta), \alpha = \begin{cases} 2 & \text{if } S \in 2/4,4/4, \\ 3 & \text{if } S \in 3/4,6/8, \end{cases} \quad (1)$$

where β is the number of pitches per beat, which is $4, 8, 16$ or 32, and S is the key signature of the melody. This gives 1 strong beat per bar for $2/4$ and $3/4$ music and 2 strong beats per bar for $4/4$ and $6/8$ music. With strong beat index j, the melody is organized as a sequence of fragments h_j, $j = 0, \ldots, n-1$. Each fragment h_j contains pitches $p_{\alpha\beta j}, \ldots, p_{\alpha\beta(j+1)-1}$. It has a structure label λ_j that takes the value of 'P', 'S', or ' ', respectively for end of phrase, end of section, which is also end of phrase, or neither. A phrase is a 4-bar or 8-bar sequence that is no longer than a section. For a fragment h_j that is not the end of phrase and section, and repeats an earlier phrase $h_{j'}$ exactly, its $\lambda_j = j'$.

FlexChord produces chords c_i, each consisting of multiple pitches q_{ik}, for $k = 1, \ldots, \kappa$ where κ is the number of pitches or voices in the chord. Chord c_j at strong beat j is matched to fragment h_j of the melody, and it is held for the duration from $i = \alpha\beta j$ to $\alpha\beta(j + 1) - 1$. Thus, it is required that $m = \alpha\beta n$. Only chords c_j have onsets; the others are sustained.

3.2 Chord-Melody Matching

Music perception studies point to the match between a chord c_j and a melody fragment h_j as the amount of dissonance between their pitches [37], [38]. The dissonance value $d(p, q)$ between two pitches p and q depends on their pitch interval $\theta(p, q) = |p - q| \; mod \; 12$, and is given in Huron's study [37]. The higher the dissonance, the less harmonious are p and q when they are played together.

A similar concept can be defined for the dissonance $d(h_j, c_j)$ between fragment h_j and chord c_j. It is measured by the average dissonance between them:

$$d(h_j, c_j) = \frac{1}{|h_j||c_j|} \sum_{p_i \in h_j} \sum_{q_{jk} \in c_j} d_p(p_i, q_{jk}), \text{ for } p_i \neq \text{rest}, \quad (2)$$

where $|h_j|$ and $|c_j|$ are the total number of pitches in fragment h_j and chord c_j respectively.

3.3 Chord Progression and Harmonization Style

In music theory, chord progression is one of the key elements in conveying harmonization style [39]. It is noted that chord progression alone may not differentiate between styles like Baroque, Classical and Romantic. Nevertheless, it can differentiate between pop and jazz, and between high-tension and low-tension harmonization. It can also characterize the styles of specific music pieces.

Let c and c' denote consecutive chord pairs in an example music set, where c' immediately follows c. The chord transition probability $P(c'|c)$ is easily computed by measuring the frequency of occurrence of chord pair (c, c') in the set. These transition probabilities form the chord transition model C that captures the required style. By using different chord transition models that captures different styles, our algorithm FlexChord can produce different results that follows the style of the chord transition model. As for Markov chains [28], [29], preparation of chord transition model is done in data preparation, which does not involve FlexChord algorithm. On the other hand, HMMs' computation of hidden states involves the HMMs themselves, which is analogous to DNN training.

As for [16], FlexChord's chord transition model can be prepared with as few as a single example music piece (Section 5.5). A caveat of using single example piece is that some chord transitions may be absent in the example. A common practice in HMM is to replace zero transition probability by a small arbitrary value. [16] comments that this approach can introduce inappropriate transitions, and resorts to using chord transition blending to enrich possible chord transitions. Although chord transition blending has been demonstrated for several styles such as Bach Chorales, Jazz, neo-tonal and post-tonal harmony [40], it is unclear whether it is feasible for other styles. This approach also creates additional complexity in the harmonization method. The issue of zero transition probability is significant for existing HMMs because they use short 8-bar to 12-bar examples. In contrast, our approach uses full-length example music piece to generate chord transition model, significantly mitigating the issue of zero transition probability. If needed, more example music pieces can be used. In essence, we choose the simpler data-driven approach over more complex approach of [16].

The chords in chord transition model C are represented by Roman numerals and captured at the root position. This representation is independent of music key and actual chord positions. Thus, the chord transition model C can be applied to the harmonization of melodies of any key. In contrast, most DNNs represent each position of the same chord (e.g., C-E-G, E-G-C) as a different chord.

3.4 Cadence

In music theory, cadence refers to melody or harmony at the end of a phrase or section that creates a sense of full or partial resolution [39]. Different cadences create different senses of resolution. They help to clarify the overall hierarchical structure of music [41].

In practice, it remains difficult to define a fixed set of cadences across multiple music styles [42]. A simplified approach is to refer to cadence as specific chord-pairs at the phrase and section endings. Each chord pair (c, c') in chord transition model C is given a structure label that takes the value 'P', 'S' , ' ', as defined in Section 3.1. Then,

cadence can be handled by matching melody fragments labeled 'P' and 'S' to chord pairs with the same labels (see Section 4.2 for details).

3.5 Voice Leading

Voice leading refers to the interaction of individual voices that creates harmony [43]. It is an important aspect of music where multiple voices play supporting roles in held chord generation. According to the pitch proximity principle, a chord progression that minimizes voice fluctuation has a good or efficient voice leading, which improves the coherence of the music [44, 43]. The voice fluctuation between two successive chords can be defined as the average distance moved by the implied voice parts between them. For two chords $c_i = \{q_{ik}\}$ and $c_{i-1} = \{q_{i-1,k}\}$, the voice fluctuation δ_i between them is defined as

$$\delta_i = \begin{cases} \sum_{k=1}^{\kappa} |q_{ik} - q_{i-1,k}| & \text{for } q_{ik} \neq \text{rest and } q_{i-1,k} \neq \text{rest} \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

3.6 Constraint and Optimization Objectives

The conditions of harmonization discussed in the previous sections are translated into a hard constraint for cadence and three optimization objectives:

1. Good chord progression and style matching:
   Style is captured in the chord transition model C. By maximizing chord transition probability $P(c_j|c_{j-1})$, for $j = 1, \ldots, n-1$, the generated chords c_j follows the style of C. Note that chord c_0 has no previous chord. å
2. Good chord-melody matching:
   By minimizing overall chord-melody dissonance D
   $$D = \sum_{j=0}^{n-1} d(h_j, c_j) \quad (4)$$
   the generated chords will match the melody well in terms of harmonization.
3. Smooth voice leading:
   Minimizing voice fluctuation Δ produces smooth chord voices:
   $$\Delta = \sum_{i=1}^{m-1} \delta_i \,. \quad (5)$$
   For held chords, voice fluctuation δ_j can also be defined between the chords at successive strong beat indices $j - 1$ and j because the chords between them remain unchanged.

4 Multi-objective Optimization

4.1 Overview

In general, a problem with multiple objectives cannot be perfectly solved by optimizing all the objectives simultaneously. A principled way of handling more than one objective

is to apply multi-objective optimization [45], [46]. This family of methods generalizes the optimality condition of single-objective optimization to the condition of **Pareto optimality**. A multi-objective optimization problem can have many Pareto optimal solutions. The set of all Pareto optimal solutions defines the **Pareto front**. A good method should be able to produce all possible Pareto optimal solutions, for example, by varying internal parameters randomly.

One commonly used approach for multi-objective optimization, called linear scalarization, combines multiple objective functions into a single function by a weighted sum. Then, the single weighted function is optimized using standard methods. This approach is used by many solution methods including Markov models and DNNs. Despite its simplicity, it has an inherent shortcoming — it cannot produce some Pareto optimal solutions when the Pareto front is non-convex [45], [46]. In other words, these methods can miss many good solutions regardless of how well they are designed and trained.

An alternative to linear scalarization is the lexicographic method. This method takes a set of objective functions sorted in decreasing order of importance by the decision maker. It first minimizes the most important objective function. If this step produces a set of optimal solutions, then it looks for solutions in the set that minimizes the next most important objective function. This procedure is continued until a unique solution remains or after all objective functions are optimized. A shortcoming of the lexicographic method is that less important objective functions are minimized only when there are multiple optimal solutions or ties. If there is no tie, less important objective functions are ignored.

The lexicographic method can be modified to overcome its shortcomings. In each step, instead of looking for solutions that minimize each objective function, the modified method looks for solutions on the Pareto front that are close to and including the minimum [46]. The bounds that specify closeness are manually set. When the bounds are small, the Pareto solutions obtained are very near to the minima of the first few objective functions. On the other hand, when the bounds are large, the Pareto solutions are far from the first few objective functions and nearer to the last few objective functions. Thus, by varying the bounds, the modified lexicographic method can produce all solutions on the Pareto front.

4.2 Harmonization Algorithm

FlexChord (Algorithm 1) adopts discrete modified lexicographic method (Lines 3–4, 14–15). Its overall structure is to assign a chord to the first beat of each fragment and hold the chord for the duration of the fragment length of $\alpha\beta$ by repeating the same chord without onset (Line 17). The first chord of the melody, with $j = i = 0$, is set to a chord c from the chord set C having a high transition probability to any chord with low dissonance with the first fragment h_0 (Lines 2–5).

For a fragment h_j at $j > 0$, several cases are considered. If h_j is just before a phrase or section ending labelled ‘P’ or ‘S’, a chord-pair (c, c') that matches structure label λ_{j+1} and has the lowest combined dissonance is chosen. Chord c is assigned to c_j, and c' is assigned to c_{j+1} (Lines 6–10) to satisfy the cadence constraint. Chord c_j is set at a position with the least voice fluctuation and c_{j+1} is set at the root position as cadence.

If λ_j is any integer j', it means that fragment h_j repeats $h_{j'}$ exactly. So, chord c'_j is copied to chord c_j (Lines 11–12). This method generates the same chord sequence for repeated phrases.

Otherwise, K chords having the largest transition probabilities are selected (Line 14). This step maximizes chord transition probability (Objective 1). Among these K chords, the chord c with the least dissonance is selected (Line 15). This step minimizes chord-melody dissonance (Objective 2). When K = 1, the algorithm ignores Objective 2. When $K = |C|$ (all the chords in C), it ignores Objective 1. A good balance between Objectives 1 and 2 is found with $1 < K < |C|$. This balancing of Objectives 1 and 2 are also applied to the first chord at $j = 0$ (Lines 3–4). Line 16 sets chord c_j at a position of c that minimizes voice fluctuation (Objective 3). Objective 3 is omitted at end of phrase and section to cater to cadence. In summary, FlexChord performs modified lexicographic optimization of 3 objective functions and 1 constraint.

FlexChord performs local optimization at each strong beat j sequentially. Although it does not perform global optimization as for the HMMs of [15], [16], it can handle full-length music via structure labels. In contrast, [15], [16] are tested only on 12-bar phrases. This approach neatly balances the handling of global structure versus algorithm complexity and efficiency. Experimental results show that this approach is feasible for producing good harmonization results.

5 Experiments and Discussions

5.1 Data Preparation

Two public datasets, namely Chord Melody Dataset (CMD) [47] and Hook Lead Sheet Dataset (HLSD) [48] were adapted for testing. The original datasets contain a mixture of pop, jazz, classical and new age music. For the purpose of style-specific tests, three sets of songs of distinct genres, namely pop and jazz, were extracted from the combined pool of CMD and HLSD (Table 1). The Pop-L and Jazz-L sets have more than 17,000 phrases, which are typical of the training sets used in [17]–[23].

To cater to the DNNs that were compared [19], [22], only songs in 4/4 time were selected. The songs were transposed to all 12 music keys for DNN training. Each song was encoded as a step-function with number of pitches per beat $\beta = 16$. The songs were split into 8-bar phrases, and a chord was sampled every half-bar.

Each dataset was split into training, validation and testing set at the ratio of 8: 1: 1. Subsets for neural network training and validation consisted of 8-bar phrases with an overlap of 2 bars. On the other hand, subsets for testing consisted of 8-bar phrases aligned to phrase boundaries without overlap. For testing, each 8-bar phrase in the test set was assigned a structure label where $\lambda_j =$ 'P' for $j = 7, 15$ and $\lambda_j =$ ' ' otherwise.

As a direct optimization method, FlexChord does not require training. The training set was used to build the chord transition model during data preparation and the testing set was used for performance testing. The validation set was not used. Chord transition probabilities were computed by measuring the frequency of occurrence of consecutive chord-pairs in the training set (Section 3.3). Structure labels were assigned to the chord-pairs as described above.

Algorithm 1: FlexChord Harmonization

Input: Melody M, style-specific chord transition model C.
Output: Chords c_i, $i = 0, \ldots, m-1$.

1 **for** $j = 0, 1, \ldots, n-1$ **do**
2 **if** $j = 0$ **then**
3 Select K chords with the largest transition probabilities to any chord.
4 Among K chords, select chord c with the least dissonance $d(h_0, c)$.
5 Set c_0 at the root position of c.
6 **else if** λ_{j+1} = *'P' or 'S'* **then**
7 Select a matching chord-pair (c, c') with the lowest combined dissonance.
8 Set c_j at the position of c with the least voice fluctuation δ_j.
9 **else if** λ_j = *'P' or 'S'* **then**
10 Set c_j at the root position of c'.
11 **else if** λ_j *is an integer* j' **then**
12 Set $c_j = c_{j'}$.
13 **else**
14 Select K chords with the largest transition probabilities $P(c \mid c_{j-1})$.
15 Among K chords, select chord c with the least dissonance $d(h_j, c)$.
16 Set c_j at the position of c with the least voice fluctuation δ_j.
17 Set $c_i = c_{i-1}, i = \alpha\beta j + 1, \ldots, \alpha\beta(j+1) - 1$.
18 Set chord onset for chord c_j.

5.2 Quantitative Evaluation

To evaluate the performance of FlexChord (FC), a handicapped version of FlexChord called FC− was also tested. In essence, FC− is FC without Lines 6– 12. That is, FC− does not handle cadence and is unaware of structure labels. In addition, FlexChord was compared with two well-known DNNs, namely VTHarm [19] and DAT-VAE [22]. Both DNNs have different architectures but use similar loss functions for harmonization.

Both VTHarm and DAT-VAE were trained on the training set (Section 5.1) with published hyperparameters [19], [22], validated on the validation set and tested on the testing set. FC and FC− used the chord transition model derived from the training set. They were tested on the testing set without training. The number of candidate chords K was set to 5 for both variants. This procedure was repeated for each of the three datasets of Pop-S, Pop-L and Jazz-L.

TABLE 1. SUMMARY STATISTICS OF THE DATASETS USED FOR THE COMPARATIVE EVALUATION.

Dataset	no. of songs	no. of phrases	no. of chord types
Pop-S	30	2,867	43
Pop-L	1000	81,625	70
Jazz-L	266	17,969	71

A. Quantitative Metrics

A set of simple but musically meaningful metrics were identified:

- Voice Fluctuation V (Eq. 3)
  A low V value means that voices in successive chords have lower pitch difference. This enhances the smoothness of chord progression.
- Fraction of inverted chords IC
  IC is the fraction of inverted chords in the chord sequence. A high IC value means that a large variety of chord positions beside root position exist in the chord sequence, and they encourage lower voice fluctuation.
- Chord histogram entropy H

  The entropy H of chord histogram is defined as:

  $$H = \sum_{k}^{N} p_k log p_k \,, \tag{6}$$

  where p_k denotes the probability of the k-th bin of the histogram and N is the total number of chord types present in the sequence. It ignores inversions. A higher H indicates a wider variety of chords used.
- Dissonance D

  The average dissonance D between melody notes and chords is defined as:

  $$D = \frac{1}{n} \sum_{j} d\left(h_j, c_j\right), \tag{7}$$

  where $d\left(h_j, c_j\right)$ denotes the average dissonance between pitches of melodic fragment h_j and chord c_j (Eq. 2). The lower the D the more harmonious is the matching between the chord and the melody.

- Pure chord tone to non-chord tone ratio R0

  R0 is the ratio of chord tones n_c to non-chord tones n_n in the melody fragment h_j in relation to its assigned chord c_j:

  $$R0 = n_c/(n_c + n_n). \tag{8}$$

  The higher the R0, the better is the match between the chord and melody.

TABLE 2. QUANTITATIVE EVALUATION FOR CHORD DIVERSITY AND CHORD-MELODY MATCH.

		Chord diversity			Chord-melody matching		
Dataset	Method	V↓	IC↑	H↑	D↑	R0↑	R1↑
Pop-S	VTHarm	NA	0.00	1.15	6.81	0.80	**1.23**
	DAT-VAE	NA	NA	1.46	**4.25**	0.63	0.63
	FC–	**0.66**	**0.61**	1.57	5.02	0.86	1.12
	FC	0.90	0.48	**1.74**	5.01	**0.88**	1.15
Pop-L	VTHarm	NA	0.00	**1.72**	6.44	0.83	1.27
	DAT-VAE	NA	NA	1.44	**4.36**	0.57	0.57
	FC–	**0.69**	**0.52**	1.44	5.10	**0.89**	**1.32**
	FC	0.97	0.44	1.68	5.21	0.88	1.31
Jazz-L	VTHarm	NA	0.00	**2.09**	7.27	0.81	1.23
	DAT-VAE	NA	NA	2.00	**4.55**	0.55	0.55
	FC–	**0.63**	**0.63**	1.57	5.75	0.88	**1.42**
	FC	0.96	0.53	1.91	5.75	**0.88**	1.41

- Chord tone to non-chord tone ratio R1

  R1 is similar to R0, except that proper non-chord tones are counted as chord tones [21]. A proper non-chord tone is a non-chord tone p_i that is within two semitones of its next tone p_{i+1}.

$$R1 = \big(n_c + n_p\big)/(n_c + n_n), \tag{9}$$

  where n_p is the number of proper non-chord tones.

Among these metrics, V, IC and H measure chord diversity, while D, R0 and R1measure chord-melody match. In particular, H and R1 are part of the canonical metrics [21] used in existing DNN-based harmonization works [19]. The remaining metrics in [21] are omitted because they are related to H and R1. Metric R1 uses the concept of proper non-chord tone that is not defined in music theory. Therefore, the alternative R0 is included along with R1. The metrics V, IC and D are added to contrast the performance of FlexChord and DNNs.

B. Quantitative Results

Table 2 tabulates the quantitative results. FC− inverts chord positions to minimize voice fluctuation (Lines 8, 16). Thus, it achieves the lowest voice fluctuation V and highest number of inverted chords IC. FC satisfies the cadence constraint that enforces chords at root position (Lines 9–10). Hence, it has a slightly higher V and slightly lower IC compared to FC−. DAT-VAE has no V and IC values because its results do not include information about chord position. VTHarm has no V value and has IC of 0 because its results are in root position.

FC− chooses frequently-used chords that maximize chord transition probability (Lines 3, 14). This restricts its chord choice, leading to lowest chord histogram entropy H. FC caters to cadence that enriches its chord diversity, thus achieving larger H than does FC−. DAT-VAE and VTHarm minimize chord reproduction error and KL divergence rather than explicitly maximizing chord transition probability. Their H scores appear to depend on the training and testing sets.

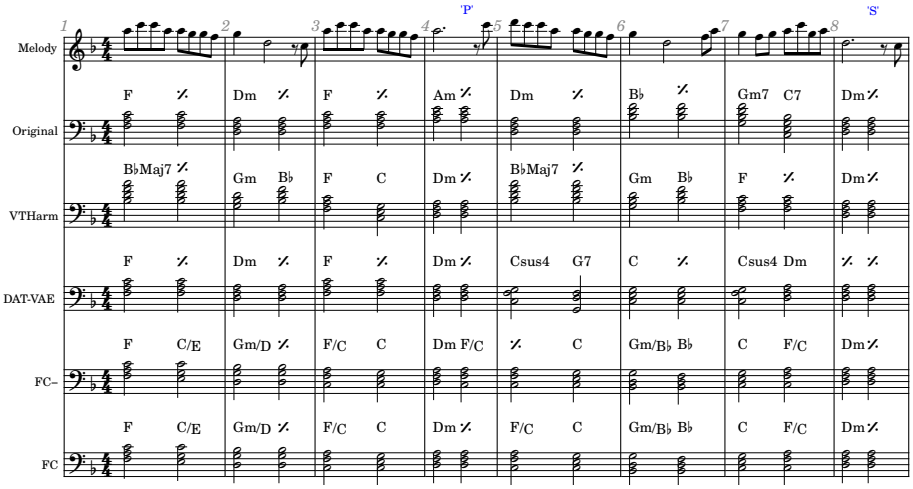

Fig. 1. Chords generated by various methods for the first 8 bars of *Colors of the Wind*.

FC and FC− have very similar dissonance D. They balance between the maximization of chord transition probabilities (Lines 3, 14) and the minimization of dissonance (Lines 4, 15). DAT-VAE has the lowest D because it minimizes chord reproduction error, allowing it to choose chords with the lowest dissonance. In contrast, VTHarm has a self-attention mechanism that allows it to choose less frequently-used chords, leading to largest dissonance D.

R0 is negatively correlated to D as more chord tones leads to lower dissonance. FC and FC− follow this negative correlation, with similar large R0 and small D. In contrast, the R0 scores of VTHarm and DAT-VAE seem to be positively correlated to their D scores, which is counter-intuitive.

R1 is expected to be larger than (or equal to) R0 because it counts proper non-chord tones as chord tones. All methods exhibit this property. Thus, R0 could replace R1 as a better performance metric because it does not require proper non-chord tone, which is a concept that is not defined in music theory.

Overall, FC achieves the best performance compared to other methods. It balances between good chord-melody matching (small D, large R0) and good chord progression that follows the required style. It caters to cadence and smooth voice leading (small V, large IC). Although DNNs may excel in an individual metric, they do not handle cadence and voice leading. Moreover, their D and R0 scores have a relationship that is opposite to the expected negative correlation.

5.3 Qualitative Evaluation

Figure 1 illustrates the chords generated by various methods for the first 8 bars of *Colors of the Wind*, a song in the testing set. Note that VTHarm and DAT-VAE generate chord pitches without information of exact chord positions. For display purpose, their chord labels are inferred from the generated pitches and their

chords are shown in root position. The original chords of *Colors of the Wind* are included to serve as a reference for evaluation.

Figure 1 reveals that FC and FC− produce chords that are stylistically similar to the original chords of *Colors of the Wind*. They include F and C, the tonic and dominant chords, that form the harmonic basis for the melody, which is in the key of F major. Furthermore, they include the relative minor chord Dm, which is integral to the original harmonization. Dm shares two pitch classes, namely F and A, with the F chord. This commonality ensures smooth transitions between major and minor chords within the key.

Figure 1 also illustrates how FC handles cadence. For instance, FC produces the chord transition Dm–Dm in bar 8, which is identical to the original chords. In contrast, FC− is unable to consistently produce cadences because it does not handle cadence during chord generation. For example, in bar 4, it produces Dm–F, which is not a proper cadence.

VTHarm produces a chord progression that noticeably deviates from the original chord progression. It produces B♭maj7 twice with unexpected chord progressions that are not in the original chord sequence. In bars 1 and 2, the chords change from B♭maj7 to Gm to B♭, resulting in an unusual harmonic shift. Similarly, DAT-VAE introduces surprising progressions that include Csus4 and G7. Nevertheless, it fares slightly better than VTHarm since it reproduces the tonic F chord in bar 1 that reinforces the melody's key. However, the sequences of Csus4–G7–C and Csus4–D are disruptive to the overall chord progression.

5.4 Subjective Evaluation

Subject evaluation was conducted to compare the perceived harmonization quality of various methods. 27 subjects with strong music background (play piano, at least 5 years of music training) were recruited via CloudConnect. Each subject was presented with two 8-bar songs with chord accompaniment. One of the pair was either the original manual harmonization or FlexChord's result. The other was a result of either VTHarm or DAT-VAE. The subject was asked to pick the one that was more harmonious. In addition, the subject was asked to rate how harmonious each song was on a 5-point Likert scale. The ratings were used to verify whether the subject's choice was consistent with the ratings. Each subject repeated the above procedure for 40 pairs of pop songs. These pairs included 10 unique melodies presented in different contexts. Jazz songs were omitted because very few subjects recruited were familiar with jazz harmonization.

The subjects' assessments were grouped into one of the following cases:

- O+: original harmonization is more harmonious than DNN's result.
- O−: original harmonization is less harmonious than DNN's result.
- F+: FlexChord's result is more harmonious than DNN's result.
- F−: FlexChord's result is less harmonious than DNN's result.

Table 3 tabulates the subjects' assessments. A large majority of subjects judge the original harmonizations as more harmonious than those of VTHarm DAT-VAE (68.3% and 74.7% respectively). This expected result indicates that the subjects understand and

TABLE 3. SUBJECTIVE EVALUATION FOR VARIOUS CASES.

Cases	VTHarm	DAT-VAE
O+	179 (68.3%)	195 (74.7%)
O−	83 (31.7%)	66 (25.3%)
F+	158 (60.3%)	159 (60.9%)
F−	104 (39.7%)	102 (39.1%)
O+ and F+	124 (47.3%)	132 (50.6%)
O− and F−	55 (21.0%)	63 (24.1%)
O+ and F+	34 (13.0%)	27 (10.3%)
O− and F−	49 (18.7%)	39 (15.0%)
Total votes	262 (100%)	261 (100%)

appreciate harmonization. Among them, DAT-VAE's results are the least harmonious. Similarly, most subjects judge FlexChord's harmonizations to be more harmonious than those of VTHarm and DAT-VAE (60.3% and 60.9% respectively), with DAT-VAE results being the least harmonious. In addition, more subjects judge both the original and FlexChord's harmonizations to be more harmonious than those of the DNNs (47.3%, 50.6%), compared to the other three cases. These subject test results indicate that FlexChord's harmonizations are better than those of the DNNs, and are comparable to those of the original harmonizations.

5.5 Harmonization with Custom Styles

FlexChord generates chord sequence according the chord transition model that captures harmonization style. Moreover, it performs chord generation without the need for time-consuming training. These properties allow FlexChord to perform harmonization with custom style.

To demonstrate FlexChord's ability to harmonize with custom style, Hitchcock's *Psycho* was selected as the reference music for generating spooky style. Chord pairs were extracted from *Psycho* and chord transition probabilities were computed for the spooky chord transition model.

Figure 2 illustrates the result of harmonizing the first 8 bars of *In the Hall of the Mountain King* with the spooky chord transition model. Compared to the original harmonization, FlexChord generates a higher concentration of dissonant chords, similar to the style of *Psycho*. For instance, the A#dim and F#7 chords in bars 2 and 6 clash with the melody's natural notes, intensifying the overall tension. In contrast, the original chords are triads such as Bm, C#, and D which are more consonant with the melody.

Section 5.3 presents the harmonization of the first 8 bars of *Colors of the Wind* in pop style. FlexChord can also harmonize in various styles. Figure 3 shows chords generated using two chord transition models: one based on jazz and the other based on the folk songs of Constantinidis, a Greek composer. The jazz version adds seventh chords like Dm7 and Gm7 with chromatic voice leading. The Constantinidis version generally avoids the dominant C chord except for bars 5 and 7, creating slightly unresolved harmony typical of modal music.

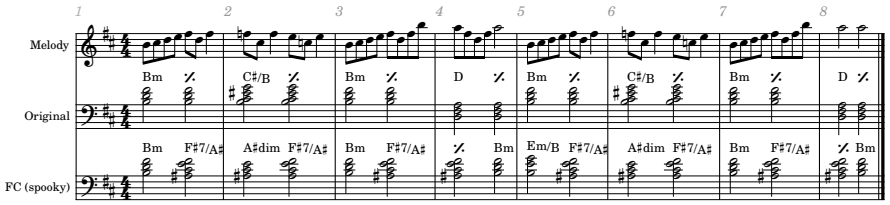

Fig. 2. Custom-style chords for the first 8 bars of *Hall of the Mountain King*.

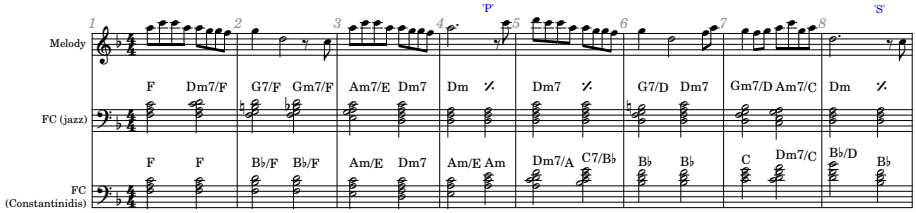

Fig. 3. Harmonization of *Colors of the Wind* in various styles.

5.6 Full-length Melody Test

FlexChord can harmonize full-length songs, taking into account the global structures of the songs. To illustrate this capability, FlexChord was tested on generating the chords for the full song of *Colors of the Wind*, which includes diverse melodic variations and intervals. For comparison, VTHarm and DAT-VAE were applied to generate the chords for individual 8-bar phrases.

Figure 4 illustrates selected fragments of the test results as it is too long to show all 62 bars of *Colors of the Wind*. For all methods, the results for the first 8 bars are identical to those of the short-phrase test (Fig. 1). With the exception of bars 15 and 16, the melody of section A is repeated exactly in section B. FlexChord (FC) repeats the chords of section A exactly in section B, due to the enforcement of repetition constraint (Algorithm 1, Lines 11, 12).

For the transition to section C (bars 16 to 17), FC creates a relevant major-major contrast from chord F to Am. Both chords share two pitches, which are A and C. Similarly, VTHarm creates a major-minor contrast with chord Dm transitioning to B♭. Both chords also share two pitches, which are D and F. However, DAT-VAE does not create contrast when transitioning to section C. Instead, it produces repeated chords of Dm–Dm.

Section C contains the chorus of the song, with the melodic climax located in bar 17 and its partial resolution in bar 18. The tension leading to the climax is built by the ascending pitches F–A–C in bar 16, which leap to F in the next octave, followed by descending notes into bar 18. FC follows the same trend by selecting chords that initially create dissonance with the melody, followed by chords that are consonant with the melody. For example, in bar 17, the pitch E of chord Am clashes with the melody pitches F and D, producing a high dissonance. This is followed by a F–Dm cadence in bar 18, which is consonant with the melody. Thus, FC effectively builds tension to the climax and then resolves it. The results of VTHarm follow the same trend but uses the

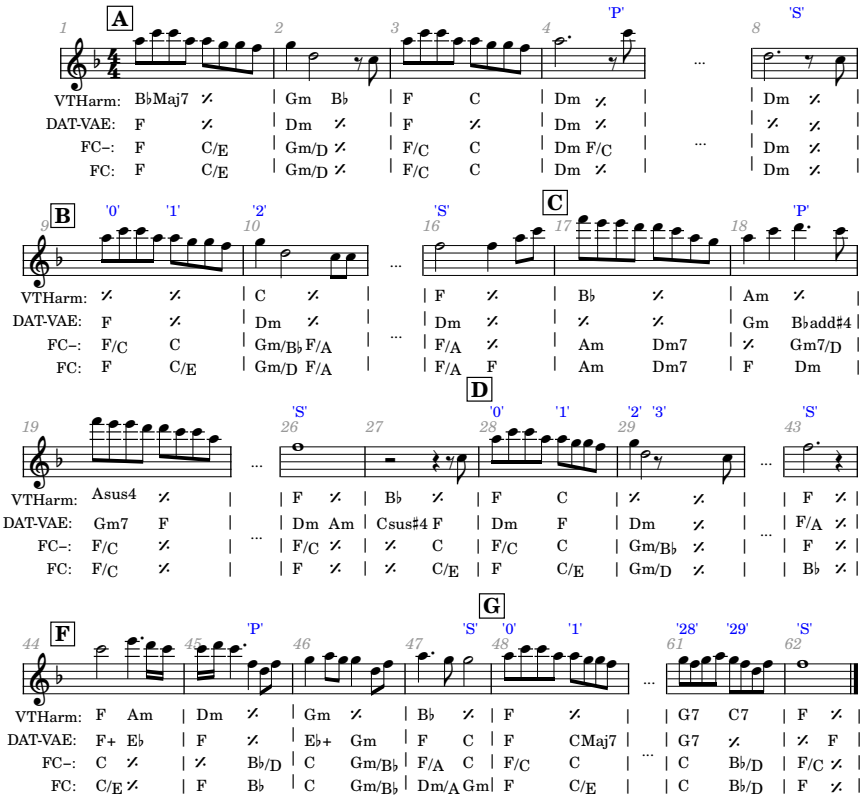

Fig. 4. Selected fragments from the test results of the full song *Colors of the Wind*.

chord transition B♭–Am instead to build and release tension. In contrast, the results of DAT-VAE go against the melodic trend. It repeats the Dm chord in bar 17 which is consonant with the melody transitions to Gm–B♭add#4 that are dissonant with the melody.

For the transitions to section F (bars 43 to 44), FC creates a tonic-dominant contrast with the transition from chord F to C. In contrast, VTHarm does not feature any chord contrast in the transition from F to F. DAT-VAE generates an F–F+ transition, which introduces a chromatic shift. This transition is harmonically incoherent with the key of F major and the pop style because F+ chord creates an unexpected dissonance.

These test results demonstrate that FlexChord can produce key-relevant chord progressions that align with the full-length melody with section contrasts to support the global melody structure. In contrast, VTHarm and DAT-VAE occasionally generate chords that disrupt the overall chord progression, resulting in a lack of or odd contrast between sections.

6 Conclusion

This paper proposes a style-specific harmonization method called FlexChord that overcomes the shortcomings of existing methods. FlexChord achieves style specificity by

producing chord sequence according to a chord transition model that captures the required style. Chord transition model is easily prepared by counting the frequency of occurrence of consecutive chord pairs in a set of example music. The example set can contain as little as a single music piece.

FlexChord applies multi-objective optimization during chord generation that does not require training. Thus, it can easily produce chord sequence of a specific style. In particular, it can harmonize a melody to custom styles.

FlexChord balances between good chord-melody matching and good chord progression that follows the required style. It caters to cadence and smooth voice leading effectively. It can generate chords for full-length melodies, complete with global music structure. Test results show that FlexChord's harmonization is better than those of comparable DNNs and is close to that of human harmonization.

As part of future work, the chords generated by FlexChord can be utilized for generating style-specific accompaniments to full-length melodies. Generation of accompaniment involves transforming the generated chords into multiple rhythmic voices that fit a specific accompaniment style. The generated accompaniment should also match the global music structure of a full-length melody.

References

[1] D. Makris, I. Kayrdis, and S. Sioutas, "Automatic melodic harmonization: An overview, challenges and future directions," in *Trends in Music Information Seeking, Behavior, and Retrieval for Creativity*, pp. 146–165, IGI Global, 2016.

[2] G. Aguilera, J. L. Galán, R. Madrid, A. M. Martínez, Y. Padilla, and P. Rodríguez, "Automated generation of contrapuntal musical compositions using probabilistic logic in derive," *Mathematics and Computers in Simulation,* vol. 80, no. 6, pp. 1200–1211, 2010.

[3] E. Gilbert and D. Conklin, "A probabilistic context-free grammar for melodic reduction," in *Proceedings of International Workshop on Artificial Intelligence and Music* at *International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[4] M. Gogins, "Score generation in voice-leading and chord spaces," in *International Computer Music Conference* (*ICMC)*, 2006.

[5] D. Herremans and K. Sörensen, "Fux, an android app that generates counterpoint," in *IEEE Symposium on Computational Intelligence for Creativity and Affective Computing*, pp. 48–55, 2013.

[6] C. Z. A. Huang and E. Chew, "Palestrina pal: A grammar checker for music compositions in the style of Palestrina" in *Proceedings of the Conference on Understanding and Creating Music*, 2005.

[7] D. Temperley, *The Cognition of Basic Musical Structures*. MIT press, 2004.

[8] A. Yilmaz and Z. Telatar, "Note-against-note two-voice counterpoint by means of fuzzy logic," *Knowledge-Based Systems*, vol. 23, no. 3, pp. 256–266, 2010.

[9] A. Gartland-Jones, "Can a genetic algorithm think like a composer?" in *Generative Art Conference*, 2002.

[10] T. Kitahara, S. Giraldo, and R. Ramírez, "Jamsketch: Improvisation support system with GA-based melody creation from user's drawing," in *Music Technology with Swing*, pp. 509–521, Springer International Publishing, 2018.

[11] S. Phon-Amnuaisuk, A. Tuson, and G. Wiggins, “Evolving musical harmonisation,” in *Artificial Neural Nets and Genetic Algorithms*, pp. 229–234, Springer Vienna, 1999.
[12] S. Phon-Amnuaisuk, G. Wiggins, et al., “The four-part harmonisation problem: a comparison between genetic algorithms and a rule-based system,” in *Proceedings of the AISB’99 Symposium on Musical Creativity*, pp. 28–34, 1999.
[13] G. A. Wiggins, G. Papadopoulos, S. Phon-Amnuaisuk, and A. Tuson, “Evolutionary methods for musical composition,” *Int. Journal of Computing Anticipatory Systems*, 1999.
[14] H. Tsushima, E. Nakamura, K. Itoyama, and K. Yoshii, “Function- and rhythm-aware melody harmonization based on tree-structured parsing and split-merge sampling of chord sequences,” in *International Society for Music Information Retrieval Conference (ISMIR)*, pp. 502–508, 2017.
[15] A. Zacharakis, M. Kaliakatsos-Papakostas, S. Kalaitzidou, and E. Cambouropoulos, Evaluating human-computer co-creative processes in music: A case study on the chameleon melodic harmonizer," *Frontiers in Psychology*, vol. 12, 2021.
[16] M. Kaliakatsos-Papakostas, K. Velenis, L. Pasias, C. Alexandraki, and E. Cambouropoulos, An HMM-based approach for cross-harmonization of jazz standards," *Applied Sciences*, vol. 13, no. 3, 2023.
[17] H. Lim, S. Rhyu, and K. Lee, Chord generation from symbolic melody using BLSTM networks," in *International Society for Music Information Retrieval Conference (ISMIR)*, pp. 621–627, 2017.
[18] Y. Chen, H. Lee, Y. Chen, and H. Wang, “SurpriseNet: Melody harmonization conditioning on user-controlled surprise contours,” in *International Society for Music Information Retrieval Conference (ISMIR)*, pp. 105–112, 2021.
[19] S. Rhyu, H. Choi, S. Kim, and K. Lee, Translating melody to chord: Structured and flexible harmonization of melody with transformer," *IEEE Access*, vol. 10, pp. 28261–28273, 2022.
[20] C. E. Sun, Y. W. Chen, H.-S. Lee, Y. H. Chen, and H. M. Wang, Melody harmonization using orderless NADE, chord balancing, and blocked Gibbs sampling," in *IEEE International Conference on Acoustics, Speech and Signal Processing, (ICASSP*), pp. 4145–4149, 2021.
[21] Y. C. Yeh, W. Y. Hsiao, S. Fukayama, T. Kitahara, B. Genchel, H. M. Liu, H. W. Dong, Y. Chen, T. Leong, and Y. H. Yang, “Automatic melody harmonization with triad chords: A comparative study,” *Journal of New Music Research*, vol. 50, no. 1, pp. 37–51, 2021.
[22] J. Zhao, G. Xia, and Y. Wang, “Domain adversarial training on conditional variational auto-encoder for controllable music generation," in *International Society for Music Information Retrieval Conference (ISMIR)*, pp. 925–932, 2022.
[23] D. Dalmazzo, K. Deguernel, and B. L. T. Sturm, “The Chordinator: Modeling music harmony by implementing transformer networks and token strategies," in Artificial Intelligence in Music, Sound, Art and Design. EvoMUSART 2024, vol. 14633 of Lecture Notes in Computer Science, pp. 52–66, Springer, 2024.
[24] M. Kaliakatsos-Papakostas and E. Cambouropoulos, Probabilistic harmonization with fixed intermediate chord constraints" in *International Computer Music Conference* (*ICMC*), 2014.

[25] H. Hild, J. Feulner, and W. Menzel, “HARMONET: A neural net for harmonizing chorales in the style of J. S. Bach," in *Neural Information Processing Systems*, 1991.
[26] J. Lewis, “Creation by refinement and the problem of algorithmic music composition," *Music and Connectionism*, vol. 212, 1991.
[27] L. Yi, H. Hu, J. Zhao, and G. Xia, “AccoMontage2: A complete harmonization and accompaniment arrangement system,” in *International Society for Music Information Retrieval Conference (ISMIR)*, pp. 248–255, 2022.
[28] C. H. Chuan and E. Chew, “Generating and evaluating musical harmonizations that emulate style," Computer Music Journal, vol. 35, no. 4, pp. 64–82, 2011.
[29] T. Greer and S. Narayanan, “Harmonize this melody: Automatic four-part harmony generation using neo-Riemannian voice-leading," in *International Society for Music Information Retrieval Conference (ISMIR), Late-Breaking Demo*, 2021.
[30] H. R. Lee and J. S. Jang, “i-Ring: A system for humming transcription and chord generation," in *IEEE International Conference on Multimedia and Expo (ICME)*, vol. 2, pp. 1031–1034, IEEE, 2004.
[31] I. Simon, D. Morris, and S. Basu, “MySong: Automatic accompaniment generation for vocal melodies," in *SIGCHI Conference on Human Factors in Computing Systems*, pp. 725–734, Association for Computing Machinery, 2008.
[32] S. A. Raczyński, S. Fukayama, and E. Vincent, “Melody harmonization with interpolated probabilistic models," *Journal of New Music Research*, vol. 42, no. 3, pp. 223–235, 2013.
[33] M. Kaliakatsos-Papakostas, K. Velenis, L. Pasias, C. Alexandraki, and E. Cambouropoulos, “An HMM-based approach for cross-harmonization of jazz standards," *Applied Sciences*, vol. 13, no. 3, 2023.
[34] C. Ames, The Markov process as a compositional model: A survey and tutorial," Leonardo, vol. 22, no. 2, pp. 175–187, 1989.
[35] P. van Kranenburg and E. Kearns, “Algorithmic harmonization of tonal melodies using weighted pitch context vectors," in *International Society for Music Information Retrieval Conference (ISMIR)*, pp. 391–397, 2023.
[36] Y. Wu and H. H. Chen, “Emotion-flow guided music accompaniment generation," in *IEEE International Conference on Acoustics, Speech and Signal Processing, (ICASSP)*, pp. 574–578, IEEE, 2016.
[37] D. Huron, “Interval-class content in equally tempered pitch-class sets: Common scales exhibit optimum tonal consonance," *Music Perception*, vol. 11, no. 3, pp. 89–305, 1994.
[38] L. L. Trulla, N. Di Stefano, and A. Giuliani, “Computational approach to musical consonance and dissonance," *Frontiers in Psychology*, vol. 9, 2018.
[39] D. M. Randel, *The Harvard concise dictionary of music and musicians*. Harvard University Press, 1999.
[40] M. Kaliakatsos-Papakostas, M. Queiroz, C. Tsougras, and E. Cambouropoulos, “Conceptual blending of harmonic spaces for creative melodic harmonisation," *Journal of New Music Research*, vol. 46, no. 4, pp. 305–328, 2017.
[41] D. Temperley, “The cadential IV in rock," *Music Theory Online*, vol. 17, 04 2011.
[42] W. E. Caplin, “The classical cadence: Conceptions and misconceptions," *Journal of American Musicological Society*, vol. 57, no. 1, pp. 51–118, 2004.
[43] D. Huron, *Voice Leading: The Science Behind a Musical Art*. MIT press, 2016.

[44] D. Huron, “Tone and voice: A derivation of the rules of voice-leading from perceptual principles," *Music Perception*, vol. 19, no. 1, pp. 1–64, 2001.
[45] M. Ehrgott, *Multicriteria optimization*, vol. 491. Springer Science & Business Media, 2005.
[46] K. Miettinen, *Nonlinear multiobjective optimization*, vol. 12. Springer Science & Business Media, 1999.
[47] Hiehn, Steve, “Chord melody dataset." https://github.com/shiehn/chord-melody-dataset, 2019. Accessed: 2022-10-06.
[48] Anderson, Chris and Carlton, Dave and Miyakawa, Ryan and Schwachhofer, Dennis, “Hooktheory." https://www.hooktheory.com, 2021. Accessed: 2022-10-06.