

STYLE-SPECIFIC HARMONIZATION AND ACCOMPANIMENT

by

LEW YIN HUI, STEPHANIE

**A THESIS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY**

in

COMPUTER SCIENCE

in the

SCHOOL OF COMPUTING

of the

NATIONAL UNIVERSITY OF SINGAPORE

DECEMBER 2025

Supervisors:

Associate Professor Leow Wee Kheng, Main Supervisor
Assistant Professor Kathleen Agres (Centre of Music and Health), Co-Supervisor

Examiners:

Associate Professor Wang Ye
Associate Professor Sim Mong Cheng, Terence

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Lew Yin Hui, Stephanie

Abstract

Automated music composition (AMC) aims at applying computer algorithms to create music. It enables musical novices with limited knowledge to create new music and helps professional musicians to enhance their creative process. More importantly, it fosters collaboration between musicians and non-musicians, leading to diverse applications like music for videos, games, and therapy.

AMC is not a single, monolithic task. In the broadest sense, AMC can be divided into two main tasks: (1) generation of music structure and melody and (2) generation of chords and accompaniment. The first task of generating melody is well-studied. The second task is important for generating music with multiple voices that also harmonize with each other. Therefore, this thesis focuses on the task of generating chords and accompaniment given the melody of a target music piece, complete with its music structure.

A variety of methods have been developed to generate chords and accompaniment, but they have some shortcomings. Most existing methods 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 chord generation 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 thesis proposes a method of harmonization called FlexChord that overcomes the shortcomings of existing methods. FlexChord achieves style specificity by producing chord progressions that match the style of a selected set of example music, which can be as little as a single music piece. It applies non-iterative, multi-objective 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.

This thesis also proposes a method called FlexFlow for generating accompaniment to multi-section melodies using chords generated by FlexChord. It achieves style specificity by producing accompaniment that matches the style of a selected set of example music. Thus, it can easily produce monophonic or polyphonic accompaniment of a specific style or a custom style. It applies direct optimization during accompaniment generation that does not require training. The requirement of cleaning, annotating and training on large datasets is avoided. In addition, it can handle phrase and section closure. It can also generate complementary or contrasting sections consistent with the global music structure of the given melody.

FlexChord and FlexFlow show strong performance as flexible methods for harmonization and accompaniment generation respectively. Both quantitative results and musiological evaluation show that they perform better than comparable existing methods. In addition, results of human perception tests show that their generated chords and accompaniments are better than those of comparable existing methods and are close to human-composed chords and accompaniments.

Publication List

The following papers are derived from the work presented in this thesis:

1. Stephanie Y. H. Lew, Wee Kheng Leow and Kat R. Agres, Style-Specific Harmonization by Multi-Objective Optimization without Training, in *Proceedings of the International Conference on New Music Concepts (ICNMC)*, vol. 12, Accademia Musicale Studio Musica, 2025.
2. Stephanie Y. H. Lew, Wee Kheng Leow and Kat R. Agres, Style-Specific Melody Harmonization via Multi-Objective Optimization, *International Journal of Music Science, Technology and Art (IJMSTA)*, vol. 7, no. 2, 2025.

Acknowledgments

I am deeply grateful to my main supervisor, Professor Leow Wee Kheng, whose guidance, mentorship and belief in my work were essential to this research. He taught me how to approach and solve problems independently and how to communicate research effectively through writing and presentations. His patience and support were instrumental in the completion of this thesis. I also extend my sincere thanks to my thesis committee members, Professor Wang Ye and Professor Terence Sim, for reviewing my work and providing insightful feedback that strengthened this thesis. I thank Professor Leong Hon Wai for his early advice on developing a structured approach to problem solving and effective methods for reporting results.

Special thanks go to my co-supervisor, Professor Kathleen Agres for drawing my attention to the melody harmonization problem, which laid the foundation for subsequent work on accompaniment generation. Her guidance on conducting listener studies and polishing conference submissions was invaluable. She broadened my perspectives on the interdisciplinary connections between music and health research.

Later in my PhD, I had the chance to work with Professor Ng Teck Khim and assist his undergraduate student on her UROP project. I am sincerely grateful for his support and for giving me opportunities to explore related work. I deeply appreciate his guidance and generous sharing which have enriched many aspects of my personal and professional development.

I am grateful to my research colleagues, Jeffry, Gizem and Viet Nam as well as teaching colleagues Eric and Wu Biao, for their advice on technical and personal matters. I also thank Li Chen, Zhao Na and others in Media Research Lab 4 with whom I enjoyed conversations and lunches prior to the pandemic disruptions.

Most importantly, I thank my husband for his unwavering support and sacrifices, and my parents for instilling strong values, discipline and life skills. Embracing early motherhood during this research offered unique growth, perspective, and resilience. My young daughter later became a constant joy and inspiration that made the end of this PhD journey profoundly rewarding.

To all those mentioned above, and to everyone who supported me in ways both large and small, I extend my heartfelt gratitude.

Contents

Abstract	i
Publication List	iii
Acknowledgments	iv
List of Figures	vi
List of Tables	vii
1 Introduction	1
1.1 Motivation	1
1.2 Research Objectives	2
1.3 Regarding Music Style	4
1.4 Contributions of Research	4
1.5 Organization of Thesis	5
2 Music Background	6
2.1 Basic Elements of Music	6
2.2 Formal Structure of Music	8
2.3 Principles of Harmonization	11
2.4 Principles of Accompaniment	12
3 Related Work	14
3.1 Harmonization	14
3.1.1 Rule-based and Grammar-based Methods	14
3.1.2 Evolutionary Algorithms	15
3.1.3 Deep Neural Networks	16

3.1.4	Markov Models	18
3.1.5	Direct Optimization	19
3.1.6	Summary	20
3.2	Accompaniment Generation	23
3.2.1	Rule-based and Grammar-based Methods	23
3.2.2	Evolutionary Algorithms	24
3.2.3	Deep Neural Networks	25
3.2.4	Markov Models	27
3.2.5	Hybrid Methods	27
3.2.6	Summary	28
3.3	Performance of Deep Neural Networks for Music Generation	29
4	Harmonization	32
4.1	Representation of Music Elements	32
4.2	Problem Formulation	34
4.2.1	Chord Progression and Harmonization Style	34
4.2.2	Chord-Melody Matching	35
4.2.3	Cadence	36
4.2.4	Voice Leading	36
4.2.5	Constraint and Optimization Objectives	36
4.3	Multi-Objective Optimization	37
4.3.1	Overview	37
4.3.2	FlexChord Harmonization	39
4.3.3	Analysis of Time Complexity	41
4.4	Experiments and Discussions	41
4.4.1	Data Preparation	41
4.4.2	Quantitative Evaluation	43
4.4.3	Musicological Evaluation	48
4.4.4	Listener Study	50
4.4.5	Flexible Harmonization Style	52
4.4.6	Full-Length Test	56
4.4.7	Summary	58

5	Accompaniment Generation	60
5.1	Representation of Music Elements	60
5.2	Problem Formulation	61
5.2.1	Accompaniment Style	61
5.2.2	Pitch Progression	62
5.2.3	Accompaniment-Chord Matching	63
5.2.4	Accompaniment-Melody Matching	63
5.2.5	Closure	64
5.3	Algorithm Design	64
5.3.1	FlexFlow Accompaniment Generation	64
5.3.2	Extension to Polyphonic Accompaniment	69
5.3.3	Comparison to Existing Methods	70
5.4	Experiments and Discussions	72
5.4.1	Data Preparation	72
5.4.2	Quantitative Evaluation	73
5.4.3	Listener Study	78
5.4.4	Musicological Evaluation of FlexFlow Variants	80
5.4.5	Musicological Evaluation of Various Methods	82
5.4.6	Accompaniment in Various Styles	86
5.4.7	Summary	88
6	Future Work and Conclusion	90
6.1	Future Work	90
6.1.1	FlexChord	90
6.1.2	FlexFlow	91
6.2	Conclusion	91
	References	94

List of Figures

1.1	Step function of a melody.	3
2.1	Elements of a music piece.	7
2.2	Melody as a single voice.	7
2.3	Melody with one supporting voice.	7
2.4	Melody with accompaniment of four voices.	8
2.5	Lead sheet containing melody and chord labels.	8
2.6	Formal music structure of Bach's <i>Minuet in G</i>	10
2.7	Collapsed music form of <i>Twinkle, Twinkle, Little Star</i>	10
4.1	Step function of a melody.	33
4.2	Chords generated by various methods for <i>Colors of the Wind</i>	49
4.3	Human-composed and FC jazz chords for <i>Israel</i>	53
4.4	Harmonization of <i>Colors of the Wind</i> in various styles.	54
4.5	Harmonization of <i>La Vie en Rose</i> in various styles.	54
4.6	Spooky chords for <i>Hall of the Mountain King</i>	55
4.7	Happy and sad sounding accompaniments by FF.	56
4.8	Test results for the full song of <i>Colors of the Wind</i>	57
5.1	Transposition of reference accompaniment.	65
5.2	Pitch adjustment in accompaniment.	67
5.3	Section closure in accompaniment.	68
5.4	An example of polyphonic accompaniment.	69
5.5	Poly-pitch adjustment in accompaniment fragment.	71
5.6	Accompaniment of <i>Sentimental Me</i> by FF variants.	81
5.7	Accompaniment generated by various methods for <i>Sentimental Me</i> . . .	82
5.8	Accompaniment generated by various methods for <i>Sentimental Me</i> . . .	83

5.9	Accompaniment generated by various methods for <i>Sentimental Me.</i>	85
5.10	Closing accompaniment generated by various methods for <i>Sentimental Me.</i>	86
5.11	Accompaniment generated by FF in various styles.	87
5.12	Happy and sad sounding accompaniments by FF.	87

List of Tables

3.1	Comparison of methods for harmonization.	21
3.2	Comparison of methods for accompaniment generation.	30
4.1	Pitch dissonance depends on pitch interval.	35
4.2	Summary statistics of the datasets used for harmonization.	43
4.3	Quantitative evaluation for chord diversity and chord-melody match. .	47
4.4	Results of the listener study on harmonization.	52
5.1	Summary statistics of the datasets used for accompaniment generation.	73
5.2	Quantitative evaluation for accompaniment-melody match.	76
5.3	Results of the listener study on accompaniment generation.	79

Chapter 1

Introduction

1.1 Motivation

Automated music generation (AMG) or automated music composition (AMC) aims at applying computer algorithms to create music. It enables musical novices with limited knowledge to create new music and helps professional musicians to spark new creative ideas by generating a wide range of musical ideas quickly, speeding up the creative process. More importantly, it can facilitate new forms of collaboration between musicians and non-musicians resulting in a variety of applications such as music for videos, games and clinical therapy [HS13].

A variety of methods have been developed to generate music compositions automatically in the past decade. For example, rules and grammars [HS13; CLC13; Agu+10; WC16b], and Markov models [CC11b; BC15; Mo22] are used to generate accompaniments, whereas integer programming is used to incorporate constraints for the accompaniment in the music structure [CSH18]. Genetic algorithms are used to generate variations of melody and accompaniment [GJ02; LT12; KGR18]. More recently, deep neural networks are trained to generate music in the style of Bach [Lia16; HPN17; Hua+17], pop [Zhu+18; Zhu+20; Che+21; Rhy+22; Yi+22] and folk music [ZX21]. More details of automated music generation are available in [HCC17; BHP20; Civ+22].

Despite the success of recent deep neural networks (DNNs) for AMC, they still have shortcomings. For example, they are restricted to training and testing on 8-bar phrases rather than complete music pieces [BP20]. Thus, they do not handle global structure of complete music pieces. They lack meaningful controls for a composer to constrain the music according to a specific style [Dég+22]. According to

performance studies on deep neural networks for music [Yin+21; Yin+23], the state-of-the-art DNNs tend to produce copies of training examples and their performance improvement is negligible compared to Markov methods for music generation.

In addition, existing methods typically handle music style at the coarse level of genre such as pop and jazz. They can generate music for different genres but not necessarily music that conveys the styles of various composers and emotions. Most existing methods are trained on a training set of a specific style. Training on a mixed-style set that contains examples of different styles causes the methods to mix up various styles and produce results that are diluted in style. To accommodate to different styles, they need to train a different version of the method on the training examples of a different style, increasing the total training time needed. Notably, existing deep neural network methods require human preparation of training sets and application of iterative optimization during training, which are tedious and time-consuming. Moreover, they cannot cater to cadence and voice leading, and only work on 8-bar phrases rather than full-length music pieces. Thus, they cannot handle global music structure. This thesis research aims to address these shortcomings.

1.2 Research Objectives

The overall research goal is to develop an AMC framework. AMC is not a single, monolithic task. It consists of several functional components that aim at the four essential elements of music, namely melody, harmony, rhythm and timbre [HCC17]. In the broadest sense, AMC can be divided into two main tasks: (1) generation of music structure and melody and (2) generation of chords and accompaniment. The problems of generating music structure and melody are considered together as a single task as music structure relates to phrase repetition and contrast in the melody. The combination of phrase repetition and contrasts lead to sections and overall music form in complete music. The second task is the generation of chords and accompaniment that support a melody. The first task of generating melody is well-studied [Jaq+16; JKW17; Lop+17; CSG17; Wil+17; JKA17; Goi+19; HH20; Hua+20; SGA20; HBE20; SW20; GDG21; TT21; Sab+21; Sut+21; Dai+21]. Therefore, this thesis focuses on the second task of generating chords and accompaniment given the melody of a target music piece, complete with its music structure. Chord generation

The figure displays a musical score with three staves. The top staff is the Melody. Below it are two staves labeled (a) Chords and (b) Accompaniment. The Chords staff shows stacks of simultaneous pitches for each measure, with chord symbols (F, Dm, Am, Bb, Gm7, C7, Dm) written above. The Accompaniment staff shows a pitch sequence derived from the same chords. Labels include 'section label' (A8), 'chord symbol' (F, Dm, Am, Bb, Gm7, C7, Dm), and 'structure labels' (P, S).

Figure 1.1: (a) A series of chords as stacks of simultaneous pitches across time.
(b) Accompaniment as a pitch sequence derived from the same chords.

refers to selecting a series of chords that harmonize with the melody. In contrast, accompaniment generation produces one or more pitch sequences that support the melody and maybe derived from target chords. Figure 1.1 illustrates the differences between chords and accompaniment.

The chords and accompaniment along with the melody constitute music with multiple voices. Another related task in the generation of music with multiple voices is chorale generation. In chorale generation, chords are expanded into individual notes for each of the four standard voices, namely soprano, alto, tenor and bass, in traditional chorales [Ebc88]. It is a specific type of accompaniment generation that has been extensively studied [TA99; HFM91; PAW+99; AW04; GM07; YG07]. Moreover, well established rules in traditional chorale music can be easily enforced on each individual accompaniment voice to produce chorale music. Hence, chorale generation is regarded as a simple case of accompaniment generation and is excluded from the scope of this work.

Other important aspects to consider for AMC are instrumentation and rearrangement. Instrumentation is the assignment of instruments to voices or parts in a composition, while rearrangement is the reassignment of notes in the voices according to the pitch ranges of the instruments. They are important as the selection of instruments for rendering musical parts influences the overall timbre and sound quality of the music that is played. In this thesis, piano is chosen as the target instrument for a multi-voice composition as it offers a wide range of 88 pitches that achieves rich sound quality. Other forms of instrumentation such as string quartet and modern band with unpitched drums may be generalized from piano accompaniment, and are excluded from this work.

1.3 Regarding Music Style

In music composition, style is characterized by various elements such as melody, rhythm, harmony, voice leading, and accompaniment. The concept of style can be understood at multiple levels. At the coarsest level, style refers to the genre of music such as pop, jazz and classical. Style can also refer to music in specific historical periods such as Baroque, Classical, Romantic, etc.

At a fine level, style can refer to the composition style of individual composers. For example, Beethoven, Mozart and Haydn belong to the Classical period but their composition styles are very different. For a prolific composer, the style of different compositions can differ significantly. For example, Chopin’s Mazurkas and Nocturnes have very different styles. At the finest level, each piece of music can have a different style. For example, Chopin’s Mazurkas No. 1 and 2 have different styles even though they are both Mazurkas [Col+11].

1.4 Contributions of Research

This research work aims to achieve style-specific harmonization and accompaniment generation. Its contributions include the following:

- Generate harmonization and accompaniment of required style at various style levels, including custom styles. Existing methods require re-training to accomodate to new styles.
- Achieve style-specificity using as little as a single example music piece.
- Apply direct optimization that does not require training, thus eliminating the need to collect, clean and annotate large amounts of training data.
- Handle cadence and voice-leading that are absent in existing work.
- Generate harmonization and accompaniment for full-length melodies, taking into account global music structure. On the other hand, existing methods cannot handle global music structure adequately.

As discussed in Section 1.3, music style is a complex concept. It can be challenging for a user to provide information of the desired style. This issue will be addressed in Chapters 4 and 5.

1.5 Organization of Thesis

To understand how chords and accompaniment support melody, it is necessary to first understand the elements of music and structure (Chapter 2). Music elements such as melody, chords, voices and rhythm are involved in harmonization and accompaniment generation. The principles that guide quality harmonization and accompaniment are discussed in Section 2.3 and 2.4. Related methods for harmonization and accompaniment generation are reviewed in Chapter 3. Chapter 4 discusses the proposed method for chord generation and Chapter 5 presents the proposed method for accompaniment generation. Finally, Chapter 6 outlines the direction of future work and concludes the thesis.

Chapter 2

Music Background

2.1 Basic Elements of Music

The melody and accompaniment of a music piece comprise several elements. The most basic element in a melody is a **note**, which has three main attributes: pitch, duration and onset. **Pitch** is the perceptual property that relates to the frequency of sound and **duration** informs the length of time that a pitch is played. **Onset** is the starting time of a note. When no note is present, a **rest** is indicated. Like notes, rests have varying durations.

In conventional Western musical notation (Fig. 2.1), a sequence of notes in a melody can be divided into segments called **bars**. A bar's relative duration is encoded by the **time signature**, which specifies how many beats are contained in each bar, and which note value is equivalent to a beat. The actual duration of a bar is encoded by the **tempo**, which indicates the number of beats per minute. Melody is usually perceived by the listener as a distinct entity because of its rhythm and unique sequence of pitches. **Rhythm** refers to the pattern of sounds and silences that occur over time.

Two or more notes played concurrently form a **chord**. The notes in a chord can be played sequentially in rising and descending order. They can also be reordered in different **chord inversions**. For example, a C major chord consisting of C, E and G notes can take the **root position** (C-E-G), 2nd inversion (E-G-C) or 3rd inversion (G-C-E). These chord positions are labeled as C, C/E and C/G respectively. In addition, the notes in a chord can be packed tightly in closed position (e.g., C4-E4-G4 in octave 4) or span more than an octave in open position (e.g., C3-G3-E4 across octave 3 and 4).

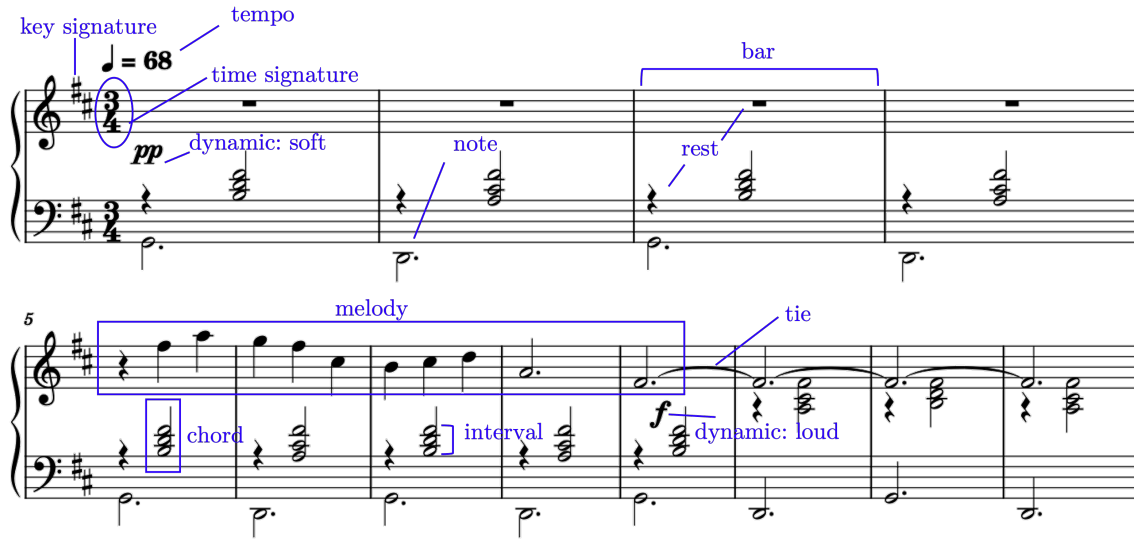


Figure 2.1: Elements of a music piece. They include information about pitch, rhythm, tempo of the melody and the chords. This is an excerpt from *Gymnopédie No. 1* by Erik Satie.



Figure 2.2: Melody as a single voice. Figure 2.3: Melody with one supporting voice.

A chord can also be represented by a chord symbol in Roman numeral which encodes chord types with respect to a music key. For example, I, II, III are major chords, i, ii, iii are minor chords, I^+, II^+, III^+ are augmented chords and $I^\circ, II^\circ, III^\circ$ are diminished chords. Note that chord symbols do not indicate the actual position of a chord. Depending on the arrangement of its pitches, a chord can be in root position or an inversion.

The notes of a chord sounding together forms **harmony** and a sequence of chords in motion forms a **chord progression** which functions as chordal accompaniment to the melody. A chord progression of at least 2 chords that ends a phrase or section (Section 2.2) of a piece of music is called cadence. **Cadences** function like punctuation at the end of phrase, giving music phrases a sense of partial or full resolution. When chords are present or assigned, notes in the melody or



Figure 2.4: Melody with accompaniment of four voices. The accompaniment includes bass and three inner voices.



Figure 2.5: Lead sheet containing melody and chord labels.

accompaniment can also be referred to as **chord tones**, if they share the same pitch name as a chord pitch, or **non-chord tones**, otherwise. Non-chord tones that transition between chord tones are called **passing tones**.

Voice refers to a sequence of notes. For example, the melody itself is a voice (Fig. 2.2). Music that consists of melody alone is monophonic music. Some music consists of melody and accompaniment that each appears as a single voice (Fig. 2.3). Accompaniment can also consist of multiple voices. For example, the music in Figure 2.4 consists of a melody, bass voice and inner voices that comprises a sequence of triad chords. This example effectively has 5 voices. A lead sheet is different from an accompaniment. In a lead sheet, the chords are located at the beginning of the bars (Fig. 2.5). In an accompaniment, the chords can appear anywhere that is appropriate.

Voice leading refers to the smooth transition of notes within a voice. According to the pitch proximity principle [Hur01; Hur16], transitions that minimize voice fluctuation is considered as efficient or good voice leading. Voice fluctuation is the average difference between the pitches of successive notes in a voice.

2.2 Formal Structure of Music

Music can be organized hierarchically in the **form** of sections, phrases and bars. Together, these musical entities make up the formal structure of music with 3 distinct levels. Figure 2.6 illustrates a music piece that contains two sections. Each section contains 4 phrases that span 4 bars each. For sufficiently long music, a phrase may have more than 4 bars and may be organized into sub-phrases. In shorter music, such as *Twinkle Twinkle Little Star*(Fig. 2.7), a section may contain only one phrase.

A phrase may be followed by a repeat or a variation of itself, or a contrasting phrase. A phrase variation resembles the original phrase, with some minor changes to pitch, rhythm or chord. On the other hand, a contrasting phrase is melodically or harmonically distinct from the original phrase. A combination of repeating, varying or contrasting phrases form a distinct section which is typically labeled using letters (e.g., A, B). The label includes a number if it is a variation of another section (e.g., A2). Repeating, contrasting and varying sections are arranged into a **music form**.

Typical music forms include binary (A B), ternary (A B A) and rondo (A B A C A). An example of a music piece in binary form is Bach's *Minuet in G major* (Fig. 2.6), where the opening section is concluded with a contrasting section. On the other hand, music in ternary form such as *Twinkle, Twinkle, Little Star* (Fig. 2.7) begins with a main theme, transitions to a contrasting section and concludes with the main theme. Music in rondo form can be viewed as an extension of either ternary form or binary form, where the main section (referred to as either "section A" or "refrain") returns throughout a music piece and is juxtaposed with contrasting sections (referred to as "section B", "section C", or "episodes"). An example of music in rondo form is Beethoven's *Für Elise*, where the main theme appears three times and is separated by contrasting sections.

Existing literature often use the terms short-term structure and long-term structure to refer to music structures of different length. For consistency, this thesis proposal uses these terms as follows:

- Short-term structure: relationship between nearby pitches.
- Long-term structure: relationship between music phrases.
- Formal structure: relationship between music sections.

Section A
(bars 1-16)

Phrase 1
bar

Phrase 2

G (key): I

I – V
(HC in G)

9

Phrase 3

Phrase 4

V – I
(PAC in G)

Section B
(bars 17-32)

17

Phrase 5

Phrase 6

D (key): V
(HC in D)

V – I
(PAC in D)

25

Phrase 7

Phrase 8

G (key): V
(HC in G)

V – I
(PAC in G)

Figure 2.6: Formal music structure of Bach’s *Minuet in G*. It consists of two sections, namely Section A and Section B. Each section contains 4 phrases and each phrase contains 4 bars. At the phrase level, typical cadences such as Half cadence (HC) and Perfect Authentic Cadence (PAC) are found at phrase endings.

Section A

Section B

Section A

Phrase 1

Phrase 2

Phrase 3

bar 1 2 3 4 5 6 7 8 9 10 11 12

Figure 2.7: Collapsed music form of *Twinkle, Twinkle, Little Star*.

2.3 Principles of Harmonization

Harmonization or melody harmonization is an important functional aspect of automated music composition [HCC17]. It is the process of creating chords to match and enhance a given melody.

The key principles of harmonization include the following:

1. **Chord-Melody matching:** Chord-melody matching is the process of finding chords that go well with a melody. The quality of chord-melody matching is mainly influenced by chord-melody pitch relationships and style-specific elements aligned with the composer's preferences. Chord-melody pitch relationships can be measured in terms of dissonance, a sense of tension caused by the intervals between chord and melody pitches. The opposite of dissonance is consonance, which contributes to a sense of resolution. Music that is highly consonant can be considered boring while prolonged dissonances can be jarring and undesirable. Good quality chord-melody matching is dependent on the composer's preferred balance between dissonance and consonance.
2. **Chord Progression, Cadence and Voice Leading:** Chord progression refers to the sequence of chords in music and cadence refers to the two chord progressions located at phrase or section endings. Voice leading refers to the smooth and logical movement of individual melodic voices within a chord progression. It ensures that each voice moves smoothly from one chord to the next, creating a pleasing and coherent harmonic flow. Cadences and stylistically appropriate voice leading in bass line contribute to good quality chord progressions.
3. **Harmonic Coherence:** Harmonic coherence refers to the consistency and logical flow of chords within a piece of music. It ensures that chord progressions and voice leading make sense within the context of the melody and the key.
4. **Style Specificity:** Different musical styles have a specific set of harmonic conventions and tendencies. Understanding and adhering to these conventions can help create harmonizations that are stylistically appropriate and effective. A melody can be harmonized with different chord progressions that reflect

different styles. A composer can choose to harmonize a melody by the chord progression that suits his preferred style.

By applying these harmonization principles, chords that match and enhances the melody can be applied to enrich the overall music composition.

In addition, harmonization enhances the music structure in several ways:

1. **Phrases and Sections:** Harmonization helps to shape the phrases and sections within a piece. The choice and arrangement of chords can create cadences (points of harmonic resolution) that mark the end of phrases or sections.
2. **Form:** Harmonization also helps to enhance different sections of a musical form. For example, in a typical form like AABA, each section may have a distinct harmonic progression, with the A sections often sharing a similar harmonic framework and the B section providing contrast. The choice and progression of chords in each section contribute to the overall form and provide a sense of organization and coherence.
3. **Contrast:** Harmonization can enhance contrasts within music through changes in harmonic progression that introduces local contrasts between phrases.
4. **Development:** Harmonization can aid the development of music composition through the transformation and elaboration of harmonic material such as chords, chord progressions and voicing. The transformation of relationships between a series of chords and the music key creates a sense of tonal stability or instability. In addition, the elaboration of voicings between chords create variety and interest in music texture.

2.4 Principles of Accompaniment

Accompaniment is the auxiliary part of composition designed to support the main melody in music. It can consist of one or more voices. Depending on the number of written voices, the accompaniment can be played by a monophonic instrument like violin or polyphonic instrument like piano. The key principles of accompaniment include the following:

1. **Accompaniment-Melody matching:** Accompaniment-melody matching refers to the degree of match between the accompaniment and the melody pitches. Like harmonization, the quality of accompaniment-melody matching is mainly influenced by the pitch relationships between accompaniment and melody, and the aesthetic preferences of the composer. Accompaniment-melody matching can be facilitated by generating accompaniment according to the chords that match the melody.
2. **Rhythm Matching:** Rhythm matching refers to the degree of match between the rhythm of accompaniment and the rhythm of a melody. It determines the overall pace of music and gives a sense of momentum and drive throughout a music piece. The quality of rhythm matching is subject to the composer's preferences.
3. **Harmonic Coherence:** Harmonic coherence refers to the consistency and smooth progression of accompaniment pitches. In accompaniment generation, this entails producing pitch progressions that support the melody in a way that is harmonically consistent.
4. **Section Closure:** Section closure refers to the musical elements that mark the ends of phrases and sections in the accompaniment, often realized through sustained pitches with cadences.
5. **Style Specificity:** Like harmonization, style-specific conventions and tendencies in accompaniment can vary among different composers. The accompaniment for a melody is not restricted to one style. A composer can choose to support a melody by the accompaniment that suits his preferred style.

Applying these principles allows the accompaniment to support and enrich the melody. In addition, accompaniment enhances the music structure in a manner similar to chords.

Chapter 3

Related Work

The terms “harmonization” and “accompaniment” refer to different music concepts although some existing works use them interchangeably [Hua+18]. Harmonization refers to the matching of chords and melody in terms of their pitch relationships. On the other hand, accompaniment refers to the one or more voices that support the melody via chordal harmonization, rhythm and music texture. Hence, this chapter organizes the related works of harmonization and accompaniment generation in separate sections. Section 3.1 discusses existing harmonization methods that generate chord symbols or collections of pitches as held chords. The related task of generating additional voices for chorale is omitted. Section 3.2 discusses existing methods for generating accompaniment. Most of them take melody and generated chords as inputs.

3.1 Harmonization

Harmonization methods can be grouped into five categories, namely rule-based methods, evolutionary algorithms, Markov models, deep neural networks and direct optimization. The first two categories consist of older methods that are also reviewed in [MKS16].

3.1.1 Rule-based and Grammar-based Methods

Methods in this category use symbolic rules and grammars to generate a sequence of chords. These rules and grammars describe the structural relationships between various music elements including chord-melody matching and chord progression. They contain music notes and chords as atomic elements. Rules can be combined to represent long-term and formal structures such as phrases and sections. The rules

and grammar are manually defined based on one or more music examples. During chord generation, they are applied to generate chords that match the melody and chord progression.

Rules and grammars have been used to generate harmony as a sequence of chord symbols in the style of Palestrina [Agu+10; Ebc88; GC07; Gog06; Gwe02; HS13; HC05; YT10], Bach [GC07] and Schoenberg [AM09]. They are usually discrete rules [Ebc88], but can be extended to fuzzy rules [Gwe02; YT10] and probabilistic rules [Agu+10].

The strength of rule-based and grammar-based methods is their ability to capture short-term, long-term and formal structures of the example music. In addition, by constraining the harmony generation system with encoded rules and grammar, note sequences of similar harmonies can be easily generated.

Despite their strength, rules or grammars tend to be not comprehensive enough for generating all the cadences or chord sequences for harmonization. Although the rules and grammar are well defined for certain styles such as popular music and Bach chorale, they cannot be easily generalized to other complex styles unless probability is included to make probabilistic or fuzzy rules. Moreover, it is difficult to determine what combination of probabilities can be added to the rules as they may also differ between music styles.

3.1.2 Evolutionary Algorithms

Methods that use evolutionary algorithms [GJ02; KGR18; PAW+99; PATW99; Wig+99] offer some flexibility by maintaining a population of harmonic solutions to choose from, based on a fitness function. The fitness function describes the goodness of fit for chord-melody matching and chord progression. They repeat a cycle of 3 stages, namely evaluation, selection and reproduction with variation. In the evaluation stage, candidate solutions generated from user-specified examples or a random process are evaluated using a fitness function that captures chord-melody matching and chord progression. In the selection stage, a new set of candidate solutions are probabilistically copied from the old set in proportion to its fitness. In the reproduction stage, some operators are applied to the set of candidates to generate new candidate solutions. This three-stage evolution cycle repeats until

music pieces with high fitness is obtained. For effective evolution, a large population of possible music of a wide range of varieties is needed.

Methods using evolutionary algorithms differ mainly in the choice of fitness function. They may include weighted sum of features related to chord-melody matching, pitch intervals between voices, chord progression between two adjacent chords [PAW+99; PATW99; Wig+99] and distance between target music and training examples [GJ02]. Another variant, known as multi-objective evolutionary algorithm, involves balancing between two fitness functions [DPZZ20; FG11], one for melody and the other for chords separately. With evolutionary algorithms, new variations of given music can be produced easily. This is achieved by the re-selection of music phrases from existing musical pieces for modification, thus retaining short-term and possibly long-term structure and chord progression patterns.

Evolutionary algorithms have some shortcomings. Firstly, there is no guarantee that the optimal solution will be found. In fact, they are not able to deterministically enable harmonic patterns to evolve into concrete formal structures with contrasting sections because there is no mechanism for constraining musical phrases into a distinct sections with appropriate cadences. Another shortcoming is the difficulty of defining an appropriate fitness measure for the overall harmonization result, which needs to be changed to suit different styles.

3.1.3 Deep Neural Networks

The third category of methods uses deep neural networks to generate a sequence of chords that fit the chord-melody mapping and chord progressions of training samples [Che+21; Ji+23; LRL17; Rhy+22; Sun+21; Yeh+21]. These neural networks encode a note or chord as a numeric value. They learn a highly complex nonlinear mapping between the inputs and corresponding training outputs. During training, the weights of the neural networks are learnt using stochastic gradient descent and backpropagation. During the generation phase, the network’s output is synthesized from the features extracted from the input and the features learned from training examples by feedforward neuron activation.

Recent families of neural networks used for chord generation include long-short-term networks (LSTMs), Transformers and variational autoencoders (VAEs). LSTMs

have cyclical links between neurons that serve as a memory structure and maintain information in memory for longer periods of time. As a result, longer dependencies between states can be captured by the neural networks. The LSTM in [ZL21] generates chords automatically for specific melody segments. The methods in [LRL17; Sun+21; Yeh+21] utilize a variant of LSTM model called bidirectional LSTM to predict a chord for each bar given a melody represented in a pitch-class histogram. The method in [Yeh+21] augments the chord information in the training set with simplified harmonic function labels from music theory, whereas the method in [Sun+21] masks parts of the chord sequence during training to steer the model to generate new chords at masked positions.

Transformer networks are encoder-decoder models that use attention layers to relate different positions of a single sequence. This approach improves the networks' awareness of long-term structure. In [Rhy+22], Transformer networks are used to map melody notes to chords which are represented as either polyphonic pitch onset events or chord symbols. In addition, [Rhy+22] uses a variant of Transformer with an additional encoder to capture the relationships between melody, chords and music key. On the other hand, the VAE in [Che+21] learns the mapping between chord emotion, chords and melody. It generates chords that match the target melody and emotion.

Using deep neural networks for chord generation can be advantageous in some settings. Firstly, they do not require much knowledge of music theory. Suitable chord progressions for a given melody are learnt from existing melody-chord pairs in the training examples. Secondly, they can sometimes generate unexpected chords outside the melody key that are still harmonious to the melody [Sun+21].

Deep neural networks have several limitations. Firstly, a large training set is required to train a deep neural network. This can cause the harmonization styles of the individual training samples to be mixed and diluted in the harmonization result. Secondly, chord variety is limited to chord types in training examples. This is evident in [Sun+21; Yeh+21] where chord types are restricted to simple triads or a few seventh chords. Therefore, their chord progressions are stylistically limited. Moreover, these networks require additional data pre-processing such as transposing melodies and chords to the C key [Rhy+22]. Trained networks can only harmonize a melody with the specific style of the training set. They require re-training or

fine-tuning to harmonize in other styles [Yi+22].

3.1.4 Markov Models

Like deep neural networks, Markov models [Pac; PEB06; Tsu+17] also learn chord-melody matching and chord progression from training samples. However, they require smaller training sets than those used for deep neural networks. They describe the statistical properties of the change of state that captures the pitch of a note or pitches of a chord. They typically assume that the current state is dependent only on the immediately preceding state. They can be extended to higher-order Markov models that describe the statistical properties of the current state in terms of more than one past states.

Markov models with observable states are called Markov chains. Since their states are observable, their state change probabilities can be calculated directly from the input data samples and represented in a state transition matrix. Markov chains are used in [CC11a; GN21] to generate chords for melody. Their state transition probabilities are calculated using frequency counts of state transitions present in the training set. In [CC11a], Markov chains are combined with music-theoretic rules to model the harmonization process. It relies on only a few training samples that capture a specific style. The chord transition probability in [GN21] is defined based on neo-Riemannian principles [FM81] of music theory and includes a control parameter for the conservativeness of chord types, defined in terms of neo-Riemannian chord distances.

Markov models with unobservable states are called hidden Markov models (HMMs). They capture both state transition probabilities and output probabilities which are typically estimated based on maximum likelihood approach via algorithms such as expectation maximization and Markov chain Monte Carlo. HMMs have been used for several melody harmonization works [KP+23; LJ04; RFV13; SMB08] to model the relationship between a hidden chord sequence and the observed melody sequence in training data. They apply the Viterbi algorithm, a kind of dynamic programming algorithm, to obtain a globally optimal chord sequence from all possible chord sequences in the training data. Also, these HMMs generate pitches for notes with fixed durations and do not generate rhythmic variations for the chords. They

have been used to generate harmonizations in pop and jazz style or a blend of styles [KP+23].

Methods based on Markov chains are simple and efficient whereas methods based on hidden Markov models are more complex. Both types of methods can generate chord variations due to its probabilistic state transition and output production. Low-order Markov models (e.g., order ≤ 3) can represent local statistical properties, which enable them to capture new short-term musical structure such as the relationship between neighbouring pitches.

In theory, high-order Markov models can capture the long-term structure of chord sequences, but their space and time complexities are too large to be practically viable. For example, high-order Markov chains produce high-dimensional transition matrices with many empty entries and they need a lot of training examples to fill the matrix entries. Moreover, there is no guarantee that the same chord progression can be reproduced in repeated sections. For the same reason, they are not practically viable for capturing long-term or formal structure for harmonization. Markov chains can also work with a single training music piece, but in this case they tend to produce state transition matrices with many empty entries.

3.1.5 Direct Optimization

Direct optimization methods [KK23; WC16b; Yi+22] for chord generation produce chord sequences by directly optimizing an objective function using optimization algorithms. Typically, they represent a note and a chord as a discrete item, although continuous numerical representation is possible. For harmonization, they tend to find the best possible set of chords for a given melody by directly optimizing an objective function that measures the quality of the generated chord-melody matching and chord progression. The objective function can include a variety of factors, such as the harmony, rhythm, and timbre of the chords.

Existing works based on direct optimization include [WC16b] and [Yi+22]. The harmonizer in [Yi+22] optimizes the weighted-sum of note-level chord-melody matching and phrase-level chord progression, whereas the harmonizer in [WC16b] optimizes the weighted-sum of chord-melody matching, chord transition probability and emotional valence value. The harmonizer in [KK23] uses a heavily parameterised

scoring function based on expert knowledge to determine the best chord transitions. These methods apply dynamic programming (DP) to solve for the optimal chord sequence that matches the melody best. Unlike deep neural networks and Markov models, their DP algorithms do not require training.

Direct optimization is a flexible method. The objective function can be designed to reflect specific characteristics of the harmonies. For example, the objective function could favor harmonies that are in a specific key, that have a certain level of complexity, or that are similar to a particular style of music.

While direct optimization is a powerful technique for harmonization, existing methods have some limitations. One limitation is that it is difficult to define a single objective function that generalizes well to various music styles or aesthetic preferences. For example, in [Yi+22], minimizing the dissonance between the melody note and the bass note of a chord is suited to pop music that uses basic major and minor triads. This approach may not be as applicable to jazz, which often features a mix of dissonant extended chord—such as seventh and ninth chords—as well as consonant chords. Secondly, a single objective function cannot cater to other features that are important for harmonization, such as cadence and correspondence to melody structure. Finally, the harmonization output from direct optimization is directly dependent on the chord transition model used. Novel chords with other pitch combinations cannot be generated from direct optimization alone.

3.1.6 Summary

Existing chord generation methods can be compared according to five aspects: training required, amount of example music needed, style specificity, cadence and voice leading, and temporal structure (Table 3.1). Chord-melody matching and chord progression are handled by all these methods so they are not included in the table for comparison. *Training* indicates whether or not a method has to be trained with specific training set. Methods that require training include hidden Markov models and deep neural networks. Markov chains, rule-based methods, evolutionary algorithms and direct optimization methods do not require training. Markov chains use chord transition probabilities that are pre-computed from a set of example music. Rule-based methods require manual creation of rules and grammars.

Table 3.1: Comparison of methods for harmonization.

Methods	Training required	Example music	Style specificity	Cadence, Voice leading	Temporal structure
Rules & Grammars	no	0 or more	difficult	no	short-term, long-term, formal
Evolutionary Algorithms	no	many	difficult	no	short-term, long-term
Deep Neural Networks	yes	many	tedious	no	short-term, long-term
Markov Chains	no	1 or more	possible	no	short-term
Hidden Markov Models	yes	1 or more	tedious	no	short-term
Direct optimization	no	1 or more	possible	no	short-term, long-term, formal
Proposed method	no	1 or more	easy	yes	short-term, long-term, formal

Evolutionary algorithms require at least one pre-defined cost function for evaluating chord selection. Existing direct optimization methods apply dynamic programming to match the chords with the melody.

Example music refers to the amount of example music pieces required for harmonization. They do not necessarily refer to training examples, which imply the need for training. Rule-based and grammar-based methods may not require any example music pieces if the rules and grammars are defined based on music theory. Markov models and direct optimization methods need at least one example music piece as a reference for suitable chord-melody matching. Evolutionary algorithms and deep neural networks need many examples. In particular, deep neural networks require large datasets of more than 500 music pieces for effective training.

Style specificity indicates whether or not a method can generate a chord sequence that matches a specific music style. Music style refers to different genres of music such as pop, jazz, hard rock and classical. For rule-based and grammar-based methods, it is difficult to manually create rules and grammars for various different

styles. For evolutionary algorithms, it is difficult, if not impossible, to define cost functions that are specific to a particular style. For Markov chains, it depends on how the chord transition probabilities are computed. It is easy to compute chord transition probabilities based on example music of a specific style. On the other hand, it is difficult to define chord transition probabilities based on neo-Riemannian principles. Thus, it is possible for Markov chain methods to achieve style specificity if the correct approach is adopted. Deep neural networks and hidden Markov models require manual preparation of training examples, which is tedious and time-consuming. To achieve style specificity, they need to train separate versions that each handles a specific style. In principle, direct optimization methods can incorporate reference inputs or constraints in the optimization process to achieve style specificity. Unfortunately, existing direct optimization methods do not adopt any approach that can make them style specific.

Cadence and voice leading denote the ability to guarantee specific chord functions at phrase endings and control over voice leading in a chord sequence. These properties are based on music theory. In principle, all categories of methods, except deep neural networks, rule-based and grammar-based methods, can handle cadence as a hard constraint at phrase ending or section ending. Additionally, they can seek to minimize voice fluctuation in the process of generating chords. Unfortunately, existing methods have not considered cadence and voice leading in harmonization.

Temporal structure describes the ability of a method to produce harmony with regard to short-term, long-term and formal music structure. Most existing methods are able to generate harmony with short-term and long-term structure. Markov models are an exception. To handle long-term structures, they require very large transition matrices that often have too many empty entries, leading to incomplete results. Besides rules, grammars, and direct optimization methods, other methods cannot capture and handle global music structure over the entire music piece. They lack constraints needed to enforce larger structural elements like section boundaries across chord sequences. A notable exception is the method of [Con18] that combines Markov chains with structural rules. This approach allows the method to handle global music structure.

My proposed method of harmonization overcomes the shortcomings of existing methods. It applies multi-objective optimization that does not require training.

Therefore, the costs of cleaning, annotating and training on large datasets are avoided. It achieves style specificity by producing chord sequences that match the style of a selected set of example music. In addition, by applying multi-objective optimization, it can handle voice leading as an additional objective and enforce cadence as a hard constraint. It can also handle long-term and formal structure of the given melody according to pre-defined structure labels.

3.2 Accompaniment Generation

Methods for generating accompaniment patterns can be divided into four categories, namely rule-based methods, evolutionary algorithms, deep neural networks and hybrid methods. Methods involving chordal accompaniment [Niu23] are excluded from this section as it is trivial and does not contain complex textures and rhythm.

There is a related area called automated accompaniment that involves score following [BLB07; GD97; Kle05; OD01; PB05; Rap04; SOS04]. This category of works addresses the playback of accompaniment from an existing score that must be appropriately sped up or slowed down to match the performance of a melody soloist. Although similar techniques are used to match accompaniment to melody, it does not create new accompaniment patterns. Hence, works in automated accompaniment for score following are omitted from this section.

3.2.1 Rule-based and Grammar-based Methods

Methods in this category use rules and grammar to generate accompaniment that consists of single or multiple tracks. These rules and grammar represent the musical features that contribute to the rhythmic, textural or emotional aspects of music accompaniment. They may be manually defined [CLC13; WC16b] or learnt from a training set [Spa99].

The variability and style specificity of the accompaniment generated by these methods is highly dependent on the rules and grammars. Some methods rely on a small set of rules that map music emotion to musical features in existing accompaniment templates to produce emotional accompaniment [CLC13; WC16b]. Other works used extensive sets of rules and grammars based on one target style

such as Bach [Spa99] and pop [SDS10] to achieve multi-track accompaniment.

The advantage of rule-based and grammar-based methods is that accompaniment targeting pop and classical music can be achieved using rules and grammars guided by traditional music theory. However, these methods can neither generalize well to other styles nor allow for personalized styles outside of music theory without any rule violation [WC16a]. It is difficult to create a comprehensive set of rules that cover all possible accompaniment styles. While this shortcoming can be overcome using a probabilistic combination of general and style-specific rules, this approach requires those probabilities to be learnt [Spa99] or the justification of using manually selected probabilities [SDS10].

3.2.2 Evolutionary Algorithms

Evolutionary algorithms for accompaniment generation [Dos05; LT12] rely on one or more fitness functions to generate and evaluate newly generated accompaniment variations to suit an existing melody. The fitness function describes the goodness of fit for accompaniment-melody matching. Similarly to evolutionary algorithms for harmonization, they repeat a cycle of evaluation, selection and reproduction with variation. In the evaluation stage, candidate solutions generated from user-specified examples or a random process are evaluated using a fitness function that captures accompaniment-melody phrase matching. In the selection stage, a new set of candidate accompaniment phrases are probabilistically copied from the old set, in proportional to its fitness. In the reproduction stage, some operators are applied to the set of candidates to generate new candidate accompaniment phrases. This three-stage evolution cycle repeats until the accompaniment with high fitness is obtained.

Evolutionary algorithms for accompaniment generation differ mainly in the choice of music theory rules that inform the fitness function. In [Dos05], the fitness function evaluates the match between drum grooves and a non-melodic phrase to produce a drum accompaniment. In [LT12], three fitness functions are designed to generate the chordal accompaniment, bass part and final chord arrangement. The first fitness function determines chord tone, non-chord tone and rests for the given melody as the main chordal accompaniment. The second fitness function utilizes the chord

information from the chordal accompaniment to generate a matching bass line. The last fitness function determines the basic chord arrangement to be used among four templates, given the chordal accompaniment and bass part.

With existing evolutionary algorithms, accompaniment variations with short-term and long-term structure can be obtained by exploring various combinations of accompaniment phrases that match the given melody. However, there are some shortcomings. Firstly, there is no guarantee that the accompaniment progression is smooth across phrases. Secondly, it is difficult to encourage correspondence between the accompaniment and the formal structure of melody as these fitness functions do not consider music structure. Finally, it is difficult to define an appropriate fitness function that covers a set of music rules that suit various accompaniment styles.

3.2.3 Deep Neural Networks

Deep Neural Networks (DNNs) for generating accompaniment [Hua+17; Hsi+21; LWY18; LY18; Wan+20a; WMX24; Yi+22; Zhu+18; Zhu+20] have gained traction in recent years. Similar to their application in chord generation, the DNNs for accompaniment generation encode notes and chords as numeric values, and generate polyphonic accompaniment output. During training, they learn a highly complex non-linear mapping between melody and one or more accompaniment tracks, using stochastic gradient descent and backpropagation. During the generation phase, features extracted from the input and the features learned from training examples are processed via feedforward neuron activation to generate polyphonic output.

Families of DNNs used for accompaniment generation include Convolutional Neural Networks (CNNs), Generative Adversarial Network (GANs), Transformers and Diffusion models. The CNNs for accompaniment generation applies cross-correlation on polyphonic training examples to generate accompaniments in Bach style [Hua+17]. GANs simultaneously train two neural networks with existing music and random noise to synthesize new music that closely resemble training examples. In [LWY18; LY18], GANs are used to generate polyphonic accompaniment that matches a given lead sheet melody, whereas the GAN in [Zhu+20] generates multi-track accompaniment for melodies targeting pop, classical and jazz styles but require re-training to switch between styles. Transformer networks incorporate a self-attention

mechanism that relate different positions of a single sequence in its encoder and decoder. In [Hsi+21], Transformers are used to generate accompaniment from lead sheet and corresponding polyphonic piano arrangements in an autoregressive manner. [Zhu+18] also use a similar attention mechanism but with significantly fewer neurons to generate multi-instrument accompaniment in pop style. More recently, [WMX24] proposes a hierarchical approach for accompaniment generation using a four-level cascaded diffusion model. This method first captures the global song structure, then generates lead sheets, and finally produces full piano accompaniments that align with the input melody and chords. By conditioning each stage on the previous one, the method ensures coherence between the accompaniment and melody, chords and music structure.

The major strength of DNNs is that the compositional principles of accompaniment such as texture and voice leading need not be explicitly defined to generate polyphonic accompaniment. This is because the relationships between melody and single or multi-track arrangements are captured implicitly as learnt weights during training. With a large training set representing the target accompaniment that belong to a particular genre or similar accompaniment styles, trained DNNs can generate accompaniments variations in the desired style.

DNNs for accompaniment generation have several shortcomings. DNNs require large training sets, which can be expensive to collect and annotate [JYL24]. Due to the complexity of DNNs, they require a large amount of time to train. DNNs generate accompaniment in the style present in their training set. If the training set includes a mixture of musical styles, the results generated by DNNs can become stylistically diluted, lacking a clear stylistic identity. They require re-training or fine-tuning to harmonize in other styles [Zhu+20]. DNNs cannot directly generate structured accompaniment that corresponds to the input melody structure. They require other methods to impose structural constraints [Wan+20a; Yi+22]. Recently, [WMX24] proposed a hierarchical cascaded diffusion model that introduce structure. However, it does not necessarily enforce harmonic coherence throughout the generated output. Moreover, the approach suffers from low interpretability, lacks integration of music-theoretic principles, and is primarily trained on pop music, limiting its ability to generalize to other musical styles.

3.2.4 Markov Models

Only one existing work uses Markov model to generate accompaniment [Mo22]. It consists of two hidden Markov models (HMMs). The first HMM generates the rhythm of accompaniment given an input melody. The second HMM generates a chord sequence given the generated accompaniment rhythm. Finally, the Viterbi algorithm is applied to produce a sequence of accompaniment phrases, based on a set of example accompaniment phrases, that match both the generated rhythm and generated chord sequence

The main advantage of HMMs is that it can generate appropriate output sequences after being trained on large set of training examples. However, HMMs are unable to generate accompaniment with long-term or formal structure, unless the training set contains long-term phrases. In addition, high-order HMMs require large transition matrices that can contain many empty entries. Consequently, there is no guarantee that a complete accompaniment solution can be found.

3.2.5 Hybrid Methods

This category of methods combines deep neural networks (DNN) with other methods to generate accompaniment for melody. The method in [LZ22] consists of a DNN and a HMM. The DNN takes the melody as input and generates a sequence of chords labels. The HMM takes the generated chord labels as inputs and generate accompaniment. In contrast, the method in [Zha+24; ZX21] uses a HMM followed by a DNN. The HMM uses the Viterbi algorithm, a variant of dynamic programming, to produce a globally optimal sequence of chords for the given melody. The DNN is trained on pairs of reference accompaniment and chords. During the generation phrase, the DNN takes a reference accompaniment, reference chords and chords generated by the HMM as input and rearranges the accompaniment to suit the generated chords.

These hybrid methods leverage the strengths of both approaches: HMMs ensure globally consistent and musically plausible chord progressions by explicitly modeling chord transitions over time. DNNs learn complex, high-dimensional mappings from chord and melody inputs to actual accompaniment notes, capturing subtle stylistic nuances such as rhythmic patterns, voicing preferences, and textural variations

directly from the training data without relying on hand-crafted rules. As a result, hybrid methods achieve better harmonic coherence than pure DNN methods and greater stylistic diversity than rule-based methods.

Nevertheless, hybrid methods still have shortcomings. Due to the use of DNNs, the tedious and expensive process of collecting large training sets, and the time-consuming training procedure, cannot be avoided. HMMs also require training although it can be done with smaller training sets.

3.2.6 Summary

Accompaniment generation methods can be characterized according to seven aspects, namely, training required, amount of example music needed, style specificity, section closure and temporal structure (Table 3.2). Chord-melody matching and rhythm matching are handled by all these methods, so they are not included in the table for comparison. *Training* indicates whether or not a method has to be trained with a specific training set. Methods that require training include hidden Markov models and deep neural networks. In contrast, rule-based and grammar-based methods do not require training, but require manual creation of rules and grammar.

Example music refers to the amount of example music pieces required for accompaniment generation. They do not necessarily refer to training examples, which imply the need for training. Rule-based and grammar-based methods may not need any example music piece if they are defined based on music theory. Markov models need at least one example music that encapsulates suitable accompaniment-melody matches. Evolutionary algorithms and DNNs need many examples for accompaniment generation. In particular, DNNs require large datasets of more than 500 music pieces for effective training.

Style specificity refers to the ability of a method to generate accompaniment that matches a specific music style. For rule-based and grammar-based methods, it is difficult to manually create rules and grammars for various accompaniment styles. For evolutionary algorithms, it is difficult, if not impossible, to define cost functions that are specific to a particular accompaniment style. Deep neural networks and hidden Markov models require manual preparation of training examples, which is tedious and time-consuming. To achieve style specificity in accompaniments, they

need to train separate versions that each handles a specific style.

Section closure refers to a method’s ability to generate accompaniment endings that clearly mark the end of a melodic phrase or section. Rule and grammar-based methods can define note durations and harmonic resolutions to achieve this. However, existing evolutionary algorithms, Markov models, and deep neural networks lack direct control over closure that matches formal music structure. As a result, the presence of closure is not guaranteed.

Temporal structure refers to the ability of a method to produce accompaniment with regard to short-term, long-term and formal music structure. Most existing methods can generate accompaniment with short-term and long-term structure. Markov models are an exception. To handle long-term structure, Markov models require very large transition matrices that may contain many empty entries, leading to an incomplete results. Besides rules and grammars, other methods cannot handle formal music structure.

My proposed method for accompaniment generation overcomes the shortcomings of existing methods. It applies direct optimization that does not require training. It achieves style specificity by producing accompaniment that matches the style of a selected set of example music. In addition, it can handle phrase and section closure. Therefore, the costs of cleaning, annotating and training on large datasets are avoided. It can also handle long-term and formal structure of the given melody according to pre-defined structure labels.

3.3 Performance of Deep Neural Networks for Music Generation

Two studies have been conducted to compare the performance of deep neural networks and Markov models in generating monophonic and polyphonic music. They investigate the quality of generated 30-second melodic phrases [Yin+21; Yin+23] and polyphonic music in the style of a classical string quartet [Yin+23].

The study by [Yin+23] indicates that existing deep neural networks show negligible or no improvement in performance compared to Markov models in terms of stylistic conformance to a target music style. For instance, according to [CL17], the stylistic conformance ratings achieved by the MAIA Markov model is comparable

Table 3.2: Comparison of methods for accompaniment generation.

Methods	Training required	Example music	Style specificity	Section closure	Temporal structure
Rules & Grammars	no	0 or more	difficult	yes	short-term, long-term, formal
Evolutionary Algorithms	no	many	difficult	no	short-term, long-term
Deep Neural Networks (DNN)	yes	many	tedious	no	short-term, long-term
Hidden Markov Models (HMM)	yes	1 or more	tedious	no	short-term
DNN + HMM [LZ22]	yes	many	tedious	no	short-term
HMM + DNN [ZX21]	yes	many	tedious	maybe	short-term
Proposed method	no	1 or more	easy	yes	short-term, long-term formal

to those achieved by the state-of-the-art DNNs such as MusicVAE [Rob+18] and Music Transformer [Hua+18]. Consistent with the results of [Hua+18], the study reported significantly higher stylistic conformance ratings for human-composed music compared to auto-generated music.

The authors of [Yin+21] identified a noticeable lack of originality in the music generated by deep neural networks. They define originality as the degree to which an algorithm copies from example music pieces on which it is trained. To assess originality, the authors conducted tests on melodies generated by Music Transformer [Hua+18] and MAIA Markov model [Rob+18]. They also established a baseline as the degree of copying in human-composed melodies with respect to an existing set of melodies. Test results revealed that 2-bar and 4-bar melodies generated by Music Transformer exhibited a lower degree of originality compared to the baseline. In contrast, the melodies produced by MAIA Markov model demonstrated similar levels of originality to those of human composers. Further analysis indicated

that the Music Transformer tended to reproduce exact copies of training phrases, particularly when the network was trained for a long period of time.

Both studies conclude that the performance of existing deep neural networks in terms of stylistic conformance and originality are no better than that of a well-trained Markov model. They advocate caution to the prevalent view of deep learning as the primary approach to automated music generation.

Chapter 4

Harmonization

The goal of this research consists of two parts, namely harmonization and accompaniment generation. This chapter presents a method for harmonization called FlexChord that is completed as the preliminary work of the thesis. In automated music composition, music needs to be represented in a form that facilitates computation. Thus, this chapter begins with the computational representation of music elements.

4.1 Representation of Music Elements

A melody consists of a sequence of notes and rests. A note has a pitch value that is encoded as a MIDI note number ranging from 0 to 127, which corresponds to C1 and G9 respectively. A rest has no pitch value. Nevertheless, for computational convenience, a rest is often represented by the pitch value of 0. This is acceptable because pitch value 0 has very low frequency and is rarely used in music. The melody also includes other attributes, including key in major or minor mode and time signature S .

The relationship between two pitches can be described and measured in several ways. The signed difference between two pitches p and q is called **pitch difference** d_p :

$$d_p(p, q) = q - p. \quad (4.1)$$

With p and q in MIDI note numbers, this difference measures the number of semitones between p and q . The **pitch interval** $\theta(p, q)$ between pitches p and q is defined as

$$\theta(p, q) = |p - q| \bmod 12.$$

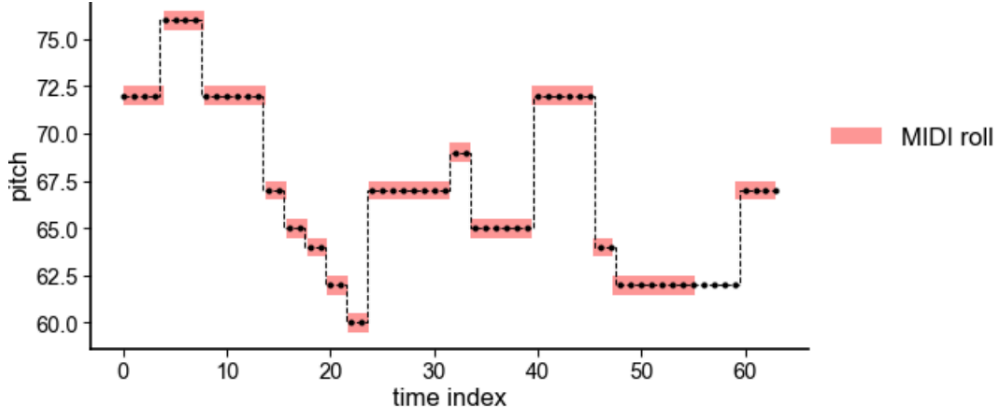


Figure 4.1: Step function of a melody. Sustained pitches are highlighted in red. Horizontal lines that are not highlighted indicate rests.

Two pitches that are separated by exactly one octave share the same pitch class, which is an integer between 1 and 12. Thus, **pitch class** pc of pitch p is computed as

$$pc(p) = p \bmod 12. \quad (4.2)$$

In computation, melody is typically represented as a step function (Fig. 4.1) of m pitches over integer time index $i = 0, \dots, m - 1$. Each pitch p_i at time index i is encoded by a MIDI pitch number. A change of pitch corresponds to a pitch onset. Rests are represented separately outside the step function. The horizontal part of a step function between a pitch onset and an immediately following onset or rest corresponds to the duration of the pitch.

The objective of harmonization is to generate a chord at each **strong beat** to match the pitch sequence and structure of a given melody. Let j denote strong beat index, for $j = 0, \dots, n - 1$. It is related to time index i by

$$j = \text{int}(i/\alpha\beta), \quad \alpha = \begin{cases} 2 & \text{if } S = \{2/4, 4/4\}, \\ 3 & \text{if } S = \{3/4, 6/8\}, \end{cases} \quad (4.3)$$

where β is the number of pitches per unit duration and S is the time signature of the melody. The unit duration is the duration of 1/4 note for melodies with time signature S of 2/4, 3/4 and 4/4, and the duration of 1/8 note for 6/8 melodies.

The number β can be 4, 8, 16 or 32. Therefore, Equation 4.3 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 , $j = 0, \dots, n - 1$. Each fragment h_j contains pitches $p_{\alpha\beta j}, \dots, p_{\alpha\beta(j+1)-1}$. 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 $\lambda_j = j'$.

The objective of harmonization is to produce chords c_i , each consisting of multiple pitches q_{ik} , for $k = 1, \dots, \kappa$, 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 (i.e., kept unchanged) 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.

4.2 Problem Formulation

Harmonization is the process of forming chords to support and enhance a given melody. It needs to fulfill various aspects namely style-specific chord progression, chord-melody matching, cadence, and voice leading. These aspects are formulated as computational objectives and constraints.

4.2.1 Chord Progression and Harmonization Style

In music theory, chord progression is one of the key elements that convey harmonization style [Ran99]. It is noted that chord progression alone may not be sufficient for differentiating music styles such as 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 a reference music, where c' immediately follows c . Then, chord transition probability $P(c' | c)$ can be computed by measuring the frequency of occurrence of chord pair (c, c') in the reference music. These transition probabilities form the chord transition model C that captures the style of the reference music. Therefore, a user can easily specify music style by

Table 4.1: Pitch dissonance depends on pitch interval.

Pitch Interval $\theta(p, q)$	Pitch Dissonance $d(p, q)$	Example: $p = C$
0	0	C
1, 11	1	C#, B
2, 10	0.8	D, A#
3, 9	0.5	D#, A
4, 8	0.4	E, G#
5, 7	0.3	F, G
6	0.7	F#

providing one or more reference music pieces of the desired style. There is no need for the user to provide detailed information that describe the desired style. Given the user’s choice of reference music, it is easy to apply a simple preprocessor to derive the chord transition model from the reference music.

The chords in chord transition model C are represented by Roman numerals (Section 2.1). 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 neural network methods represent each position of the same chord (e.g., C-E-G, E-G-C) as a different chord. [Yi+22] also uses Roman numerals to represent chords, but it limits its chord types to major and minor triads. In contrast, this research work uses all chord types.

4.2.2 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 [Hur94; TDSG18]. According to Huron’s study [Hur94], the dissonance value $d(p, q)$ between two pitches p and q depends on their pitch interval $\theta(p, q) = |p - q| \bmod 12$ (Table 4.1). 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 their pitches:

$$d(h_j, c_j) = \frac{1}{|h_j||c_j|} \sum_{p_i \in h_j} \sum_{q_{jk} \in c_j} d(p_i, q_{jk}), \text{ for } p_i \neq \text{rest}, \quad (4.4)$$

where $|h_j|$ and $|c_j|$ are the total number of pitches in melody fragment h_j and chord c_j respectively. [Yi+22] uses the same concept but with slightly different dissonance

values.

4.2.3 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 [Ran99]. Different cadences create different senses of resolution. They help to clarify the overall hierarchical structure of music [Tem11].

In practice, it remains difficult to define a fixed set of cadences across multiple music styles [Cap04]. A simplified approach is to refer to cadence as specific chord-pairs at the end of a phrase or a section. Each chord pair (c, c') in chord transition model C is given a structure label that takes the value ‘P’, ‘S’ or ‘ ’, as defined for melody fragments h_j (Section 4.1). Structure labels are provided by user with help of a melody structure analysis tool [DZD20]. Then, cadence can be handled by matching melody fragments labeled ‘P’ and ‘S’ to chord pairs with the same labels (see Section 4.2.5 for details).

4.2.4 Voice Leading

Voice leading refers to the interaction of individual voices that creates harmony [Hur16]. 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 [Hur01; Hur16]. 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

$$\delta_i = \sum_{k=1}^{\kappa} |q_{ik} - q_{i-1,k}| \quad (4.5)$$

4.2.5 Constraint and Optimization Objectives

The conditions of harmonization discussed in the previous sections are translated into two hard constraints of cadence and phrase repetition, and three optimization objectives:

1. Chord progression and style matching:

Harmonization style is captured in the chord transition model C . By maximizing the chord transition probability $P(c_j | c_{j-1})$ of the selected chords c_j , for $j = 1, \dots, n - 1$, the generated chords c_j follows the style of C . Note that chord c_0 has no previous chord.

2. Chord-melody matching:

By minimizing overall chord-melody dissonance D :

$$D = \sum_{j=0}^{n-1} d(h_j, c_j), \quad (4.6)$$

the generated chords will match the melody well in terms of harmonization.

3. Smooth voice leading:

Minimizing overall voice fluctuation Δ produces smooth chord voices:

$$\Delta = \sum_{i=1}^{m-1} \delta_i. \quad (4.7)$$

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.

4.3 Multi-Objective Optimization

4.3.1 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 [Ehr05; Mie99]. This family of methods generalizes the optimality condition of single-objective optimization to the condition of **Pareto optimality**. A Pareto optimal solution is a solution such that no objective function can be further improved without compromising other objective functions. 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 deep neural networks. Despite its simplicity, it has an inherent shortcoming—it cannot produce some Pareto optimal solutions when the Pareto front is non-convex [Ehr05; Mie99]. 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 [Ehr05; Mie99]. 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 have been 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 never minimized—they are effectively ignored. In addition, lexicographic methods can only produce a very small subset of solutions on the Pareto front.

The lexicographic method can be slightly modified to overcome its shortcomings [Mie99]. In each step, instead of looking for solutions that minimize an objective function, it looks for solutions that are good enough, i.e., solutions that are close to and including the minimum of the objective function. 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 minima of the first few objective functions and nearer to those of the last few objective functions. Therefore, by varying the bounds, the modified lexicographic method can produce all solutions on the Pareto front [Leo24].

In the harmonization problem formulated in Section 4.2, time is discretized into integer index i . Strong beat index j is an integer and a chord is a discrete item. In other words, the solution space of harmonization is discrete. Therefore, the algorithm for solving the harmonization problem has to be a discrete optimization algorithm. Standard constrained optimization methods for the continuous domain

are not applicable.

4.3.2 FlexChord Harmonization

FlexChord (Algorithm 1) adopts a discrete version of the 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 transition model C having a high transition probability and 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 labeled ‘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). These steps 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. The actual chord is determined by the chord’s Roman numeral and the melody’s key. For example, chord I is C major chord for C major key, G major chord for G major key, etc.

If λ_j is an integer j' , it means that fragment h_j repeats $h_{j'}$ exactly. So, chord $c_{j'}$ is copied to chord c_j (Line 11–12). This method generates the same chord sequence for repeated phrases and takes care of the long-term structure of repeating phrases. An alternative to this approach is to begin each phrase in the same manner as the beginning of melody (Lines 2–5). Then, being deterministic, FlexChord will generate the same chord sequence for the repeating phrase. This approach sacrifices the voice leading at the beginning of the earlier repeated phrase.

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 $K = N$, it ignores Objective 1. A good balance between Objectives 1 and 2 is found with $1 < K < N$. This balancing of Objectives 1 and 2 are also applied to the first chord at $j = 0$ (Lines 3–4). A chord selected in Line 15 can have several chord positions (Section 2.1). 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. Therefore, FlexChord performs modified lexicographic optimization of 3 objective functions and 1 constraint.

Note that the chord transition model C can be prepared from a large set of reference music pieces (Section 4.4.1) or a single reference music piece (Section 4.4.5). A small reference set may contain chords without pairing to other chords. Such chord-pairs are assigned 0 transition probability in the chord transition model. Thus, the chord transition model contains all possible pairings of chords for each chord type present in the reference set. Chord-pairs with 0 transition probability may be selected if it is among the top K pairs with the highest transition probability. This ensures that the selection of top K chords in Line 3 and 14 is always successful.

Algorithm 1: FlexChord Harmonization

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

- 1: **for** $j = 0, 1, \dots, n - 1$ **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** $\lambda_{j+1} = 'P' \text{ or } 'S'$ **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** $\lambda_j = 'P' \text{ or } 'S'$ **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 $P(c | c_{j-1})$.
- 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 $c_i = c_{i-1}, i = \alpha\beta j + 1, \dots, \alpha\beta(j + 1) - 1$.
- 18: Set chord onset for chord c_j .

4.3.3 Analysis of Time Complexity

FlexChord runs for exactly n iterations, one for each strong beat index j , to produce n chords. In each iteration, lines 3, 4, 14 and 15 perform discrete search for appropriate chords. Discrete search can be implemented in several ways with different time complexity. Linear search has a time complexity of $O(N)$, where N is the number of chords in the chord transition model C . Indexed lists of chords sorted in decreasing order of transition probability has time complexity of $O(1)$. Lines 3 and 14 select K chords among all chord types in the chord transition model C , and Lines 4 and 15 select one chord among the K chords. Since K is a small constant parameter, it has no significant effect on time complexity. Line 7 selects a chord pair with appropriate structure label. The number of chord pairs with structure labels is small compared to all possible pairings of chords. It is estimated to be of order $O(N)$. Therefore, the time complexity of FlexChord is either $O(nN)$ or $O(n)$ depending on the implementation of discrete search.

The number of iterations n depends on the length of the melody. In the experiments (Section 4.4), the value of n ranges from 16 to 124. The number of chords N in the chord transition models used in the test is less than 100 (Table 4.2). Therefore, the actual execution time of FlexChord is very short.

4.4 Experiments and Discussions

This section presents two types of tests: short-phrase test and full-length test. Short-phrase test is used for quantitative evaluation of FlexChord’s performance compared to two existing deep neural network methods (Section 4.4.2). Both short-phrase test and full-length test are used for musicological evaluation.

4.4.1 Data Preparation

Two public datasets, namely Chord Melody Dataset (CMD) [Hie19] and Hook Lead Sheet Dataset (HLSD) [And21] were adapted for the harmonization task. The original datasets contain a mixture of pop, jazz, classical and new age music. Algorithms that are trained on these datasets could produce arbitrary results that reflect a mixture of genres as opposed to a specific target genre. To avoid such confusion, two large sets of songs of distinct genres, namely pop and jazz, were

extracted from the combined pool of CMD and HLSD resulting in 1000 pop songs and 266 jazz songs respectively. This selection of songs catered to a substantial number of training examples for a fair comparison with recent deep neural networks. A smaller set of 30 pop songs randomly selected from the pool was also prepared in order to assess if there were any performance differences between the use of small and large pop datasets.

Jazz songs were selected to contrast with pop songs due to their greater chord and harmonic complexity. They contained extended chords and frequent key changes. In contrast, pop harmonies consisted of triads or basic seventh chords and were less adventurous with chord progressions.

The deep neural networks that were used for comparison worked only on songs with 4/4 time signature. Thus, only such songs were selected for testing. Similar to existing work, their tempo was set to 120 beats per minute and they were transposed to all 12 music keys to ensure that the neural networks could handle test melodies of various keys. Each song was encoded in step-function representation with number of pitches per unit duration β set to 16. A chord was sampled every half bar over 8 bars of music.

Songs for each dataset were split into 8-bar phrases. 8-bar phrases were used because existing neural networks were trained and evaluated on 8-bar phrases. With 8-bar phrases, deep neural networks could be trained with shorter training time and facilitate easier quantitative evaluation.

Each dataset was split into training, validation (i.e., fine tuning) 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 $\lambda_j = \text{'P'}$ for $j = 7, 15$ and $\lambda_j = \text{' '}$ otherwise. Table 4.2 summarizes the statistics of the prepared datasets.

FlexChord does not require training. The training set was used to build the chord transition model of FlexChord and the testing set was used for performance testing. The validation set was not used by FlexChord. For the chord transition model, all chords were represented by Roman numerals. The Roman numerals were captured at the root position, ignoring the actual inversions. Chord transition probabilities

Table 4.2: Summary statistics of the datasets used for harmonization.

Dataset	Number of Songs	Number of 8-bar phrases	Number of unique chord types
Pop-S	30	2867	43
Pop-L	1000	81,625	70
Jazz-L	266	17,969	71

were computed from the frequency of occurrence of consecutive chord-pairs in the training set. Structure labels were assigned to the chord-pairs as described in the previous paragraph.

4.4.2 Quantitative Evaluation

(a) Test Procedure

The objective of this test is to investigate the performance of FlexChord (FC). For comparison, 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 deep neural networks, namely **VTHarm** by Rhyu *et al.* [Rhy+22] and **DAT-VAE** by Zhao *et al.* [ZXW22]. Both neural networks have different architectures but use similar loss functions for harmonization. They were trained on the same datasets (Section 4.4.1) with published training parameters [Rhy+22; ZXW22].

VTHarm is a Transformer-based neural network that minimizes chord reproduction error. It also minimizes the Kullback–Leibler (KL) divergence between the learned distribution and target Gaussian distribution of chords. Like the standard Transformer, it has a self-attention mechanism that functions as a weighting scheme for capturing chord similarity at different time points within the sequence. It also has an additional encoder that encodes melody and chords, along with the melody’s key signature.

DAT-VAE encodes the melody and chords with a variational autoencoder network (VAE). The VAE aims to minimize a loss function that consists of 2 terms, namely chord reproduction loss and KL divergence between the learned distribution

and target Gaussian distribution of chords. Additionally, it has a preprocessor neural network (DAT) that is trained to map all 12 transpositions of melody to the melody of a canonical key. DAT’s goal is to normalize the key of the melodies so that the whole DAT-VAE can learn to generate suitable chord progressions regardless of the actual key of the melody.

The training objective of VTHarm and DAT-VAE focuses on reproducing training chords and modeling the Gaussian distribution of chords. They do not explicitly maximize chord transition probability and minimize dissonance between the generated chords and the melody. In addition, they use different training inputs and architectures to handle melodies in different keys. VTHarm is specifically trained with melody-chord pairs along with the melody key signatures. It also integrates a self-attention mechanism that considers chord similarity with other chords in the sequence. In contrast, DAT-VAE does not require melody key signatures at training. Instead, its DAT network normalizes the input melodies to a canonical key.

Both VTHarm and DAT-VAE were trained on the training set, fine tuned using the validation set, and then tested on the testing set. FC and FC— used the chord transition model derived from the training set (Section 4.4.1). They were tested on the testing set without requiring training and fine tuning. 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.

The experiments were conducted on a MacBook Pro equipped with a quad-core Intel Core i7 processor at 2.2 GHz and 16GB of RAM. The average execution time per test song was 0.09 s, 0.34 s and 0.35 s for FlexChord, VTHarm and DAT-VAE respectively.

(b) Quantitative Test Metrics

Note that a melody can be harmonized with different chord sequences to produce equally good results. There is no one chord sequence that is better than the others. Therefore, there is no ground truth for performance evaluation.

For objective comparison between FlexChord and neural networks, a set of simple but musically meaningful metrics were identified:

- Voice Fluctuation V

V is the average voice fluctuation in the chord sequence (Eq. 4.5). A low V value means that voices in successive chords have lower pitch difference. This enhances the smoothness of chord progression. The value of V ranges from 0 for no difference to κ , which depends on the actual pitch difference.

- 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 within a chord sequence. The presence of chord inversions create movement and they encourage lower voice fluctuation V . The value of IC ranges from 0 to 1.

- Chord histogram entropy H

H is the amount of entropy (or disorder) in a chord histogram. It is defined as:

$$H = \sum_k^N p_k \log p_k, \quad (4.8)$$

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 and the order of chords produced. A higher H indicates a wider variety of chords used. The value of H ranges from 0 to 2 or more.

- Dissonance D

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

$$D = \frac{1}{n} \sum_j d(h_j, c_j), \quad (4.9)$$

where $d(h_j, c_j)$ denotes the average dissonance between pitches of melodic fragment h_j and chord c_j (Eq. 4.4). Using dissonant chords around a specific melody note can draw attention to it, adding richness to the overall harmony. The lower the D the more harmonious is the match between the chord and the melody. Nevertheless, the appropriate level of dissonance depends on the target music style. The value of D ranges from 0 to 1.

- Chord pitch ratio R0

This metric counts the ratio of chord pitches n_c to non-chord pitches n_n in the melody fragment h_j in relation to its assigned chord c_j :

$$R0 = \frac{n_c}{n_c + n_n}. \quad (4.10)$$

This metric measures the harmony between chord and melody. The higher the R0, the better is the match between the chord and melody. The value of R0 ranges from 0 to 1.

- Chord pitch ratio R1

R1 is similar to R0, except that proper non-chord pitches are counted as chord pitches. This metric is usually called chord tone to non-chord tone ratio in the literature [Yeh+21]. A proper non-chord pitch is a non-chord pitch p_i that is within two semitones of its subsequent pitch p_{i+1} .

$$R1 = \frac{n_c + n_p}{n_c + n_n}, \quad (4.11)$$

where n_p is the number of proper non-chord pitches. The higher the R1, the better is the match between the chord and melody. The value of R1 ranges from 0 to 1.

Among these metrics, V, IC and H measure chord diversity, while D, R0 and R1 measure chord-melody match. In particular, H and R1 are part of the canonical metrics [Yeh+21] used in existing DNN-based harmonization works [Rhy+22]. The remaining metrics in [Yeh+21] are omitted because they are related to H and R1. Metric R1 uses the concept of proper non-chord pitch that is not defined in music theory. Therefore, the alternative R0 is included along with R1. The additional metrics V, IC and D are included to contrast the performance of FlexChord and DNNs.

(c) Quantitative Comparison of Various Methods

Table 4.3 tabulates the quantitative results of all the methods. FC– achieves the lowest voice fluctuation V indicating that it produces chords with the smoothest voices. This is because it directly minimizes voice fluctuation (Lines 8, 16). On the other hand, FC has a slightly higher V than does FC– because it enforces the cadence constraint that fixes chords in root position at phrase or section endings (Lines 9–10). Unlike the FC variants, VTHarm and DAT-VAE do not have V scores because they do not produce chords with actual pitch values. Instead, VTHarm produces chord labels in root position while DAT-VAE produces a set of pitch classes without chord position information.

Table 4.3: Quantitative evaluation for chord diversity and chord-melody match.

		Chord diversity			Chord-melody match		
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	1.24
	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

FC– consistently achieves the highest number of inverted chords IC because it inverts chord positions to minimize voice fluctuation (Line 8, 16). Unlike FC–, FC needs to satisfy the cadence constraint that fixes chords in root position at phrase or section endings (Lines 9–10). Hence, its IC scores are lower than those of FC–. Since VTHarm and DAT-VAE do not produce chords with chord position information, they do not have IC scores.

FC– consistently achieves low chord histogram entropy H for all test sets, indicating its narrow range of chords used. It chooses the most frequently used chords by maximizing chord transition probability (Lines 3, 14), thus restricting its choices of chords. In contrast, by catering to cadence, FC can select chords that do not necessarily have the largest transition probabilities. Thus, it has a slightly higher H compared to FC–. Unlike the FC variants, both DAT-VAE and VTHarm do not explicitly maximize chord transition probability. Instead, they minimize chord reproduction error and KL divergence between the learned and target distributions. Their H scores appear to depend on the training and testing sets. In particular, VTHarm achieves the lowest H score for Pop-S but the highest H score for Pop-L and Jazz-L. On the other hand, DAT-VAE’s H score is similar to that of FC– for Pop-S and Pop-L, but similar to that of VTHarm for Jazz-L.

FC and FC– have very similar dissonance D scores. Both FC and FC– 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 does not need to handle chord transition probability. By minimizing chord reproduction error and KL divergence of chord distributions, it could choose chords with the lowest dissonance. In contrast, VTHarm has a self-attention mechanism that allows it to choose less frequently-used chords. Being less frequently-used, these chords tend to have large dissonance. Therefore, VTHarm has the largest dissonance D compared to all other methods.

Chord pitch ratio R0 is a measure that is negatively correlated to dissonance D because more chord tones leads to lower dissonance. The FC variants have similar large R0 for their small D. In contrast, the R0 scores of VTHarm and DAT-VAE seem to be positively, rather than negatively, correlated to their D scores, which is counter-intuitive. In particular, DAT-VAE has the lowest D and lowest R0 among all methods, whereas VTHarm has the highest D and relatively high R0.

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

Overall, FC achieves the best performance compared to other methods. It caters to cadence by sacrificing chord diversity slightly (slightly small H). It chooses chords with smooth voices (small V) and relatively low dissonance (small D), indicating good chord-melody matching. Although deep neural networks may excel in an individual metric, they do not handle cadence and voice fluctuation. Moreover, their D and R0 scores have a relationship that is opposite to the expected negative correlation. There is noticeable difference when FC is applied with Pop-S compared to Pop-L, but the difference is not significant. This indicates that FC’s performance is stable even with a small reference set.

4.4.3 Musicological Evaluation

(a) Musicological Evaluation of Various Methods

Figure 4.2 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

Figure 4.2: Chords generated by various methods for *Colors of the Wind*.

generate chord pitches without information of exact chord positions. For display purposes, their chord labels are inferred from the generated pitches and their chords are shown in root position. The human-composed (original) chords of *Colors of the Wind* are included to serve as a reference for evaluation.

The figure 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 foundation 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.

Yet, FC and FC– remain creative in how the same chords are differently applied to phrase and section endings indicated by 'P' and 'S'. For example, both FC and FC– reproduce the original Dm chord in bar 8. However, in bar 4, FC uses Dm, while FC– uses F to conclude the phrase. Although both FC and FC– do not use the original Am chord in bar 4, their chord choice remains harmonious with the melody.

Figures 4.2 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. Its bass notes descend relatively smoothly. 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 chord transition, 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 found in the original chord sequence. In bars 1 and 2, the chords change from B♭maj7 to Gm to B♭, resulting in a harmonic shift that may sound unusual compared. Moreover, the tonic F chord found in bar 1 of the original chord sequence is absent in the chord sequence of VTHarm. In fact, without the melody, it is difficult to determine the key of VTHarm’s chord progression.

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. The Csus4 chord introduces tension and creates a sense of expectation for resolution. In the Csus4 – G7 – C progression, although Csus4 resolves to C, G7 introduces an additional layer of complexity and shifts the harmonic focus away from the melody key. Similarly, in the Csus4 – D progression, Csus4 creates a suspension that disrupts the smooth flow of the progression and leads to an unexpected harmonic shift when it moves to D.

These examples show that the harmonization results of VTHarm and DAT-VAE can diverge from the intended pop style and melody key. This could be due to their training on a diverse collection of pop music, diluting the style specificity of the generated chord sequence. In other words, their results may not be harmonically coherent. This disparity underscores the complexity of capturing the nuances of pop harmonization.

4.4.4 Listener Study

A listener study was conducted to compare the perceived harmonization quality of various methods. 27 participants aged 18 to 65 years (average age = 39.2 years, SD = 13.2 years) were recruited via CloudResearch. All participants reported

having at least five years of music training and being able to play the piano. Each participant was presented with two 8-bar songs with chord accompaniment. One of the pair was either the human-composed harmonization (HC) or FlexChord’s result. The other was a result of either VTHarm or DAT-VAE. The participant was asked to pick the one that was more harmonious. In addition, the participant was asked to rate how harmonious each song was on a 5-point Likert scale. The ratings were used to verify whether the participant’s choice was consistent with the ratings. To reduce listener’s fatigue, each participant repeated the above procedure for only 40 pairs of songs.

As discussed in Section 4.4.2, there is no ground truth in harmonization. A HC chord sequence does not serve as a ground truth but is used to verify whether the participants’ evaluation are reliable. For example, if the participants judge HC chords to be less harmonious than a DNN result, it does not mean that HC chords are less harmonious than DNN chords in reality. Instead, it means that the participants do not understand harmonization well and therefore, the participants’ evaluation is unreliable. On the other hand, if the participants judge HC chords to be more harmonious than the DNNs results, it means that the participants’ evaluation is reliable. Therefore, FlexChord is not compared with HC chords, which helps to reduce listener’s fatigue.

The participants’ assessments were grouped into one of the following cases:

- H+: Human-composed harmonization is more harmonious than DNN’s result.
- H−: Human-composed 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 4.4 tabulates the participants’ assessments. A large majority of participants judge the human-composed harmonizations as more harmonious than those of VTHarm and DAT-VAE (68.3% and 74.7% respectively). This expected result indicates that the participants understand and appreciate harmonization. Among them, DAT-VAE’s results are the least harmonious. Similarly, most participants judge FlexChord’s harmonizations to be more harmonious than those of VTHarm

Table 4.4: Results of the listener study on harmonization.

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

and DAT-VAE (60.3% and 60.9% respectively), with DAT-VAE results being the least harmonious. In addition, more participants judge both the human-composed and FlexChord’s harmonizations to be more harmonious than those of the DNNs (47.3%, 50.6%), compared to the other three cases. The results of this listener study show that FlexChord’s harmonizations are better than those of the DNNs, and its quality compared to that of DNNs is close to the quality of human-composed harmonization compared to DNNs’ results.

4.4.5 Flexible Harmonization Style

(a) Harmonization with Various Styles

FlexChord (FC) can generate various harmonization styles for the same melody by using chord transition models of the corresponding styles. To test FC’s flexibility beyond pop style, the alternative style should be distinct enough to highlight differences in chord types and demonstrate style-specific characteristics. One such style is jazz. Jazz harmony is characterized by more chromatic movement, where notes often shift by small intervals. It frequently incorporates extended and altered chords such as ninths, elevenths, and thirteenths, creating richer textures and more tension. In contrast, pop music typically relies on simpler triads and conventional progressions, favoring clear cadences and predictable resolutions.

To evaluate FC’s ability to generate quality jazz harmonizations without training, a test using jazz melody was conducted using a chord transition model derived from

Figure 4.3: Human-composed and FC jazz chords for *Israel*.

Jazz-L examples. It involved generating a jazz harmonization for the jazz song *Israel*. Figure 4.3 illustrates the jazz chords generated by FlexChord for bars 17–24 of the song. It shows how FC’s chord progression includes chromatic movements that add harmonic richness and tension. It also employs the classic ii7–V–I(maj7) progression (C#m7–F#–B(maj7) in bars 18–19, 22–24), which is a hallmark of jazz. This progression creates a strong sense of anticipation and resolution, with the ii7 chord leading smoothly to the V chord before resolving to the I chord. Moreover, FC introduces chromatic transitions to chords like VII, vii and vi (G#, G#m and A#m in bars 19–21) or their seventh extensions. These transitions, achieved by minimizing voice fluctuation, demonstrate FC’s ability to produce style-specific jazz harmonies while maintaining harmonic coherence with a jazz melody.

Section 4.4.3 presents the harmonization of the first 8 bars of *Colors of the Wind* in pop style. FlexChord is also able to harmonize the same melody in various styles. Figure 4.4 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. It shows how FlexChord adapts its output to reflect each style’s unique harmonic characteristics. The jazz version enhances the pop version (Figure 4.2) with seventh chord extensions, such as ii7, II7, iii7 and vi7 (Gm7, G7, Dm7, Am7). It also features chromatic voice leading in the transitions between chords. For example in bar 2 to 3, the chord progression moves from G7/F to Gm/F and then Am7/E. This sequence

Figure 4.4: Harmonization of *Colors of the Wind* in various styles.

Human:	F	∕	Fmaj7	∕	F6	∕	Gm7	C7
FC (pop):	F	Dm/F	C/E	F	Dm/F	F	Bb/F	F
FC (jazz):	F	∕	D7	Am7/E	Gm7/F	Am7/E	Gm7/F	C7
FC (Greek):	F	∕	D7	BbMaj7/D	Gm/D	Am/E	BbMaj7/F	F

Human:	Gm7	∕	C7	∕	Gm7	C7	Fdim7	C7
FC (pop):	C/E	Bb/D	C	∕	Gm/Bb	∕	Bb	F
FC (jazz):	C7	Gm7/D	Am7/C	∕	Gm7/D	C7	Gm7/D	C7
FC (Greek):	Bb/F	∕	Am/E	∕	Bb/F	C7/E	BbMaj7/F	F

Figure 4.5: Harmonization of *La Vie en Rose* in various styles.

creates smooth chromatic movement in the form of small pitch change of F to E in the bass and B to B $\flat$ to A in the inner voice. In contrast, the Constantinidis version generally avoids the dominant C chord except for bars 5 and 7 which are not phrase or section endings. This creates a slightly unresolved tension typical of modal music.

Figure 4.4 presents harmonizations of the first 8 bars of *La Vie en Rose* using three distinct chord transition models, namely pop, jazz, and a folk-based model derived from the songs of the Greek composer Constantinidis. All three versions begin with the same chord and their subsequent chords reflect chord transitions specific to their models. The human-composed chord sequence includes a mix of basic triads and extended chords (e.g., sevenths, ninths). In contrast, the pop version contains mostly simple triads and their inversions, yielding harmonizations that are more direct and repetitive in structure. The jazz version introduces more complex chords, often with added sevenths or altered tones, leading to denser and more chromatically varied sequences. In contrast, the Greek version tends to repeat bass notes and uses non-standard chords like B $\flat$ Maj7 in bars 2, 4, and 8. It creates a

Figure 4.6: Spooky chords for *Hall of the Mountain King*.

slightly unresolved tension and voice leading typical of modal music.

(b) Custom Harmonization Style

FlexChord can generate harmonizations in a custom style using as little as a single example music piece. This is achieved by creating a customized chord transition model, where chord transitions and probabilities are manually defined based on the example piece. This approach enables FlexChord to produce custom style harmonizations without the need for additional training.

To test FlexChord’s ability to harmonize in a 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 4.6 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 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.

To further demonstrate that FlexChord can create different emotional styles from only one reference piece, two additional styles were constructed, a happy style and a sad style. The happy style was based on a short example that mainly used major chords, while the sad style was created from an example that featured more minor chords. Custom transition models built from these examples were used by FlexChord to harmonize the first four bars of Ode to Joy. As shown in Figure 4.7,

The figure shows a musical score in 4/4 time. The top staff is the Melody in treble clef. Below it are two accompaniment staves in bass clef. The first accompaniment is labeled 'FC (Happy)' and the second is 'FC (Sad)'. Chords are written below each staff. The Happy version uses major and dominant chords (C, G/B, F/A, G, Dm/A), while the Sad version uses minor and dominant chords (Am, Dm/A, E7/G#, E7).

Measure	Melody	FC (Happy)	FC (Sad)
1	C4	C	Am
2	D4	G/B	Am
3	E4	F/A	Dm/A
4	F4	G/B	Dm/A
5	G4	F/A	Am
6	A4	Dm/A	Dm/A
7	B4	F/A	E7/G#
8	C5	G	E7

Figure 4.7: Happy and sad sounding accompaniments by FF.

the happy version includes major chords such as C, G, and F, which give the melody a bright and cheerful character. In contrast, the sad sound uses minor chords like Am and Dm, producing a more somber and darker sound. These results show that FlexChord can use a single reference piece to generate harmonizations with clearly different emotional qualities.

4.4.6 Full-Length Test

(a) Test Procedure

The objective of this test is to assess FlexChord’s ability to generate harmonization across extended musical structures. Both FC and FC– were executed on the full song of *Colors of the Wind* that includes diverse melodic variations and intervals. For comparison, VTHarm and DAT-VAE were executed on consecutive 8-bar phrases of *Colors of the Wind* after being trained on the Pop-L training set.

(b) Musicological Analysis and Discussions

Figure 4.8 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. 4.2). With the exception of bars 10, 15 and 16, the melody of section **A** is repeated exactly in section **B**. For FC, the chords of **A** are also repeated exactly in **B**, due to the enforcement of repetition constraint (Algorithm 1, Lines 11, 12). Both section **A** and **B** constitute the verse of the song. Slight variations of this verse can also be found in sections **D**, **E** and **G**.

The chord transitions to section **C** (bars 16 to 17) of FC and FC– create a relevant major-major contrast from chord F to Am. Both chords share two pitches,

1 **A** 2 3 4 **P'** 8 **S'**

VTHarm: B $\flat$ Maj7 / Gm B $\flat$ / F C / Dm / Dm /

DAT-VAE: F / Dm / F / Dm / / /

FC-: F C/E / Gm/D / F/C C / Dm F/C / ... / Dm /

FC: F C/E / Gm/D / F/C C / Dm / Dm /

9 **B** '0' '1' '2' 16 **S'** **C** 17 **P'** 18

VTHarm: / / C / / F / B $\flat$ / Am /

DAT-VAE: F / Dm / / Dm / / / Gm B $\flat$ add#4 /

FC-: F/C C / Gm/B $\flat$ F/A / ... / F/A / Am Dm7 / / Gm7/D /

FC: F C/E / Gm/D F/A / / F/A F / Am Dm7 / F Dm /

19 **D** 26 **S'** 27 28 **'0'** '1' '2' '3' 43 **S'**

VTHarm: Asus4 / / F / B $\flat$ / F C / / / F /

DAT-VAE: Gm7 F / / Dm Am / Csus#4 F / Dm F / Dm / / F/A /

FC-: F/C / / F/C / / C / F/C C / Gm/B $\flat$ / / F /

FC: F/C / / F / / C/E / F C/E / Gm/D / / B $\flat$ /

44 **F** **P'** 45 46 47 **S'** '0' '1' 48 61 **'28'** '29' 62 **S'**

VTHarm: F Am / Dm / Gm / B $\flat$ / F / / G7 C7 / F /

DAT-VAE: F+ E $\flat$ / F / E $\flat$ + Gm / F C / F CMaj7 / / G7 / / F /

FC-: C / / B $\flat$ /D / C Gm/B $\flat$ / F/A C / F/C C / ... / C B $\flat$ /D / F/C /

FC: C/E / F B $\flat$ / C Gm/B $\flat$ / Dm/A Gm / F C/E / / C B $\flat$ /D / F /

Figure 4.8: Test results of the full song of *Colors of the Wind*.

which are A and C. VTHarm transitions from chord to chord B $\flat$, resulting in a subtler major-to-major contrast. Both chords also share two pitches, which are D and F. However, DAT-VAE does not create contrast when transitioning to **C**. Instead, it produces repeated chords of Dm–Dm.

Section **C** also 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 chord transition B♭–Am instead to build and release tension.

In contrast, the results of FC– do not follow the same trend. While FC– builds tension to the climax using the same chords as FC in bar 17, it does not release tension due to its lack of handling cadence in bar 18. At bar 18, it repeats the Dm7 chord and transitions to another seventh chord (Gm7). 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 dissonance with the melody.

The chord transitions to F (bars 43–44) of FC and FC– are B♭–C and F–C respectively, producing a clear major-to-major contrast. In contrast, VTHarm does not feature any chord contrast in the transition from F to F. DAT-VAE generates an F to 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.

Colors of the Wind ends in bar 62 with the F note. Interestingly, all methods generate the F chord or a chord rooted in F. In particular, FC, FC– and VTHarm generate an F to F chord transition. On the other hand, DAT-VAE generates the chord G7 which contains the pitch F, followed by chord F. This shows a general agreement among the methods that F is an important note to include in the chord.

4.4.7 Summary

Quantitative results illustrate the strengths of FlexChord for short phrases. It directly minimizes voice fluctuation to produce chord progression with smooth voice leading (small V). It excels in generating a large number of inverted chords (large IC) and caters to cadence by sacrificing chord diversity (small H) slightly. It chooses chords with relatively low dissonance (small D) and achieves a favorable ratio of chord tones to non-chord tones (large R0), indicating good chord-melody matching.

Musicological evaluation shows that FlexChord achieves style specificity by producing chord sequence that matches the style of a selected set of example music. It produces a pop chord sequence that is similar to the human-composed chord

sequence, but remains creative in how the same chords are used as cadences to mark phrase and section endings. Although FlexChord does not always reproduce the original cadence, the alternate cadence that FlexChord selects remains harmonious with the melody.

The listener study results reveal that most participants judge FlexChord’s harmonizations to be more harmonious than DNN harmonizations. In addition, more participants judge both the human-composed and FlexChord’s harmonizations to be more harmonious than those of the DNNs, compared to the other cases. This indicates that FlexChord’s harmonizations are better than those of the DNNs, and are close to those of the human-composed harmonizations.

Additional tests results show that FlexChord is flexible at various style levels. Custom or specific styles are explored using a chord transition model of the required style, without the need for training on large datasets. Tests using a jazz chord transition model show that FlexChord can generate high-quality jazz chord sequences that maintain harmonic coherence with the melody. In a separate test, FlexChord uses a custom chord transition model derived from a single example piece. It successfully generates a custom style harmonization that incorporates more dissonant chords than the original, effectively conveying tension.

Full-length melody test shows that FlexChord can produce harmonically coherent chord progressions that match a full-length melody, complete with global music structure. Chord repetitions and key-relevant chords are used to ensure consistency among chords and effective matching with the melody. In addition, FlexChord generates cadences that result in the tension and resolution needed for contrasting sections. Overall, FlexChord shows promise as a flexible harmonizer that can create pop and jazz chord sequences for complete songs.

Chapter 5

Accompaniment Generation

Accompaniment is the auxiliary part of composition designed to harmonically and rhythmically support the melody in music. A critical aspect of generating a good accompaniment is to match the accompaniment to the chord sequence that defines the harmonic structure of the music. Matching the chord sequence ensures that the accompaniment is harmonically consistent and in alignment with the melody. The rhythmic part of the accompaniment, which guides timing, duration, and placement of notes, is equally important. Without harmonic and rhythmic support, the generated accompaniment may become inconsistent or misaligned with the melody, undermining musical coherence.

This chapter presents a method for accompaniment generation called FlexFlow that builds upon the harmonization result of FlexChord. It begins by introducing the computational representation of musical elements necessary for style-specific accompaniment generation.

5.1 Representation of Music Elements

As described in Section 4.1, the melody M is represented as a sequence of pitches p_i over time index $i = 0, \dots, m - 1$. It is organized into melody fragments h_j with strong beat index $j = 0, \dots, n - 1$. Each melody fragment h_j is associated with a chord c_j that is either set manually or generated by FlexChord. It has a structure label λ_j that takes the value of ‘P’, ‘S’, or ‘ ’ respectively for end of phrase, end of section or neither. If fragment h_j repeats an earlier fragment $h_{j'}$ exactly, then its structure label $\lambda_j = j'$. In addition, fragment h_j has a section label that indicates its section type according to the melody’s music form such as A B, A B A and A B A C A (Section 2.2).

The objective of accompaniment generation is to produce, for each melody fragment h_j , an accompaniment fragments g_j of the same length. For ease of discussion, let's consider the case of monophonic accompaniment where g_j contains a single sequence of pitches. Extension to polyphonic accompaniment will be discussed in Section 5.3.2.

5.2 Problem Formulation

Accompaniment generation involves creating harmonic and rhythmic support for a given melody. It must address several key aspects, including accompaniment style, pitch progression, accompaniment-chord matching, accompaniment-melody matching and closure. Rhythmic pattern plays a secondary role because accompaniment typically contains regular rhythmic pattern. Thus, this thesis considers only rhythmic pattern at the end of phrases and sections that indicate closure. Other possible rhythmic patterns are highlighted in Section 6.1.

5.2.1 Accompaniment Style

The style of an accompaniment is captured by the reference accompaniment phrases. Let F denote a reference accompaniment phrase. It contains one or more reference accompaniment fragments f_l , for $l = 0, \dots, \eta - 1$. The number of reference fragments η typically takes the value of 1, 2, 4, 8, 16, etc. A reference fragment f_l has the same length as a melody fragment h_j and a target accompaniment fragment g_j . Reference fragments are related to target fragments by a specific correspondence between them:

$$l = j \bmod \eta. \quad (5.1)$$

In particular, f_0 corresponds to g_j , for $j = 0, \eta, 2\eta, \dots$, f_1 corresponds to g_j for $j = 1, \eta + 1, 2\eta + 1, \dots$, etc.

For a melody with multiple sections such as A B A, the melody of section B is composed to contrast that of section A. To reflect the stylistic contrast between section A and section B, two contrasting reference accompaniment phrases F_a and F_b are needed to support the two sections. There are many possible choices of contrasting reference phrases F_a and F_b , each conveying a different accompaniment

style. Therefore, the stylistic choice of reference phrases should be left to the user to decide.

The simplest way for a user to specify accompaniment style is to provide one or more reference music pieces with accompaniments of the desired style. The contrasting reference phrases can come from different sections of a single reference music, or from different reference music pieces. The length η is also specified by the user. For simple accompaniment with repeating rhythmic pattern, η can be as small as 1 or 2. For more interesting and complex accompaniment, η can be larger, up to the length of a melody phrase. Given the reference music and the user's choice, it is easy to apply a preprocessor to extract reference accompaniment phrases from the reference music.

5.2.2 Pitch Progression

The style of a reference accompaniment is conveyed in its pitch progression and rhythmic pattern. FlexFlow keeps the rhythmic pattern, i.e., pitch onset pattern, of a reference accompaniment fragment unchanged in the target accompaniment fragment, except for closure (Section 5.2.5). It changes only the pitch levels in the target accompaniment fragment while retaining the pitch progression of the reference accompaniment fragment.

Pitch progression can be characterized in two ways, yielding detailed property and overall property. Detailed property of pitch progression is characterized by two factors, namely pitch direction and pitch difference. Let p_i denote a non-rest pitch in reference accompaniment fragment f_l . Then, its pitch direction is $\text{sign}(p_i - p_{i-1})$ and pitch difference is $|p_i - p_{i-1}|$. Suppose reference fragment f_l corresponds to target fragment g_j . To preserve the detailed property of pitch progression, the pitches in fragment g_j should follow the same pitch directions and similar pitch differences as those in fragment f_l .

Overall property characterizes the shape of pitch progression or pitch contour. [CZB+21] demonstrates that the shape of pitch contour can be captured by the Discrete Cosine Transform (DCT) of the pitch sequence. The first DCT coefficient represents the overall pitch level of the pitch sequence. The next three DCT coefficients form a 3D vector that represents the overall shape of pitch contour

that is independent of the overall pitch level. Two fragments with similar pitch contours have similar DCT vectors. The difference in DCT vectors is used to measure the consistency of generated accompaniment fragments in the same melody section (Section 5.4).

5.2.3 Accompaniment-Chord Matching

Matching of accompaniment to chord progression is important for ensuring harmonic coherence of the generated accompaniment. A key principle of accompaniment-chord matching is that most, if not all, accompaniment pitches should be chord pitches, which anchor the harmonic structure in music. Non-chord pitches, such as passing or suspension notes, may be used sparingly to introduce tension or variation. Excessive use can weaken harmonic coherence and disrupt the relationship between accompaniment and melody [FW13; Smi+18]. In FlexFlow, accompaniment-chord matching is achieved by ensuring that all pitches of accompaniment fragment g_j are chord pitches with respect to their associated chord c_j . Allowance of non-chord pitches is achieved by accompaniment-melody matching (Section 5.2.4).

5.2.4 Accompaniment-Melody Matching

Music perception studies point to the match between an accompaniment fragment g_j and a melody fragment h_j as the amount of dissonance between their pitches [Hur94; Hur06; TDSG18]. The dissonance between two pitches p and q is a function of their pitch interval $\theta(p, q) = |p - q| \bmod 12$ (Section 4.2.2). Lower dissonance value corresponds to higher harmonic coherence when the pitches are sounded together.

A similar concept can be defined for the dissonance $d(h_j, g_j)$ between melody fragment h_j and accompaniment fragment g_j . It is measured according to the dissonance between the pitches p_i of melody fragment h_j and the pitches r'_i of accompaniment fragment g_j :

$$d(h_j, g_j) = \sum_i d(p_i, r'_i), \text{ for } p_i, r'_i \neq \text{rest}, \quad (5.2)$$

5.2.5 Closure

The melody of a phrase or a section is often ended by a sustained pitch with an appropriate duration. These sustained pitches give a sense of partial or full closure, analogous to comma and period in English sentences. They help to punctuate a melody and clarify the organization of phrases and sections in the melody’s music structure [Tem11]. In the same way, the generated target accompaniment should end with sustained pitches at the end of selected phrases and sections to provide a sense of closure. The durations of the sustained pitches should match those in the melody. Phrase and section ending that requires closure is labeled with structure label $\lambda_j = \text{‘C’}$. Structure labels are provided by user with help of a melody structure analysis tool [DZD20].

5.3 Algorithm Design

The proposed algorithm FlexFlow generates accompaniment by modifying reference accompaniment to match the melody and the chords. It produces target accompaniment that is harmonious with the melody by minimizing the overall dissonance D between the target accompaniment and the melody:

$$D = \sum_{j=0}^{n-1} d(h_j, g_j). \quad (5.3)$$

To capture the style of the reference accompaniment, FlexFlow keeps the pitch directions identical between the reference and target accompaniments, and minimizes the differences in pitch intervals between them. In addition, it uses appropriate sustained pitches at phrase and section endings to provide closure.

5.3.1 FlexFlow Accompaniment Generation

The overall structure of FlexFlow (Algorithm 2) is to use a reference accompaniment fragment f_l to generate the corresponding target accompaniment fragment g_j . The first step in Line 1 is to determine the accompaniment octave in which the target accompaniment lies. The general guideline is that the accompaniment pitches should lie in an octave that is lower than the melody pitches when both parts are played on a piano. Separating the accompaniment pitches and the melody

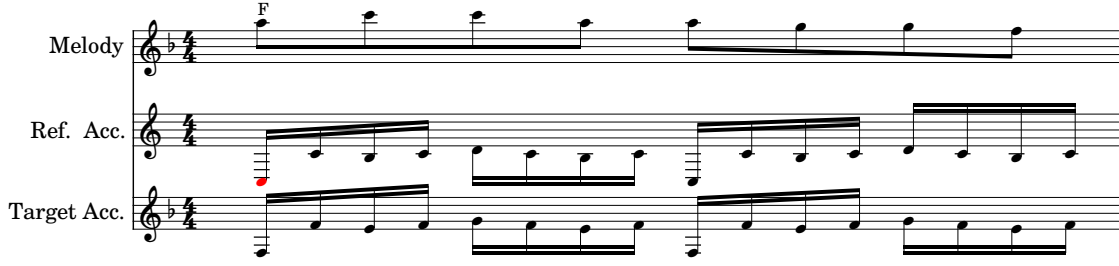


Figure 5.1: Transposition of reference accompaniment fragment. The lowest pitch of chord F is 53 in octave 3, and the lowest pitch (red note) of the reference fragment is 48 in octave 3. For the target accompaniment to be one octave below the melody, a transposition interval $48 - 53 + 12(1) = 7$ is applied. Thus, the reference fragment is transposed up by 7 to the target fragment to match the melody and chord F.

pitches into two different octaves helps to differentiate the audio qualities of the accompaniment and the melody. Depending on the pitch ranges of the melody and the reference accompaniment, the accompaniment octave can be set at 1 or 2 octaves below the melody. The actual choice of octave is to be decided by the user.

Next, Line 3 selects the reference accompaniment phrase F that matches the section label of melody fragment h_j . Line 4 selects from phrase F the accompaniment fragment f_l that corresponds to melody fragment h_j in accordance with Eq. 5.1.

Lines 5 and 6 perform pitch transformations to ensure harmonic coherence between the accompaniment and the melody and the associated chord. Line 5 transposes reference fragment f_l to target fragment g_j by matching the key of melody M and chord c_j , in the appropriate accompaniment octave. The pitch interval θ of transposition is calculated as the difference between the lowest pitch q in chord c_j and the lowest pitch r in reference fragment f_l , adjusted to fit the accompaniment octave:

$$\theta = r - q + 12c, \quad (5.4)$$

where $c \geq 1$ is dependent on the pitch ranges of the melody and the reference accompaniment. Typically, $c = 1$ or 2 . Then, the target accompaniment pitches r'_i in fragment g_j is computed by subtracting the transposition interval θ from the reference pitches r_i (Fig. 5.1).

$$r'_i = r_i - \theta. \quad (5.5)$$

Line 6 adjusts selected pitches in the target accompaniment fragment g_j to their

nearest desired pitches while preserving their pitch directions. Two adjustment options are available:

1. Adjust to chord pitch:

Non-chord pitches r'_i with respect to the associated chord c_j are selected. Each selected pitch r'_i is adjusted to the nearest chord pitch in c_j , in the same pitch direction as its corresponding reference pitch r_i . This adjustment ensures that all adjusted accompaniment pitches are chord pitches with respect to chord c_j , and they preserve the pitch directions and pitch differences in the reference accompaniment.

2. Adjust to reduce dissonance:

Accompaniment pitches r'_i that are dissonant with the corresponding melody pitch p_i are selected. An accompaniment pitch r'_i is considered dissonant with p_i if $d(p_i, r'_i) > 0.4$. Each selected pitch r'_i is adjusted to the nearest pitch such that dissonance $d(p_i, r'_i) \leq 0.4$ while retaining the pitch direction of its corresponding reference pitch r_i . This adjustment ensures that all accompaniment pitches are not dissonant with their corresponding melody pitches.

Pitch adjustment for option 1 is performed as follows. A pitch r'_i may be set in one of three octaves, namely $r'_i, r'_i + 12$ and $r'_i - 12$. With respect to the pitch $r'_i, r'_i + 12$ leads to a rising pitch, $r'_i - 12$ leads to a lowering pitch, and r'_i leads to either rising, lowering or level pitch. There is no need to consider octaves further than ± 12 because the goal is to choose the nearest pitch. A triad chord has three pitches, and there are altogether 9 candidate pitches to choose from. A 7th chord has four pitches, and there are altogether 12 candidate pitches to choose from. Thus, the adjustment of pitch r'_i is to find, among these candidate pitches, the one that has the same pitch direction as reference accompaniment pitch r_i , and is nearest to r'_i in terms of pitch difference. Then, pitch r'_i is changed to the chosen candidate pitch level. Pitch adjustment for option 2 is performed in a similar way, except that low-dissonant pitches are considered instead of chord pitches (Fig. 5.2). Alternatively, pitch adjustment can be applied only to onset pitches, with pitch direction and pitch difference defined for consecutive onsets. Then, non-onset pitches



Figure 5.2: Pitch adjustment in accompaniment. Rows 2, 3, 4 and 5 illustrate, respectively, target accompaniments after transposition, adjustment to chord pitches, adjustment to minimize dissonance and adjustment to reduce dissonance while keeping many chord pitches. The red notes in row 3 denote non-chord pitches that are adjusted to chord pitches, and those in row 4 and 5 denote pitches adjusted to reduce dissonance.

r'_i that follow an onset pitch are set to the same adjusted pitch level as the onset pitch.

Option 1 ensures that all accompaniment pitches are chord pitches regardless of the melody pitches. So, the dissonance between accompaniment and melody may be higher than 0.4. On the other hand, option 2 ensures that the accompaniment is harmonious with the melody regardless of the associated chord. Consequently, some accompaniment pitches may be non-chord pitches and may not fit the melody mode. Therefore, option 1 and option 2 are conflicting options. One way of balancing option 1 and option 2 is by relaxing them as follows. First, apply option 1 to obtain only chord pitches in the accompaniment. Next, identify the accompaniment pitches, which are now chord pitches, with dissonance > 0.5 compared to their corresponding melody pitches. Then, adjust these pitches to the nearest pitches with dissonance ≤ 0.5 while preserving their pitch directions. This method retains a small number of non-chord pitches while keeping the overall dissonance ≤ 0.5 . Fig. 5.2 row 5 shows that the accompaniment pitches belong to the major mode despite allowing some non-chord pitches.

Lines 7 and 8 adjust the last pitch onset and its duration in target fragment g_j with structure label λ_j of 'C' to provide closure. For example, the melody always



Figure 5.3: Section closure in accompaniment. This closing section labeled ‘C’ ends with a sustained pitch. Thus, the target accompaniment ends with a matching sustained pitch (red note). Its pitch value is set to the root pitch of chord F, while preserving the pitch direction of the corresponding reference pitch.

ends with a sustained pitch at the end of a song (Fig. 5.3). The sustained pitch, along with its onset time index, is copied to the target fragment, and the pitch level is set to the bass pitch in the associated chord c_j while retaining pitch direction. This pitch adjustment is also applied to the end of a phrase if the melody ends a phrase with a sustained pitch.

In general, the rhythmic pattern of the generated accompaniment can be adjusted as well. However, pitch adjustment is more critical for ensuring that the generated accompaniment matches the target chords and harmonically supports the melody. The listener study results (Section 5.4.3) indicate that listeners are more sensitive to dissonant harmony than to minor rhythmic variations in the accompaniment. Thus, only simple adjustment of rhythmic pattern for closure is considered. Other possible ways of adjusting rhythmic pattern are highlighted in Section 6.1.

Lines 9 and 10 handle repetition. When the structure label λ_j is an integer j' , it indicates that the melody fragment h_j is an exact repetition of an earlier fragment $h_{j'}$. In this case, the accompaniment fragment $g_{j'}$ is copied to g_j .

FlexFlow runs for exactly n iterations, one for each strong beat index j , to produce n accompaniment fragments. Each iteration performs a bounded set of operations. Line 3 selects a reference phrase with a matching section label from a small set of reference phrases. Line 4 selects a corresponding reference fragment in the reference phrase. This selection can be achieved by direct access through the correspondence between the target fragment index j and reference fragment index l (Eq. 5.1). Line 5 performs transposition of $\alpha\beta$ pitches in a fragment. Line 6 adjusts fewer than $\alpha\beta$ pitches in a fragment. Each adjustment chooses an appropriate pitch



Figure 5.4: An example of polyphonic accompaniment. Each stack of notes is a poly-pitch that contains one or more pitches.

from a small set of candidate pitches. These small numbers are independent of the number of fragments n , and they can be regarded as constant. Therefore, the time complexity of FlexFlow is $O(n)$.

Algorithm 2: FlexFlow Accompaniment Generation

Input: Melody M , chords c_j , reference accompaniment phrases $\{F\}$.

Output: Target accompaniment G .

- 1: Determine the accompaniment octave.
 - 2: **for** $j = 0, \dots, n - 1$ **do**
 - 3: Select the reference accompaniment phrase F that matches the section label of melody fragment h_j .
 - 4: Select the reference accompaniment fragment f_l in F that corresponds to melody fragment h_j (Eq. 5.1).
 - 5: Transpose reference fragment f_l to target fragment g_j by matching the key of melody M and chord c_j .
 - 6: Adjust selected pitches in g_j to their nearest desired pitches while retaining their pitch directions.
 - 7: **if** $\lambda_j = \text{'C'}$ **then**
 - 8: Adjust the last pitch onset and its duration in target fragment g_j , if needed, according to those in the melody fragment h_j .
 - 9: **else if** λ_j is an integer j' **then**
 - 10: Set $g_j = g_{j'}$.
-

5.3.2 Extension to Polyphonic Accompaniment

In music theory, a polyphonic music consists of several lines of melody called voices. The simultaneous sounding of multiple voices enrich the texture of the music. In contrast, a monophonic music consists of only one single melody line. The discussions in the previous sections focus on monophonic accompaniment. This section illustrates how to extend FlexFlow to handle polyphonic accompaniment.

Polyphonic music such as chorales and quartets is organized into separate lines, one for each voice [HPN17; Yin+23]. In contrast, the polyphonic accompaniment considered in FlexFlow is not necessarily organized into separate lines (Fig. 5.4). Instead of organizing a polyphonic accompaniment into multiple lines, it is more appropriate to organize the pitches that appear at the same time index as a poly-pitch. Then, a polyphonic accompaniment can be represented by a single sequence of poly-pitches, each containing one or more pitches.

The central idea of extending FlexFlow to polyphonic accompaniment is to characterize each poly-pitch by a representative pitch. Then, a polyphonic accompaniment can be regarded as a monophonic accompaniment by FlexFlow. This concept is adopted in FlexFlow in the following ways.

After transposition, each poly-pitch is adjusted according to either option 1, option 2 or the balanced version of options 1 and 2 (Fig. 5.5). To avoid duplicate pitches within a poly-pitch, any candidate pitch that causes potential pitch collisions is not considered for pitch adjustment.

Figure 5.5 illustrates an interesting example. The poly-pitch in beat 3 that is adjusted to chord pitches happen to have low dissonance. Therefore, options 1, 2 and the balanced option all produce the same result for beat 3. On the other hand, they produce different results for beat 4.

5.3.3 Comparison to Existing Methods

Before comparing the performance of FlexFlow (FF) with comparable existing methods, namely AccoMontage and Whole Song Generation, it is useful to discuss the similarities and differences between them.

AccoMontage (AM) [Zha+24] is a hybrid method that combines hidden Markov model (HMM) and variational autoencoder (VAE). The HMM applies dynamic programming to select an accompaniment phrase from a reference set to pair with each melody phrase. It performs the selection by matching the rhythm and chords of the accompaniment phrase to those of the melody phrase. The selected accompaniment phrases are refined by the VAE, which is trained to minimize accompaniment reproduction loss and the KL divergence between the learned distribution and target Gaussian distribution of accompaniment, conditioned on



Figure 5.5: Poly-pitch adjustment in accompaniment. Rows 2, 3, 4 and 5 illustrate, respectively, target accompaniments after transposition, adjustment to chord pitches, adjustment to minimize dissonance and adjustment to reduce dissonance while keeping many chord pitches. The red notes in row 3 denote non-chord pitches that are adjusted to chord pitches, and those in row 4 and 5 denote pitches adjusted to reduce dissonance.

matching the chords of the melody.

Whole Song Generation (WSG) [WMX24] is a series of cascaded diffusion models that generate structured music across multiple levels, ranging from phrase structure labels to melody phrases, chords, and individual accompaniment pitches. The diffusion models aim to denoise the input to reproduce the training accompaniment. WSG is trained to produce accompaniment phrases that capture both fine-grained pitch relationships and the broader music structure.

During accompaniment generation, AM, WSG and FF adopt similar two-stage approach: (1) reference accompaniment selection, and (2) pitch adjustment. In Stage 1, AM selects accompaniment phrases from a reference set provided by the user. If the reference set contains only one reference phrase, then AM uses the same reference phrase for all melody phrases. WSG takes a user-specified reference phrase as input and uses it for all melody phrases. Another WSG setup supports external control for different melody sections. However, this WSG setup is not publicly available. FF uses a single reference accompaniment phrase provided by the user. Alternatively, a user can provide different reference phrases to FF for different melody sections.

In Stage 2, AM applies VAE to adjust pitches of the selected reference phrases to match the melody and its associated chords. The reference phrases with pitches adjusted are the generated accompaniment. WSG applies diffusion method to adjust the pitches of the selected reference phrase to match the melody. Chords are handled as part of the diffusion process. FF transposes the selected reference phrase according to the bass pitches of the melody chords, and then adjusts the accompaniment pitches to match the chords and, optionally, to reduce dissonance with the melody, while preserving the pitch directions and pitch differences of the reference phrase.

In summary, AM, WSG and FF all require user specified reference accompaniment phrases. They differ primarily in the way they adjust the pitches of the reference phrase to match the melody and the associated chords. AM and WSG adjust pitches using trained neural networks. On the other hand, FF performs direct, specific adjustment of individual pitches, conditioned on matching chords, reducing dissonance and preserving pitch directions and pitch differences.

5.4 Experiments and Discussions

This section presents two types of tests: single-section test and full-length test. Single-section test is used for quantitative evaluation of FlexFlow’s performance compared to two existing methods, namely AccoMontage (AM) [Zha+24] and Whole Song Generation (WSG) [WMX24] (Section 5.3.3). Both single-section test and full-length test are used for musicological evaluation.

5.4.1 Data Preparation

Two public datasets, namely Pop909 [Wan+20b] and Wikifonia-Music-Text (WKF) [WMX24] were used for the accompaniment generation task. Pop909 contains lead sheets with polyphonic piano accompaniments in pop style. It contains 857 songs in 4/4 time, sufficient for training deep neural network methods. In contrast, WKF contains lead sheets without accompaniment for many genres. Due to the lack of associated accompaniment, it is unsuitable for training. 80 pop songs were selected from WKF to serve as testing examples.

Pop909 and WKF contain songs that are organized into multiple sections. Each

Table 5.1: Summary statistics of the datasets used for accompaniment generation.

Dataset	Number of Songs	Number of Phrases
Pop909	857	11,041
WKF	80	189

section contains 2 to 16 phrases. Each phrase consists of 8 to 32 fragments, in multiples of 8 fragments. Only songs with 4/4 time signature were used because the DNN method of WSG works only on songs with 4/4 time. Each fragment has a length of half a bar. Similar to existing work, their tempo was set to 120 beats per minute and the songs were transposed to all 12 music keys to ensure that WSG could handle test melodies of various keys. Each song was encoded in step-function representation with number of pitches per unit duration β set to 16. Recall that the desired stylistic contrast between the generated accompaniments in different melody sections should depend on user’s preference (Section 5.4.2). To eliminate subjective variability, quantitative evaluation was performed only on the main section, i.e., section A, of the songs.

The Pop909 dataset was split into training and validation sets and WKF served as the testing set. This resulted in a training, validation and test set ratio of 10:1:1. Training and validation are required for AM and WSG. In contrast, FlexFlow does not require training. Thus, FlexFlow does not require training set and validation set. It uses testing set for performance testing. Table 5.1 summarizes the statistics of the prepared datasets.

5.4.2 Quantitative Evaluation

(a) Test Procedure

The objective of this test is to evaluate the performance of FlexFlow (FF). Three versions of FF, which use different pitch adjustment options, are compared. FF1 applies pitch adjustment option 1 to obtain chord pitches and FF2 uses option 2 to minimize dissonance between accompaniment and melody. FF1.5 balances option 1 and option 2 to combine their advantages. In addition, FlexFlow was compared with AccoMontage (AM) and Whole Song Generation (WSG).

As discussed in Section 5.3.3, FF and AM can use multiple user-selected reference phrases, whereas the publicly available WSG supports only single reference phrase. For fair comparison, all methods are restricted to use a single common reference accompaniment phrase for the whole melody. Thus, the reference set of AM was restricted to contain a single reference accompaniment phrase. To emulate the selection of appropriate reference phrase by a human user, a separate AM was run on the testing melody, and the reference phrase selected by this AM for the first melody phrase was used as the common reference phrase.

The experiments were conducted on a MacBook Pro equipped with a quad-core Intel Core i7 processor running at 2.2 GHz and 16GB of RAM. The average execution time per test song for FlexFlow, AM, and WSG was 0.19 s, 4.47 s and 4.56×10^4 s, respectively.

(b) Quantitative Test Metrics

For objective comparison between FlexFlow and existing methods, a set of simple but musically meaningful metrics were identified:

- Bass hit ratio B

An accompaniment fragment g_j should be set at the right pitch level to harmonize with the melody according to its associated chord c_j . This is achieved when the fragment's lowest pitch is the same as the bass pitch of its associated chord, a condition called bass hit. Let n_b denote the number of fragments that achieve bass hit and n denote the total number of fragments in the song. Then, the bass hit ratio B measures the fraction of fragments that achieve bass hit:

$$B = \frac{n_b}{n}. \quad (5.6)$$

A higher B value indicates better match between the accompaniment and the associated chords. The value of B ranges from 0 to 1.

- Chord pitch ratio R0

This metric measures the ratio of chord pitches n_c to non-chord pitches n_n in the accompaniment fragment g_j with respect to the associated chord c_j :

$$R0 = \frac{n_c}{n_c + n_n}. \quad (5.7)$$

It also measures the harmony between the accompaniment and the associated chord. The higher the R0 value, the better is the match between the accompaniment and the chord. The value of R0 ranges from 0 to 1.

- Dissonance D

The average dissonance D between melody and accompaniment is defined as:

$$D = \frac{1}{n} \sum_j d(h_j, g_j), \quad (5.8)$$

where $d(h_j, g_j)$ is the dissonance between the corresponding pitches of melodic fragment h_j and accompaniment fragment g_j at the same time index i (Eq. 5.2). The lower the D value, the more harmonious is the match between the accompaniment and the melody. The value of D ranges from 0 to 1.

- Contour Difference C

Contour difference measures the consistency between the generated accompaniment phrases in a melody section. It is based on the difference in pitch contour between corresponding fragments g and g' measured in terms of their DCT vectors $\mathbf{v}$ and $\mathbf{v}'$ (Section 5.2.2):

$$\phi(g, g') = \|\mathbf{v} - \mathbf{v}'\|. \quad (5.9)$$

Contour difference is the average difference over all pairs of corresponding accompaniment fragments in the same section S :

$$C = \frac{1}{|S|^2} \sum_{g, g' \in S} \phi(g, g'). \quad (5.10)$$

The lower the C value, the more similar is the contours of accompaniment phrases. For two contours that are an exact transposition of each other, $C = 0$. For two different contours, C can be higher than 100.

- Contour Difference C'

This metric is a variant of contour difference C. It excludes fragments at the end of a phrase or a section that are adjusted to ensure proper closure.

Table 5.2: Quantitative evaluation of accompaniment generation methods.

	Accompaniment -chord match		Accompaniment -melody match	Contour Difference		
Method	B $\uparrow$	R0 $\uparrow$	D $\downarrow$	C $\downarrow$	C' $\downarrow$	C'/C $\downarrow$
AM	0.67	0.93	0.40	29.81	26.77	90%
WSG	0.43	0.60	0.46	262.68	245.31	93%
FF1	1.00	0.99	0.39	15.22	9.54	63%
FF1.5	1.00	0.85	0.34	35.03	27.40	78%
FF2	1.00	0.71	0.32	39.31	31.39	80%

(c) Quantitative Comparison of Various Methods

Table 5.2 tabulates the quantitative results of all the methods. All FF variants achieve the perfect bass hit ratio B of 1.0, indicating that the accompaniment fragments are set at the bass pitches of their associated chords. This is achieved by Line 5 of FF algorithm that transposes the accompaniment fragments so that the fragments' lowest pitches are the same as the bass pitches of their associated chords. In contrast, WSG achieves the lowest B value because its multi-stage diffusion method cannot simultaneously handle the properties of music structure, melody and chords well. AM achieves a slightly higher B value than does WSG because it is trained to generate accompaniment conditioned on the associated chords.

In theory, FF1 should achieve the perfect chord pitch ratio R0 of 1.0 because it sets all accompaniment pitches to chord pitches. Instead, test result indicates that FF1's R0 is very closer to but not exactly 1.0. This is due to a small amount of held pitches that cross fragment boundaries. Held pitches are chord pitches with respect to the chords of the preceding fragments. As they continue unchanged across fragment boundaries, they may not be chord pitches with respect to the chords of the succeeding fragments. FF2 minimizes dissonance instead of keeping chord pitches and FF1.5 balances the option of keeping chord pitches and reducing dissonance. Therefore, among the three variants, FF2 achieves the lowest B value. AM generates accompaniment fragments conditioned on the associated chords. Since this condition can only be imposed as a soft constraint in neural network, AM achieves the second highest R0 value, just lower than that of FF1. In contrast, WSG achieves the lowest R0 because its multi-stage diffusion model cannot match the chords well.

All methods achieve an average dissonance D of 0.46 or less, indicating that their

dissonance is low. This is due to the methods' matching of accompaniment phrases to melody phrases. FF2 achieves the lowest D value because it explicitly minimizes dissonance between the accompaniment fragments and the melody fragments. FF1 keeps chord pitches instead of explicitly minimizing dissonance. Since chord pitches tend to have low dissonance with the melody, FF1's D value is lower than those of AM and WSG. FF1.5 balances between keeping chord pitches and reducing dissonance. Thus, its D value is in between those of FF1 and FF2.

Test results about contour difference C indicate that FF1 best retains the contour of the reference accompaniment phrase in the generated accompaniment phrases. In other words, FF1 best maintains consistency among generated accompaniment phrases, which is important for preserving the style of reference accompaniment. By reducing dissonance, FF1.5 and FF2 cause more changes to phrase contours, resulting in larger C values. AM retains the reference contour well, and achieves the second highest C value. In contrast, WSG does not impose consistency in the generated accompaniment phrases. Consequently, it has very high C value compared to the other methods.

To handle closure, FF variants adjust the rhythmic pattern of the accompaniment fragments at the end of phrase and section with the structure label of 'C'. This adjustment is expected to change the phrase contours and results in higher contour difference. By discounting closing fragments, C' values give better indication of how well the methods maintain consistency in the generated accompaniment. In direct correlation to the C values, FF1 has the lowest C' value, whereas WSG has the highest C' value. AM, FF1.5 and FF2 have roughly the same lower C' values than does WSG.

The ratio of C'/C indicates the amount of phrase and section closure handled by a method. FF1 explicitly handles closure and achieves the lowest C'/C ratio of 63%. FF1.5 and FF2 also explicitly handle closure. Nevertheless, their effects are less apparent due to their larger C values. In comparison to FF1, they achieve a higher C'/C ratio of about 80%. AM and WSG have very high C'/C ratios of 90% or more. This indicates that AM and WSG do not handle closure well.

In summary, FF1 has the best overall performance, with AM and FF1.5 following closely behind FF1. Indeed, the polyphonic accompaniments generated by FF1 sound more harmonious than those of FF1.5 and FF2 because the pitches generated

by FF1 are chord pitches. When these pitches appear in a poly-pitch, they sound like a regular chord, providing harmonious support to the melody. FF variants achieve good performance because the accompaniment generation algorithm can perform fine-grained adjustment of individual pitches in the generated accompaniment. AM applies VAE to adjust pitches to match melody and chords. WSG uses diffusion models to perform overall pitch adjustment, which lacks fine-grained control for simultaneously handling multiple conditions of pitch progression, accompaniment-chord matching, accompaniment-melody matching and closure.

5.4.3 Listener Study

A listener study was conducted to compare the perceived accompaniment quality of various methods. 40 participants aged 18 to 65 years (average age = 41.6 years, SD = 11.8 years) were recruited via CloudResearch. All participants reported having at least five years of music training and being able to play the piano. Each participant was presented with two 8-bar songs with polyphonic accompaniment. One of the pair was either the human-composed (HC) accompaniment or the results of FlexFlow with option 1 (adjust to chord pitches). The other was a result of either AM or WSG. The participant was asked to pick the one whose accompaniment was more harmonious with the melody. In addition, the participant was asked to rate how harmonious each song was on a 5-point Likert scale. The ratings were used to verify whether the participant’s choice was consistent with the ratings. To reduce listener’s fatigue, each participant repeated the above procedure for only 20 pairs of songs instead of 40 pairs as in Section 4.4.4 because it takes longer to listen to accompaniment results.

As discussed in Section 4.4.2, there is no ground truth in harmonization. A HC accompaniment does not serve as a ground truth but is used to verify whether the participants’ evaluation are reliable. For example, if the participants judge HC accompaniment to be less harmonious than a DNN’s result, it does not mean that HC accompaniments are less harmonious than the DNNs’ accompaniments. Instead, it means that the participants do not understand accompaniment quality well and therefore, the participants’ evaluation is unreliable. On the other hand, if the participants judge HC accompaniment to be more harmonious than the DNNs

Table 5.3: Results of the listener study on accompaniment generation.

Cases	AM	WSG
H+	61 (49.2%)	87 (70.7%)
H−	63 (50.8%)	36 (29.3%)
F+	70 (56.9%)	84 (68.3%)
F−	53 (43.1%)	39 (31.7%)
H+ and F+	38 (31.2%)	67 (55.4%)
H+ and F−	23 (18.9%)	20 (16.5%)
H− and F+	31 (25.4%)	16 (13.2%)
H− and F−	30 (24.6%)	18 (14.9%)
Total votes	122 (100%)	121 (100%)

results it means that the participants’ evaluation is reliable. Therefore, FlexFlow is not compared with HC accompaniment, which helps to reduce listener’s fatigue.

The participants’ assessments were grouped into one of the following cases:

- H+: Human-composed accompaniment is more harmonious than DNN’s result.
- H−: Human-composed accompaniment is less harmonious than DNN’s result.
- F+: FlexFlow’s result is more harmonious than DNN’s result.
- F−: FlexFlow’s result is less harmonious than DNN’s result.

Table 5.3 indicates that a large majority of participants judged human-composed accompaniments to be more harmonious than WSG’s results, and roughly equal number of participants judged either human-composed accompaniments or AM’s results as more harmonious than the other. These results indicate that the participants understand and appreciate accompaniment. Therefore, the listener test results are reliable. Most participants judged FF’s results to be more harmonious than those of AM and WSG (56.9% and 68.3% respectively). Moreover, most participants judged both FF’s results and human-composed accompaniments as more harmonious than those of AM and WSG (H+ and F+ case). These results indicate that FF’s results are better than those of AM and WSG, and its quality compared to that of DNNs is close to the quality of human-composed accompaniment compared to DNNs’ results.

The listener study results are likely due to the following reasons. The accompaniments are polyphonic. In a polyphonic accompaniment, the pitches that appear at the same time sound like chords. The participants, who play piano and have at least 5 years of music training, are more sensitive to the pitch harmony than the matching of nuanced rhythmic patterns between the accompaniment and the melody. As shown in Table 5.2, FF1’s results contain only chord pitches, AM’s results contain some non-chord pitches, and WSG’s results contain many non-chord pitches. Therefore, FF1’s results are judged to be more harmonious than those of AM and WSG. The human-composed accompaniments may contain roughly equal amount of non-chord pitches compared to AM’s results, and fewer non-chord pitches compared to WSG’s results. Therefore, the human-composed accompaniments are judged as equally less harmonious as AM’s results and more harmonious than WSG’s results.

5.4.4 Musicological Evaluation of FlexFlow Variants

Figure 5.6 shows the accompaniment generated by the FF variants for bars 9 to 16 of *Sentimental Me*, a song in the testing set. All FF variants preserve the style of the reference accompaniment by maintaining similar pitch direction and syncopated rhythmic patterns. For instance, the first two bars demonstrate an upward pitch direction and a syncopated rhythmic pattern that closely match the reference style.

The accompaniment of FF1 contains only chord pitches, resulting in minimal dissonance between accompaniment and melody pitches throughout the piece, except for bars 15 and 16 where a notable clash occurs. In bars 15 and 16, the chord pitch E in the A major chord clashes with the melody pitch F across six beats. FF2, on the other hand, ensures that accompaniment-melody dissonance ≤ 0.4 , which causes slight change to the pitch contour. Bar 10 highlights this difference, showing low dissonance pitches that create tension while avoiding direct clashes with the corresponding melody pitch at each time point. However, the sequence of poly-pitches, which sounds like a sequence of chords, tend to lack harmonic coherence. For example, the sequence moves from an augmented chord (B $\flat$ -D-F $\sharp$) to an F major chord, then quickly shifts to an F minor chord within two bars. This change from augmented to major to minor within a bar adds confusion to the mode

The figure displays two systems of musical notation for the piece "Sentimental Me". Each system consists of a Melody line and three piano accompaniment lines labeled FF1, FF2, and FF1.5. The first system covers measures 9 to 12, and the second system covers measures 13 to 16. Chord symbols are placed above the melody: Gm7, C7, and F in the first system; C7, F, and A7 in the second system. The notation shows various rhythmic patterns and chord voicings for each variant.

Figure 5.6: Accompaniment of *Sentimental Me* by FF variants.

of the bar, which makes the chord progression less harmonically coherent.

FF1.5 balances the options of keeping chord pitches and reducing dissonance. For example, bar 10 contains a poly-pitch (F–G–C $\sharp$) with greater chord pitch dissonance than the surrounding consonant poly-pitches based on E $\flat$ major and C major chords. Notably, bar 11 resolves this harmonic tension with F major chords, unlike the sequence of dissonant poly-pitches produced by FF2 for the same melody bar. This example shows that dissonant poly-pitches are usually undesirable, unless they are required for specific styles, such as the spooky style of *Hall of Mountain King* (Section 4.4.5(b)). Note that these observations apply to polyphonic accompaniment but not monophonic accompaniment because monophonic accompaniment is a

The figure displays a musical score for the first section of the song 'Sentimental Me'. It consists of a Melody line and three accompaniment lines labeled AM, WSG, and FF1. The Melody line is in the treble clef, and the accompaniment lines are in the bass clef. The key signature is one flat (B-flat major/D minor) and the time signature is 4/4. The score is divided into two systems of four bars each, with bar numbers 1-8 indicated above the Melody line. The first system (bars 1-4) starts with a box labeled 'A8' and a Gm7 chord. The second system (bars 5-8) continues with C7, F, D7, and C7 chords. The accompaniment lines show different textures: AM uses arpeggiated chords, WSG uses whole scale glissandos, and FF1 uses a sequence of single pitches.

Figure 5.7: Accompaniment generated by various methods for *Sentimental Me*. This figure illustrates bars 1–8 of the first Section A.

sequence of single pitches rather than a sequence of chords. In other words, FF1 is more appropriate for polyphonic accompaniment, whereas FF1.5 and FF2 may be appropriate for monophonic accompaniment.

5.4.5 Musicological Evaluation of Various Methods

For this comparison test, AM, WSG and FF were applied to the song *Sentimental Me* with the following structure: A8, A8, B8, A8. It begins with two sections of Section A, followed by a contrasting Section B, and ends with a final Section A. Each section consists of 8 bars. Section A and Section B should have contrasting

A8

The figure displays two systems of musical notation for the accompaniment of 'Sentimental Me'. The first system covers measures 9 through 12, and the second system covers measures 13 through 16. Each system consists of four staves: Melody (top), AM (second), WSG (third), and FF1 (bottom). The Melody staff includes chord symbols: Gm7, C7, F, and A7. The AM staff shows a complex accompaniment with many beamed eighth notes. The WSG staff shows a simpler accompaniment with fewer notes. The FF1 staff shows a bass line with eighth notes. The time signature is 4/4.

Figure 5.8: Accompaniment generated by various methods for *Sentimental Me*. This figure illustrates bars 9–16 of the second Section A.

reference accompaniment phrases. To emulate the selection of reference phrases by a user, a separate AM was run to select the reference phrases from a reference set. Then, the first reference phrase selected for the first Section A and Section B were used as the common reference phrases for the test. These two reference phrases each has a length of 8 bars.

Figures 5.7 and 5.8 illustrate bars 1 to 8 of the first Section A and bars 9 to 16 of the second Section A, respectively. Note that WSG does not generate accompaniment result for the first melody phrase. The accompaniment of WSG in Fig. 5.7 is the actual reference phrase for Section A. This reference phrase is characterized by a regular rhythmic pattern of eighth notes across the phrase. The bass pitches often

alternate between a chord pitch and its octave. Comparing Fig. 5.7 and 5.8, it is clear that both AM and FF1 preserve the style of the reference phrase well in their accompaniment phrases in both Section A. They successfully reproduce the regular rhythmic pattern of eighth notes and octave bass pattern, while matching associated target chords. This contributes to stylistic consistency across phrases of accompaniment across both Section A. In contrast, WSG’s accompaniment diverges from the style of the reference phrase. For example, its accompaniment in bar 9 does not follow the accompaniment style of bar 1. Instead, it seems to continue with the style of bar 8, not knowing that bar 9 begins the second Section A. WSG alters the pitch progression and rhythm of the reference phrase significantly. In particular, it introduces syncopation in bar 9 by omitting pitch onsets at strong beats, thus shifting the expected rhythm emphasis.

Figure 5.9 illustrates bars 13 to 16 of Section A, followed by bars 17 to 20 of Section B. Comparing Fig. 5.9 with Fig. 5.8 reveals that the accompaniment style of Section B differs from that of Section A. Both FF1 and AM can reproduce the contrasting styles of the reference phrases between Section A and Section B. However, their treatments of harmonic contrast differ. The bass pitch of FF1’s accompaniment progresses from A to D, which conforms to the V–i cadence and enhances the harmonic distinction between Section A and Section B. FF1 achieves this by ensuring bass hit, i.e., the lowest-pitch of accompaniment fragment is the same as the bass pitch of the associated chord. On the other hand, AM produces a bass pitch A that is sustained from bar 16 to bar 17, as the chord transits from A major to D minor. As pitch A appears in both A major chord (A–C#–E) and D minor chord (D–F–A), it reduces the perceptual salience of harmonic contrast. The publicly available WSG does not support specification of different reference phrases for different sections. Thus, it uses a single reference phrase to generate the accompaniment of both Section A and Section B, without injecting contrast.

Figure 5.10 illustrates the last four bars of the song. FF1 produces a final F major chord that is held from bar 31 to bar 32, with the same duration as the melody note. This sustained chord that matches the sustained melody note provides a clear closure to the song. In contrast, AM continues the regular rhythmic pattern without noticeable change of rhythm, making the accompaniment sound open-ended. WSG produces a held chord from bar 31 to bar 32. In addition, it injects a short

The figure displays a musical score for the song "Sentimental Me". It is divided into two sections: Section A (bars 13-16) and Section B (bars 17-20). The score includes a melody line and four accompaniment methods: AM (Automatic Melody), WSG (Weighted Sum of Squares), and FF1 (Fast Fourier Transform). The melody line is in 4/4 time and features a C7 chord in bar 13, F chord in bar 14, A7 chord in bar 15, and Dm chord in bar 17. The accompaniment methods show varying degrees of coherence and contrast between sections. The FF1 method is noted for preserving coherence within and between sections.

Figure 5.9: Accompaniment generated by various methods for *Sentimental Me*. Bars 13–16 belong to the second Section A and bars 17–20 belong to Section B.

sixteenth-beat chord at the end of bar 31. This creates a slight sense of tension before the song ends, disrupting closure. Moreover, the pitches of the last poly-pitch (C–F–G–B $\flat$) do not match the F major chord exactly. Thus, they undermine harmonic coherence.

In summary, FF1 preserves coherence of accompaniment within and between sections of the same type while allowing for contrast in accompaniment across sections. It ensures proper section closure, and conforms to the cadence at the end of section. AM also preserves coherence within a section and allows for contrast across contrasting sections. But, it does not ensure proper section closure and does not

The figure displays a musical score for the piece 'Sentimental Me'. It consists of four staves: Melody, AM (Automatic Music), WSG (Weighted Score Generation), and FF1 (FlexFlow). The Melody staff is in treble clef, 4/4 time, and starts at measure 29 with a C7 chord. It features a melodic line with eighth and sixteenth notes, ending at measure 32 with a C' and F note. The AM staff shows a piano accompaniment with chords and moving lines. The WSG staff shows a different accompaniment style, with more complex chordal structures. The FF1 staff shows a third accompaniment style, with a more rhythmic and syncopated pattern. The staves are labeled on the left: Melody, AM, WSG, and FF1.

Figure 5.10: Closing accompaniment generated by various methods for *Sentimental Me*.

conform to cadence. WSG cannot preserve coherence within and between sections of the same type. The WSG setup with external support may be able to allow for contrast across contrasting sections. Like AM, WSG does not ensure proper section closure and does not conform to cadence.

5.4.6 Accompaniment in Various Styles

Figure 5.11 illustrates FlexFlow’s ability to generate accompaniment based on different reference accompaniment that reflect the folk, classical and jazz genres. The reference phrases are taken from Greensleeves (Row 2), Mozart’s Sonata No. 16 (Row 3), Bach Prelude in C (Row 4) and Girl from Ipanema (Row 5). Despite the diversity in pitch progression and rhythmic pattern between the reference phrases. By generating accompaniment that matches the specific reference phrase, FF1 consistently generate accompaniment that matches the specific style of the reference. For example, Fig. 5.11 row 2 illustrates a slow folk style, row 3 illustrates a slightly faster classical style, and row 4 illustrates a fast-paced baroque style. Row 5 shows FF1’s flexibility in adapting to jazz idioms, producing a bossa nova-style accompaniment with syncopated chord rhythms and a characteristic bass pattern.

Figure 5.12 further demonstrates FlexFlow’s ability to generate accompaniments with distinct moods for the same melody. By pairing different target chords with appropriate reference accompaniment phrases, FF1 creates versions that sound clearly

Figure 5.11: Accompaniment generated by FF in various styles.

Figure 5.12: Happy and sad sounding accompaniments by FF.

happy or sad. The happy version features major chords and a dense arpeggiated pattern, resulting in a lively feel, while the sad version uses minor chords and slower paced accompaniment, resulting in a more somber mood. These results indicate that FF1 can control the mood of an accompaniment through changes in the target chords and reference style.

5.4.7 Summary

Quantitative results illustrate the strengths of FlexFlow for phrases within a section. It directly caters to strong chord bass progression with high bass hit ratio (high B). It achieves high chord pitch ratio (large R0), indicating good accompaniment-chord matching. It chooses chord pitches with relatively low dissonance (small D). It maintains consistent phrasing within sections by achieving low contour difference (small C and small C') and low contour difference ratio (small C/C').

The listener study results reveal that most participants judge FlexFlow's accompaniments to be more harmonious than AM and WSG's accompaniments. In addition, more participants judge both the human-composed and FlexFlow's accompaniment to be more harmonious than those of AM and WSG, compared to the other cases. This indicates that FlexFlow's accompaniments are better than AM and WSG, and are comparable to those of the human-composed accompaniments.

Musicological evaluation shows that FlexFlow and its variants achieve style specificity by producing accompaniment pitches that match the style of a selected set of reference music. It produces a pop accompaniment that is similar to the reference example, but remains creative in the choice of pitches that maintain pitch direction and pitch difference of the reference. FF1 is more appropriate for polyphonic accompaniment, whereas FF1.5 and FF2 may be appropriate for monophonic accompaniment.

FlexFlow achieves stylistic consistency across phrases of the same sections, both rhythmically and harmonically. It achieves clear section contrast across distinct sections by producing bass pitch progression associated with the target chords while preserving the rhythmic pattern of the reference phrase selected for each section. Although the current version of FlexFlow does not change the rhythm of the reference phrase except for section closures, the pitches that it generates support the melody harmonically and contribute to a clear sense of structure and resolution.

Additional tests results show that FlexFlow is flexible at generating accompaniments of various styles for the same melody. By using reference phrases that capture a specific style for each section, FlexFlow can generate accompaniment without the need of training on large datasets. Moreover, by carefully selecting reference phrases, a user can effectively control the rhythmic pace of the accompaniment, allowing it to

adapt to different melodic rhythms while maintaining harmonic coherence. Overall, FlexFlow shows promise as a flexible accompaniment generator that can create pop, jazz and classical accompaniment for structured melodies.

Chapter 6

Future Work and Conclusion

6.1 Future Work

The proposed methods of FlexChord and FlexFlow generate stylistically consistent and harmonically coherent chords and accompaniments for melodies. Given user-specified desired styles of chord harmonization and accompaniment, FlexChord and FlexFlow can generate the chords and accompaniment that match user's desired styles without the need of learning from training examples. Nevertheless, there are enhancements that can be made to FlexChord and FlexFlow.

6.1.1 FlexChord

One possible enhancement is to enable key modulation within and across melody sections. Currently, FlexChord harmonizes melodies with a fixed key, which limits its ability to follow melodies that shift in pitch center over time. Extending the method to detect and adapt to key changes would allow it to better match the style and structure of longer and more varied melodies.

Another possible enhancement is to enable non-uniform chord rhythms. Chords are presently placed at regular strong beats, independent of the rhythmic structure of the melody. Adjusting chord placement to match melody rhythm, accented notes or phrase boundaries could produce harmonization that match the pacing and phrasing of the melody.

A further extension involves chord blending, where transition between chords are smoothed through shared pitches. Without blending, the chord set is limited to those defined by the reference chord transition model, limiting harmonic variety.

Incorporate chord blending would allow novel chords to be produced based on the reference chords and enable smoother chord progressions.

6.1.2 FlexFlow

FlexFlow retains the rhythm of a single reference phrase throughout each section. Introducing rhythmic adaptation would allow the accompaniment to vary its pacing or rhythmic density based on the melody, while still maintaining stylistic consistency with the reference phrases.

Another direction for future work is accompaniment style blending. Each section in FlexFlow is guided by a single reference phrase. Allowing for multiple references or interpolation between reference examples could introduce smoother stylistic transitions and finer-grained control over accompaniment. This approach contrasts with methods that produce stylistic variations without adhering to the structure of the melody.

Another practical extension is the introduction of explicit user controls over aspects such as rhythmic density, voice thickness or stylistic emphasis, while matching the melody structure. While reference phrases provide indirect control, providing configurable parameters would offer greater flexibility and enable users to tailor the generated accompaniment for applications including interactive media and therapeutic settings.

6.2 Conclusion

Automated music composition (AMC) is an important topic that consists of two main tasks: (1) the generation of music structure and melody, and (2) the generation of chord and accompaniment. Since melody generation is well studied, this thesis focuses on the second task of generating chords and accompaniment.

This thesis has presented FlexChord and FlexFlow, two complementary methods for generating style-specific chords and accompaniments for melodies that overcome the shortcomings of existing methods. FlexChord produces matching chords at strong beats while FlexFlow generates monophonic and polyphonic accompaniment to match the chords, according to the melody and its structure. The structured accompaniment preserves chord bass progression of the target chords as well as

rhythmic patterns and pitch progressions from reference phrases. FlexChord and FlexFlow apply direct optimization without the need of learning from large amounts of training examples. Therefore, they can produce chords and accompaniments that match different reference styles flexibly and efficiently.

Quantitative results and musicological evaluation show that both methods successfully produce style-specific chords and accompaniment without the need to train on large datasets. They generate chords or accompaniment that match the style of a selected set of reference music, which can be as little as a single music piece or phrase. They are capable of handling full-length melodies with multiple sections, supporting global music structure by maintaining section-level consistency and introducing section contrasts where appropriate. Results of human perception tests show that their generated chords and accompaniments are better than those of comparable existing methods and are close to human-composed chords and accompaniments.

Note that harmonization and accompaniment generation are discrete problems because pitch, chord and time index are discrete. Moreover, the dissonance of two pitches sounding together does not follow a monotonic relationship of their pitch interval. In particular, two pitches that differ by a semitone, the smallest possible pitch interval, have the largest dissonance. For chords and accompaniment to harmonize well with melody, they need to have low dissonance with melody pitches. Both FlexChord and FlexFlow applies discrete optimization that handles pitch, pitch dissonance and hard constraints precisely. These algorithms produce valid solutions that satisfy all hard constraints and are locally optimal with respect to their optimization objectives.

On the other hand, DNNs essentially perform approximation by learning to generate outputs that are as close as possible to the training outputs. They can only handle soft constraints in the form of weighted sum of loss function. Their outputs are floating-point values that must be rounded to integers to produce actual pitches or to determine whether a pitch is active. For example, given a melody pitch 64, rounding an accompaniment pitch 64.5 down to 64 results in zero dissonance, whereas rounding up to 65 introduces a dissonance of 1. Thus, they are not as precise as FlexChord and FlexFlow in handling pitches and dissonance. They cannot enforce hard constraints exactly and are brittle, as small variations in input or rounding can lead to suboptimal outputs.

In summary, this thesis contributes to the growing field of automated music composition by showing that style-specific harmonization and accompaniment can be achieved through structured, reference-based methods. The results underscore the importance of modeling not just harmony and rhythm but structure and style-specificity, both of which are essential to producing coherent and expressive music.

References

- [Agu+10] G. Aguilera et al. “Automated generation of contrapuntal musical compositions using probabilistic logic in Derive”. In: *Mathematics and Computers in Simulation* 80.6 (2010), pp. 1200–1211.
- [AW04] Moray Allan and Christopher Williams. “Harmonising chorales by probabilistic inference”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 17 (2004).
- [AM09] Torsten Anders and Eduardo R Miranda. “A computational model that generalises Schoenberg’s guidelines for favourable chord progressions”. In: *Sound and Music Computing (SMC)*. 2009, pp. 48–52.
- [And21] Anderson, Chris and Carlton, Dave and Miyakawa, Ryan and Schwachhofer, Dennis. *Hooktheory*. <https://www.hooktheory.com>. Accessed: 2022-10-06. 2021.
- [BC15] L Bigo and D Conklin. “A viewpoint approach to symbolic music transformation”. In: *International Symposium on Computer Music Multidisciplinary Research*. Springer, 2015, pp. 213–227.
- [BHP20] Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet. *Deep learning techniques for music generation*. Vol. 1. Springer, 2020.
- [BP20] Jean-Pierre Briot and François Pachet. “Deep learning for music generation: challenges and directions”. In: *Neural Computing and Applications* 32.4 (2020), pp. 981–993.
- [BLB07] J. Buchholz, E. Lee, and J. Borchers. *coJIVE: A System to Support Collaborative Jazz Improvisation*. Tech. rep. RWTH Aachen Univ, 2007. URL: <https://api.semanticscholar.org/CorpusID:16092297>.

- [Cap04] William E Caplin. “The classical cadence: Conceptions and misconceptions”. In: *Journal of American Musicological Society* 57.1 (2004), pp. 51–118.
- [CLC13] Pei-Chun Chen, Keng-Sheng Lin, and Homer H. Chen. “Automatic accompaniment generation to evoke specific emotion”. In: *IEEE International Conference on Multimedia and Expo (ICME)*. 2013, pp. 1–6. DOI: [10.1109/ICME.2013.6607423](https://doi.org/10.1109/ICME.2013.6607423).
- [Che+21] Yi-Wei Chen et al. “SurpriseNet: Melody Harmonization Conditioning on User-controlled Surprise Contours”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2021, pp. 105–112.
- [CC11a] C.-H. Chuan and E. Chew. “Generating and evaluating musical harmonizations that emulate style”. In: *Computer Music Journal* 35.4 (2011), pp. 64–82.
- [CC11b] Ching-Hua Chuan and Elaine Chew. “Generating and evaluating musical harmonizations that emulate style”. In: *Computer Music Journal* 35.4 (2011), pp. 64–82.
- [Civ+22] Miguel Civit et al. “A systematic review of artificial intelligence-based music generation: Scope, applications, and future trends”. In: *Expert Systems with Applications* (2022), p. 118190.
- [CL17] Tom Collins and Robin Laney. “Computer-generated stylistic compositions with long-term repetitive and phrasal structure”. In: *Journal of Creative Music Systems* 1.2 (2017).
- [Col+11] Tom Collins et al. “Modeling pattern importance in Chopin’s mazurkas”. In: *Music Perception* 28.4 (2011), pp. 387–414.
- [CSG17] Florian Colombo, Alexander Seeholzer, and Wulfram Gerstner. “Deep artificial composer: A creative neural network model for automated melody generation”. In: *Computational Intelligence in Music, Sound, Art and Design: International Conference (EvoMUSART)*. Springer. 2017, pp. 81–96.

- [Con18] Darrell Conklin. “Chord sequence generation with semiotic patterns”. In: *Machine Learning and Music Generation*. Routledge, 2018, pp. 6–20.
- [CZB+21] Bas Cornelissen, Willem H Zuidema, John Ashley Burgoyne, et al. “Cosine Contours: a Multipurpose Representation for Melodies.” In: *Proceedings of International Society for Music Information Retrieval Conference (ISMIR)*. 2021, pp. 135–142.
- [CSH18] Nailson Cunha, Anand Subramanian, and Dorien Herremans. “Generating guitar solos by integer programming”. In: *Journal of the Operational Research Society* 69 (2018), pp. 971–985.
- [DZD20] Shuqi Dai, Huan Zhang, and Roger B Dannenberg. “Automatic analysis and influence of hierarchical structure on melody, rhythm and harmony in popular music”. In: *Proceedings of the Joint Conference on AI Music Creativity (CSMC-MuMe)*. 2020.
- [Dai+21] Shuqi Dai et al. “Controllable deep melody generation via hierarchical music structure representation”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2021, pp. 143–150.
- [DPZZ20] R. De Prisco, G. Zaccagnino, and R. Zaccagnino. “EvoComposer: An Evolutionary Algorithm for 4-Voice Music Compositions”. In: *Evolutionary Computation* 28.3 (Sept. 2020), pp. 489–530. ISSN: 1063-6560. DOI: [10.1162/evco_a_00265](https://doi.org/10.1162/evco_a_00265).
- [Dég+22] Ken Déguernel et al. “Personalizing AI for co-creative music composition from melody to structure”. In: *Sound and Music Computing (SMC)*. 2022, pp. 314–321.
- [Dos05] Martin Dostál. “Genetic Algorithms As a Model of Musical Creativity—on Generating of a Human-Like Rhythmic Accompaniment”. In: *Computing and Informatics* 24.3 (2005), pp. 321–340.
- [Ebc88] Kemal Ebcioglu. “An expert system for harmonizing four-part chorales”. In: *Computer Music Journal* 12.3 (1988), pp. 43–51.

- [Ehr05] Matthias Ehrgott. *Multicriteria optimization*. Vol. 491. Springer Science & Business Media, 2005.
- [FW13] Dan Fox and Dick Weissman. *Chord Progressions: Theory and Practice: Everything You Need to Create and Use Chords in Every Key*. Alfred Music, 2013. ISBN: 9780739070567.
- [FG11] Alan Freitas and Frederico Guimaraes. “Melody harmonization in evolutionary music using multiobjective genetic algorithms”. In: *Sound and Music Computing (SMC)*. 2011.
- [FM81] J. Fux and A. Mann. *Study of Counterpoint: From Johann Joseph Fux’s Gradus ad Parnassum*. WW Norton, 1981.
- [GJ02] A. Gartland-Jones. “Can a genetic algorithm think like a composer?”. In: *Generative Art Conference*. 2002.
- [GM07] Michael Geis and Martin Middendorf. “An ant colony optimizer for melody creation with baroque harmony”. In: *IEEE Congress on Evolutionary Computation*. IEEE. 2007, pp. 461–468.
- [GC07] 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)*. Hyderabad, India, 2007.
- [Gog06] M. Gogins. “Score generation in voice-leading and chord spaces”. In: *International Computer Music Conference (ICMC)*. 2006.
- [Goi+19] Izaro Goienetxea et al. “Statistics-based music generation approach considering both rhythm and melody coherence”. In: *IEEE Access* 7 (2019), pp. 183365–183382.
- [GN21] 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.

- [GDG21] Jacek Grekow and Teodora Dimitrova-Grekow. “Monophonic Music Generation With a Given Emotion Using Conditional Variational Autoencoder”. In: *IEEE Access* 9 (2021), pp. 129088–129101. DOI: [10.1109/ACCESS.2021.3113829](https://doi.org/10.1109/ACCESS.2021.3113829).
- [GD97] Lorin Grubb and Roger B. Dannenberg. “A Stochastic Method of Tracking a Vocal Performer”. In: *International Computer Music Conference (ICMC)*. Michigan Publishing, 1997.
- [Gwe02] N. Gwee. “Complexity and Heuristics in Rule-Based Algorithmic Music Composition”. PhD thesis. Louisiana State University, 2002.
- [HPN17] Gaëtan Hadjeres, François Pachet, and Frank Nielsen. “DeepBach: a steerable model for Bach chorales generation”. In: *International Conference on Machine Learning (ICML)*. 2017, pp. 1362–1371.
- [HBE20] Shunit Haviv Hakimi, Nadav Bhonker, and Ran El-Yaniv. “BebopNet: Deep Neural Models for Personalized Jazz Improvisations”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2020, pp. 828–836.
- [HCC17] D. Herremans, C.-H. Chuan, and E. Chew. “A Functional Taxonomy of Music Generation Systems”. In: *ACM Computing Surveys* 50 (2017), pp. 69–101. DOI: [10.1145/3108242](https://doi.org/10.1145/3108242).
- [HS13] Dorien Herremans and Kenneth Sørensen. “FuX, an Android app that generates counterpoint”. In: *IEEE Symposium on Computational Intelligence for Creativity and Affective Computing, (CICAC)*. IEEE, 2013, pp. 48–55. DOI: [10.1109/CICAC.2013.6595220](https://doi.org/10.1109/CICAC.2013.6595220).
- [Hie19] Hiehn, Steve. *Chord Melody Dataset*. <https://github.com/shiehn/chord-melody-dataset>. Accessed: 2022-10-06. 2019.
- [HFM91] Hermann Hild, Johannes Feulner, and Wolfram Menzel. “HARMONET: A neural net for harmonizing chorales in the style of JS Bach”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 4 (1991).
- [Hsi+21] Wen-Yi Hsiao et al. “Compound Word Transformer: Learning to Compose Full-Song Music over Dynamic Directed Hypergraphs”. In: *AAAI Conference on Artificial Intelligence (AAAI)*. 2021, pp. 178–186.

- [HC05] 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 (CMST)*. 2005.
- [Hua+17] Cheng-Zhi Anna Huang et al. “Counterpoint by Convolution”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2017, pp. 211–218.
- [Hua+18] Cheng-Zhi Anna Huang et al. “Music Transformer: Generating Music with Long-Term Structure”. In: *International Conference on Learning Representations (ICLR)*. 2018.
- [HH20] Chih-Fang Huang and Cheng-Yuan Huang. “Emotion-based AI music generation system with CVAE-GAN”. In: *IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE)*. IEEE. 2020, pp. 220–222.
- [Hua+20] Tsung-Min Huang et al. “Play it again IMuCo! music composition to match your mood”. In: *International Conference on Transdisciplinary AI (TransAI)*. IEEE. 2020, pp. 9–16.
- [Hur94] D. Huron. “Interval-class content in equally tempered pitch-class sets: Common scales exhibit optimum tonal consonance”. In: *Music Perception* 11.3 (1994), pp. 289–305.
- [Hur01] D. Huron. “Tone and voice: A derivation of the rules of voice-leading from perceptual principles”. In: *Music Perception* 19.1 (2001), pp. 1–64.
- [Hur16] D. Huron. *Voice Leading: The Science Behind a Musical Art*. MIT press, 2016.
- [Hur06] David Huron. *Sweet Anticipation: Music and the Psychology of Expectation*. Cambridge, MA: MIT Press, 2006. ISBN: 9780262083451.
- [Jaq+16] Natasha Jaques et al. “Generating Music by Fine-Tuning Recurrent Neural Networks with Reinforcement Learning”. In: *Deep Reinforcement Learning Workshop, NIPS*. 2016.
- [JKA17] Jaehun Jeong, Yusung Kim, and Chang Wook Ahn. “A multi-objective evolutionary approach to automatic melody generation”. In: *Expert Systems with Applications* 90 (2017), pp. 50–61.

- [JYL24] Shulei Ji, Xinyu Yang, and Jing Luo. “A Survey on Deep Learning for Symbolic Music Generation: Representations, Algorithms, Evaluations, and Challenges”. In: *ACM Computing Surveys* 56.1 (2024), 7:1–7:39. DOI: [10.1145/3597493](https://doi.org/10.1145/3597493).
- [Ji+23] Shulei Ji et al. “RL-Chord: CLSTM-Based Melody Harmonization Using Deep Reinforcement Learning”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2023), pp. 1–14. DOI: [10.1109/TNNLS.2023.3248793](https://doi.org/10.1109/TNNLS.2023.3248793).
- [JKW17] Daniel D Johnson, Robert M Keller, and Nicholas Weintraut. “Learning to create jazz melodies using a product of experts.” In: *International Conference on Innovative Computing and Cloud Computing*. 2017, pp. 151–158.
- [KP+23] Maximos Kaliakatsos-Papakostas et al. “An HMM-Based Approach for Cross-Harmonization of Jazz Standards”. In: *Applied Sciences* 13.3 (2023). ISSN: 2076-3417. DOI: [10.3390/app13031338](https://doi.org/10.3390/app13031338).
- [KGR18] Tetsuro Kitahara, Sergio Giraldo, and Rafael Ramírez. “JamSketch: Improvisation Support System with GA-Based Melody Creation from User’s Drawing”. In: *Music Technology with Swing*. Cham: Springer International Publishing, 2018, pp. 509–521. ISBN: 978-3-030-01692-0.
- [Kle05] J. A. Klein. “Pattern-Based Software Framework for Computer Aided Jazz Improvisation”. PhD thesis. RWTH Aachen, 2005.
- [KK23] Peter van Kranenburg and Eoin Kearns. “Algorithmic Harmonization of Tonal Melodies Using Weighted Pitch Context Vectors”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2023, pp. 391–397.
- [LJ04] Hong-Ru Lee and J-SR Jang. “i-Ring: A system for humming transcription and chord generation”. In: *IEEE International Conference on Multimedia and Expo (ICME)*. Vol. 2. IEEE. 2004, pp. 1031–1034.
- [Leo24] W. K. Leow. *Foundation of Media Computing*. Third Draft (to be published). 2024.

- [Lia16] Feynman Liang. “Bachbot: Automatic composition in the style of Bach chorales”. MA thesis. University of Cambridge, 2016.
- [LRL17] Hyungui Lim, Seungyeon Rhyu, and Kyogu Lee. “Chord Generation from Symbolic Melody Using BLSTM Networks”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2017, pp. 621–627.
- [LZ22] Mengqian Lin and Rui Zhao. “A Study of Piano-Assisted Automated Accompaniment System Based on Heuristic Dynamic Planning”. In: *Computational Intelligence and Neuroscience 2022 (2022)*.
- [LT12] Chien-Hung Liu and Chuan-Kang Ting. “Polyphonic accompaniment using genetic algorithm with music theory”. In: *IEEE Congress on Evolutionary Computation*. 2012, pp. 1–7. DOI: [10.1109/CEC.2012.6252869](https://doi.org/10.1109/CEC.2012.6252869).
- [LWY18] Hao-Min Liu, Meng-Hsuan Wu, and Yi-Hsuan Yang. “Lead sheet generation and arrangement via a hybrid generative model”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2018, pp. 23–27.
- [LY18] Hao-Min Liu and Yi-Hsuan Yang. “Lead Sheet Generation and Arrangement by Conditional Generative Adversarial Network”. In: *IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2018, pp. 722–727. DOI: [10.1109/ICMLA.2018.00114](https://doi.org/10.1109/ICMLA.2018.00114).
- [Lop+17] Henrique Barros Lopes et al. “Combining rules and proportions: A multiobjective approach to algorithmic composition”. In: *IEEE Congress on Evolutionary Computation (CEC)*. IEEE. 2017, pp. 2282–2289.
- [MKS16] 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*. Ed. by P. Kostagiolas, K. Martzoukou, and C. Lavranos. IGI Global, 2016, pp. 146–165.

- [Pac] “Markov constraints: steerable generation of Markov sequences”. In: *Constraints* 16.2 (Apr. 2011), pp. 148–172. ISSN: 1572-9354. DOI: **10.1007/s10601-010-9101-4**.
- [Mie99] Kaisa Miettinen. *Nonlinear multiobjective optimization*. Vol. 12. Springer Science & Business Media, 1999.
- [Mo22] Ying Mo. “Designing an automatic piano accompaniment system using artificial intelligence and sound pattern database”. In: *Mobile Information Systems 2022* (2022).
- [Niu23] Haina Niu. “Accompaniment Generation Based on Deep Learning and Genetic Algorithm”. In: *IEEE International Conference on Control, Electronics and Computer Technology (ICCECT)*. 2023, pp. 58–65. DOI: **10.1109/ICCECT57938.2023.10141257**.
- [OD01] Nicola Orio and François Déchelle. “Score Following Using Spectral Analysis and Hidden Markov Models”. In: *International Computer Music Conference (ICMC)*. Michigan Publishing, 2001.
- [PEB06] Jean-François Païement, Douglas Eck, and Samy Bengio. “Probabilistic Melodic Harmonization”. In: *Advances in Artificial Intelligence: Conference of the Canadian Society for Computational Studies of Intelligence, Canadian AI*. Vol. 4013. Lecture Notes in Computer Science. Springer, 2006, pp. 218–229.
- [PB05] Bryan Pardo and William P. Birmingham. “Modeling Form for On-line Following of Musical Performances”. In: *National Conference on Artificial Intelligence*. AAAI Press / The MIT Press, 2005, pp. 1018–1023.
- [PATW99] Somnuk Phon-Amnuaisuk, Andrew Tuson, and Geraint Wiggins. “Evolving Musical Harmonisation”. In: *Artificial Neural Nets and Genetic Algorithms*. Vienna: Springer Vienna, 1999, pp. 229–234. ISBN: 978-3-7091-6384-9.
- [PAW+99] Somnuk Phon-Amnuaisuk, Geraint Wiggins, et al. “The Four-Part Harmonisation Problem: A comparison between Genetic Algorithms

- and a Rule-Based System”. In: *AISB Symposium on Musical Creativity*. AISB London. 1999, pp. 28–34.
- [RFV13] Stanisław A Raczyński, Satoru Fukayama, and Emmanuel Vincent. “Melody harmonization with interpolated probabilistic models”. In: *Journal of New Music Research* 42.3 (2013), pp. 223–235.
- [Ran99] Don Michael Randel. *The Harvard concise dictionary of music and musicians*. Harvard University Press, 1999, pp. 10–106.
- [Rap04] Christopher Raphael. “A Hybrid Graphical Model for Aligning Polyphonic Audio with Musical Scores”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2004.
- [Rhy+22] Seungyeon Rhyu et al. “Translating Melody to Chord: Structured and Flexible Harmonization of Melody With Transformer”. In: *IEEE Access* 10 (2022), pp. 28261–28273. DOI: [10.1109/ACCESS.2022.3155467](https://doi.org/10.1109/ACCESS.2022.3155467).
- [Rob+18] Adam Roberts et al. “A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music”. In: *International Conference on Machine Learning (ICML)*. Vol. 80. Proceedings of Machine Learning Research. PMLR, 2018, pp. 4361–4370.
- [Sab+21] R. Sabitha et al. “Artificial Intelligence Based Music Composition System-Multi Algorithmic Music Arranger(MAGMA)”. In: *International Conference on Electronics and Sustainable Communication Systems (ICESC)*. 2021, pp. 1808–1813. DOI: [10.1109/ICESC51422.2021.9532706](https://doi.org/10.1109/ICESC51422.2021.9532706).
- [SOS04] Diemo Schwarz, Nicola Orio, and Norbert Schnell. “Robust Polyphonic Midi Score Following with Hidden Markov Models”. In: *International Computer Music Conference (ICMC)*. Michigan Publishing, 2004.
- [SW20] Ningning Shi and Yingfeng Wang. “Symmetry in computer-aided music composition system with social network analysis and artificial neural network methods”. In: *Journal of Ambient Intelligence and Humanized Computing* (2020), pp. 1–16.

- [SGA20] Maksym Shopynskyi, Nataliia Golian, and Iryna Afanasieva. “Long Short-Term Memory Model Appliance for Generating Music Compositions”. In: *IEEE International Conference on Problems of Infocommunications. Science and Technology*. 2020, pp. 239–242. DOI: **10.1109/PICST51311.2020.9468088**.
- [SMB08] Ian Simon, Dan Morris, and Sumit Basu. “MySong: Automatic Accompaniment Generation for Vocal Melodies”. In: *SIGCHI Conference on Human Factors in Computing Systems*. Florence, Italy: Association for Computing Machinery, 2008, pp. 725–734. ISBN: 9781605580111. DOI: **10.1145/1357054.1357169**.
- [Smi+18] Ryan Smith et al. *Hooktheory I: Music Theory. Book 1*. Hooktheory LLC, 2018. ISBN: 9780987651102.
- [SDS10] Jon Sneyers and Danny De Schreye. “AOPCALEAPS: Automatic music generation with CHRiSM”. In: Oct. 2010.
- [Spa99] Randall Richard Spangler. “Rule-based analysis and generation of music”. PhD thesis. California Institute of Technology, 1999. DOI: **10.1109/TASLP.2016.2603006**.
- [Sun+21] C.-E. Sun et al. “Melody harmonization using orderless NADE, chord balancing, and blocked Gibbs sampling”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021, pp. 4145–4149.
- [Sut+21] Phisetphong Suthaphan et al. “Music Generator for Elderly Using Deep Learning”. In: *Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunication Engineering*. 2021, pp. 289–292. DOI: **10.1109/ECTIDAMTNCN51128.2021.9425691**.
- [TT21] Senem Tanberk and Dilek Bilgin Tükel. “Style-Specific Turkish Pop Music Composition with CNN and LSTM Network”. In: *IEEE World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. 2021, pp. 000181–000185. DOI: **10.1109/SAMI50585.2021.9378654**.

- [Tem11] David Temperley. “The Cadential IV in Rock”. In: *Music Theory Online* 17 (Apr. 2011). DOI: [10.30535/mto.17.1.8](https://doi.org/10.30535/mto.17.1.8).
- [TDSG18] Lluís L. Trulla, Nicola Di Stefano, and Alessandro Giuliani. “Computational Approach to Musical Consonance and Dissonance”. In: *Frontiers in Psychology* 9 (2018). ISSN: 1664-1078. DOI: [10.3389/fpsyg.2018.00381](https://doi.org/10.3389/fpsyg.2018.00381).
- [TA99] C.P. Tsang and M. Aitken. “Harmonizing music as a discipline of constraint logic programming”. In: *International Computer Music Conference (ICMC)*. 1999.
- [Tsu+17] Hiroaki Tsushima et al. “Function- and Rhythm-Aware Melody Harmonization Based on Tree-Structured Parsing and Split-Merge Sampling of Chord Sequences”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2017, pp. 502–508.
- [WMX24] Ziyu Wang, Lejun Min, and Gus Xia. “Whole-Song Hierarchical Generation of Symbolic Music Using Cascaded Diffusion Models”. In: *International Conference on Learning Representations (ICLR)*. 2024.
- [Wan+20a] Ziyu Wang et al. “Learning Interpretable Representation for Controllable Polyphonic Music Generation”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2020, pp. 662–669.
- [Wan+20b] Ziyu Wang et al. “POP909: A Pop-song Dataset for Music Arrangement Generation”. In: *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*. 2020.
- [WC16a] Raymond P. Whorley and Darrell Conklin. “Music Generation from Statistical Models of Harmony”. In: *Journal of New Music Research* 45.2 (2016), pp. 160–183. DOI: [10.1080/09298215.2016.1173708](https://doi.org/10.1080/09298215.2016.1173708).
- [Wig+99] G. A. Wiggins et al. “Evolutionary methods for musical composition”. In: *Int. Journal of Computing Anticipatory Systems* (1999).
- [Wil+17] Duncan Williams et al. “Affective calibration of musical feature sets in an emotionally intelligent music composition system”. In: *ACM Transactions on Applied Perception (TAP)* 14.3 (2017), pp. 1–13.

- [WC16b] Yi-Chan Wu and Homer H. Chen. “Emotion-flow guided music accompaniment generation”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 574–578. DOI: [10.1109/ICASSP.2016.7471740](https://doi.org/10.1109/ICASSP.2016.7471740).
- [Yeh+21] Y.-C. Yeh et al. “Automatic melody harmonization with triad chords: A comparative study”. In: *Journal of New Music Research* 50.1 (2021), pp. 37–51.
- [Yi+22] Li Yi et al. “AccoMontage2: A Complete Harmonization and Accompaniment Arrangement System”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2022, pp. 248–255.
- [YG07] Liangrong Yi and Judy Goldsmith. “Automatic Generation of Four-part Harmony.” In: *UAI Bayesian Modeling Applications Workshop (UAI-AW)*. Vol. 268. 2007.
- [YT10] A. Yilmaz and Z. Telatar. “Note-against-note two-voice counterpoint by means of fuzzy logic”. In: *Knowledge-Based Systems* 23.3 (2010), pp. 256–266.
- [Yin+21] Zongyu Yin et al. ““A Good Algorithm Does Not Steal–It Imitates”: The Originality Report as a Means of Measuring When a Music Generation Algorithm Copies Too Much”. In: *Artificial Intelligence in Music, Sound, Art and Design (EvoMUSART)*. Springer. 2021, pp. 360–375.
- [Yin+23] Zongyu Yin et al. “Deep learning’s shallow gains: a comparative evaluation of algorithms for automatic music generation”. In: *Machine Learning* 112.5 (2023), pp. 1785–1822.
- [ZL21] Te Zeng and Francis C. M. Lau. “Automatic Melody Harmonization via Reinforcement Learning by Exploring Structured Representations for Melody Sequences”. In: *Electronics* 10.20 (2021). ISSN: 2079-9292.
- [ZX21] Jingwei Zhao and Gus Xia. “AccoMontage: Accompaniment Arrangement via Phrase Selection and Style Transfer”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2021, pp. 833–840.

- [ZXW22] Jingwei Zhao, Gus Xia, and Ye Wang. “Domain Adversarial Training on Conditional Variational Auto-Encoder for Controllable Music Generation”. In: *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*. 2022, pp. 925–932.
- [Zha+24] Jingwei Zhao et al. “Structured multi-track accompaniment arrangement via style prior modelling”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 37 (2024), pp. 102197–102228.
- [Zhu+18] Hongyuan Zhu et al. “XiaoIce Band: A Melody and Arrangement Generation Framework for Pop Music”. In: *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. ACM, 2018, pp. 2837–2846. DOI: [10.1145/3219819.3220105](https://doi.org/10.1145/3219819.3220105).
- [Zhu+20] Hongyuan Zhu et al. “Pop music generation: From melody to multi-style arrangement”. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 14.5 (2020), pp. 1–31.