N. d'ordre : XX

Université de Lille

Thèse

pour obtenir le grade de :

Docteur de l'Université de Lille

dans la spécialité

Informatique et Applications

par

Francesco Maccarini

Modeling Orchestration for Computer Assisted Analysis and Human-AI Co-creativity

Soutenance prevue le 20 décembre 2024 devant le jury composé de :

UMR 9189 CRIStAL
Université de Lille - Campus scientifique
Bâtiment ESPRIT, Avenue Henri Poincaré, 59655 Villeneuve d'Ascq
France

Contents

Abstract

Orchestration is the art of composing for an ensemble of instruments, involving the blending and contrasting of instrumental timbres to create a cohesive orchestral sound. It requires both high-level artistic planning of musical ideas and precise, fine-grained control over technical details. Each part must be playable on its specific instrument and fit well with the rest of the ensemble.
This thesis aims at formalizing and expanding the knowledge of orchestration by describing it through the lens of mathematics and computer science, while proposing models and tools for the computational analysis of orchestral music and human-machine co-creative orchestration.
A key focus of this thesis is the concept of *orchestral texture*, which refers to the roles, functions, and combinations of the instrumental parts within a composition. We introduce three abstract models of orchestration, release a multi-modal corpus of annotated orchestral scores, and propose a framework for computer-assisted orchestration, which has been successfully applied in a co-creative orchestration project.

Resumé

L'orchestration est l'art de composer pour un ensemble d'instruments, impliquant le mélange et le contraste des timbres instrumentaux pour créer un son orchestral cohérent. Elle exige à la fois une planification artistique de haut niveau des idées musicales et un contrôle précis et minutieux des détails techniques. Chaque partie doit pouvoir être jouée sur l'instrument qui lui est propre et s'accorder avec le reste de l'ensemble.
Cette thèse vise à formaliser et à élargir les connaissances sur l'orchestration en la décrivant sous l'angle des mathématiques et de l'informatique, tout en proposant des modèles et des outils pour l'analyse informatique de la musique orchestrale et l'orchestration co-créative homme-machine.
L'un des points clés de cette thèse est le concept de *texture orchestrale*, qui fait référence aux rôles, fonctions et combinaisons des parties instrumentales au sein d'une composition. Nous présentons trois modèles abstraits d'orchestration, nous publions un corpus multimodal de partitions orchestrales annotées et nous proposons un cadre pour l'orchestration assistée par ordinateur, qui a été appliqué avec succès dans un projet d'orchestration co-créative.

Acknowledgments

XXX

Villeneuve d'Ascq, 20 décembre 2024

1. Introduction

The Grove music dictionary defines orchestration as *"the art of combining the sounds of a complex of instruments (an orchestra or other ensemble) to form a satisfactory blend and balance"* [145]. Orchestration is much more than a mere distribution of the voices among the instruments, it involves blending or opposing the sounds of a possibly large ensemble to create an *orchestral texture*. One may argue that, when put in the context of the orchestra, the instruments partially lose their individuality, and the entire orchestra should be considered as one big instrument. The high variety of the possibilities offered by the individual instruments and their combinations gives to the "orchestra-as-an-instrument" potentially infinite sounds and timbres. In 1844, Berlioz wrote in its *Grand traité d'instrumentation et d'orchestration modernes* [19, p. 240 of the 1852 translation]:

> *" The orchestra may be considered as a large instrument capable of uttering at once or successively a multitude of sounds of different kinds; and of which the power is mediocre or colossal according as it employs the whole or a part only of the executive means belonging to modern music, and according as those means are well or ill chosen and placed in acoustic conditions more or less favorable. "*

A composition can be directly written for the orchestra, or alternatively *composition* and *orchestration* can be considered as two separate tasks.
In large productions for the movie industry, for example, it is common to have a lead composer that writes the main thematic material and a team of *orchestrators* that, supervised by the lead composer, take care of the arrangements that have to appear at different moments of the movie [235]. In any case, the artists have to pay attention to the characteristics of the individual instruments, what each of them plays, and that the overall combination creates the desired *"orchestral texture"*.
This complexity is the source of big challenges for the creators, and for theorists who want to formalize and teach such topics. As for other topics in music theory and composition, the formalization and teaching of instrumentation and orchestration have improved over time, starting from the 19th century. Various orchestration treatises include works by Rimsky-Korsakov, Forsyth, Koechlin, Piston, McKay [223, 82, 142, 211, 182], and more recently Adler [4]. They cover topics such as *organology* and *instrumentation*, detailing the musical capabilities of each individual instrument and their combinations to shape the sound of the orchestra. The most recent treaties also add substantial sections on combining instruments and describing perceptual effects stemming from these combinations. Music conservatories and music departments now offer dedicated orchestration classes.

The goals of this thesis are to expand the knowledge on orchestration making a step towards a better formalization of this art, and to design tools for the analysis and co-creation of orchestral music. We address these problems by exploring and proposing models to describe orchestration with the language of mathematics and computer science. A particular attention is devoted to the elusive notion of *orchestral texture*, which describes both the combinations of the instruments used and the different techniques with which each instrument is played. We use a variety of methods, including machine learning, always aiming at explicability and interpretability for the users, in the context of explainable and human-centered artificial intelligence.

After this introduction and Chapter 2 presenting the state of the art, this thesis is divided in three parts with two chapters each.

Part I: Modeling Orchestration While formal models to describe harmony in tonal music (like roman numerals) are well established in classical music theory, other important factors of symphonic music like texture and orchestration are mostly transmitted from master to apprentice without a real formalization. Building on concepts from music theory and perception we propose three abstract models of orchestral texture and orchestration (Chapter 3) and a framework for co-creative interaction in orchestration (Chapter 4).
The first model introduced in Chapter 3 is a *taxonomy of orchestral texture* to systematically describe orchestration in the western classical style with a language inspired by information science, using layers with roles and relations. In this chapter, we show its descriptive power, but also the ambiguities that emerge. The second model is an abstract version of a piece representing an orchestral composition "before orchestration" that we call *layer score*. It represents the information about notes and layers without the instrumentation. The third model is called *orchestration plan* and it can be used to prescribe the instrumentation of a piece. The main contribution from this chapter is the formalization of existing concepts into these three objects.
Two main contributions are presented in Chapter 4. The first one is a table reorganizing existing Music Information Retrieval (MIR) orchestration tasks in light of the three abstract models proposed, and identifying new tasks. The second one is a process for computer-assisted orchestration that we propose. It uses layer scores and orchestration plans as intermediate objects in a process with clear steps and tasks allowing humans and machines to collaborate in orchestrating an existing piece of music.

Part II: Orchestral Music Analysis The analysis of orchestration and orchestral texture with computational tools can be facilitated by open corpora. We published with open data licenses a corpus of first movements of classical and early-romantic symphonies annotated following the taxonomy of orchestral texture. The corpus, which is a contribution from this thesis presented in Chapter 5, has later been enriched with public domain recordings synchronized to the score, and can be explored with Dezrann[1], a

[1] `https://www.dezrann.net/`

web platform for music annotations developed by the Algomus team. Layer scores have also been automatically created starting from the annotated orchestral scores.
In Chapter 6 we propose a data driven method based on machine learning to detect orchestral blends in music scores. Instrumental blending is a fundamental process in music perception and stands as the basis of all further cognition of orchestral music. This preliminary study offers as contributions the engineering of score-based features that represent cues affecting instrumental blending, and their use in a model that matches the results of the state of the art.

Part III: Orchestration and Co-creativity In the last part of the thesis we explore co-creativity in orchestration applying our framework.
In Chapter 7 we present the orchestration of two pieces for piano solo from the suite *Angeles* by Gissel Velarde that we have created in collaboration with Mael Oudin. This project is the first real use case application of our framework for co-creative interaction in orchestration, and has resulted in several scientific and artistic contributions: we have developed a first model to generate orchestration plans, we have collected feedback from the human artists involved, and we have produced orchestral scores for the two pieces that have been played in a live performance by Orquesta Kronos conducted by Andrés Guzmán-Valdez.
In Chapter 8 we report a series of experiments about the generation of orchestration with Transformers. The main contribution from this chapter is a tokenization method to represent orchestral scores, texture annotations, layer scores, and orchestration plans. The preliminary results of these experiments are encouraging but not conclusive, suggesting that additional research is necessary to develop and train a fully operational orchestrator Transformer.

We then conclude this thesis with Chapter 9, where we summarize the results and we indicate the way for future research.

2. State of the Art

This chapter is a literature review of studies on orchestration and Music Information Retrieval (MIR) methods that are relevant to this thesis. It is divided into 5 sections. We start by discussing orchestration in relation to other music research disciplines (Section 2.1); then we introduce various symbolic formats that are used to represent music on computers for computational music analysis and generation (Section 2.2); we continue by surveying studies in computational music analysis that are relevant for orchestration (Section 2.3); after we present music generation models in relation to orchestration (Section 2.4); and finally we illustrate publicly available corpora of orchestral music (Section 2.5).

2.1. What is Orchestration?

2.1.1. Orchestration and Texture

Modeling orchestral scores is undeniably linked to the understanding of conventional musical parameters such as melody, rhythm, harmony, and form. The elusive notion of musical *texture* also plays a crucial role. The term texture has started to proliferate in music writings starting from the middle of the 20^{th} century. Its use became very widespread in English texts, while remaining very marginal in other European languages, where a direct translation does not exist or already yields another meaning (for example the Italian word *tessitura* is used to describe the range of an instrument or vocal part) [71]. The term has been employed in a metaphorical sense with regard to its usage in the visual arts, particularly painting and sculpture, but without a clear definition of its meaning in the context of music. In its orchestration treaty, Piston describes seven categories of texture: orchestral unison, melody and accompaniment, secondary melody, part writing, contrapuntal texture, chords, and complex textures [211, 89]. Complex variants of the discussed textures can be created through their combinations [182]. This 'applied' theory of texture has utility for students learning to reproduce a particular style. However, the number of categories tends to increase as composers invent new textures [71].
In his study from 1960, Nordgren gives a more quantitative description of orchestral texture [203]. He compares "textural patterns" in romantic symphonies, by modeling eight textural parameters and their evolution: instruments number, range, register, and spacing, the proportion and register of the gaps, doubling concentrations and the register of concentrations. The study identifies several characteristics of the writing

style of Beethoven, Mendelssohn, Schumann, and Brahms[1]. Along the same line Rowell [232] proposed an expansive and elaborate description of texture with eight "textural values": orientation (vertical or horizontal), tangle (the interweaving of melodies), figuration (the organization of music in patterns), focus vs interplay, economy vs saturation, density (as opposed to thinness), smoothness (as opposed to toughness), and complexity (as opposed to simplicity).
Texture cannot exist in isolation; rather, it serves an important role in connecting the various elements of music. J. Levy [155] illustrates this concept through numerous examples, demonstrating how texture in classical and early Romantic music can signal important structural changes by utilizing accompaniment patterns, solos, and unison to capture the listener's attention. Levy emphasizes that while texture alone may lack meaning, it is used to enhance the meaning carried by other musical elements.

It is nowadays widely accepted to describe texture in western classical music with four terms: *monophony* (single melodic line), *polyphony* (interweaving melodic lines), *heterophony* (melodic line with simultaneous variations), and *homophony* (melodic line and chordal accompaniment) [71, 18]. It should be noted that the boundaries between these categories are blurred, with music styles existing in between them. Huron [122] describes them as being embedded in a texture space along two axes, that he calls *onset synchronization* and *semblant motion* (see Figure 2.1). The first refers to the number of elements that are occurring simultaneously, while the latter to the degree of homogeneity or heterogeneity between these elements. Those dimensions have also been called *density* (or *volume*) and *diversity* [49].
Moreira De Sousa uses the term *texture space* in a distinct manner. He suggests the adoption of three texture spaces combined with mathematical operations to describe a texture at varying levels of detail. This method can be applied for composition [58].
Texture can further be characterized by describing the simultaneous textural elements that constitute it and the similarities and differences between them. In the description by Benward and Saker [18], each of those simultaneous parts (or *layers*) serves a different purpose and role.

- *Primary melodies* are the most important lines in a musical texture. In polyphonic textures there may be several primary melodies. It is usually found as the highest part in a composition but it does not need to be.

- *Secondary melodies* are other melodic lines that do not have the same importance as the primary melody. The decision whether a melody is primary or secondary is not trivial and it is often a matter of debate.

- *Parallel supporting melodies* are melodies that are similar in contour to a primary or secondary melody. They might maintain a constant interval with the melody

[1]Some of Nordgren's observations are unexpected. For example it was found that Brahms employs textures with narrower range (depending on the interval between the lowest and highest tones) than the other composers. The impression of a wide sonority range in Brahms, however, is explained by his frequent shifting of register.

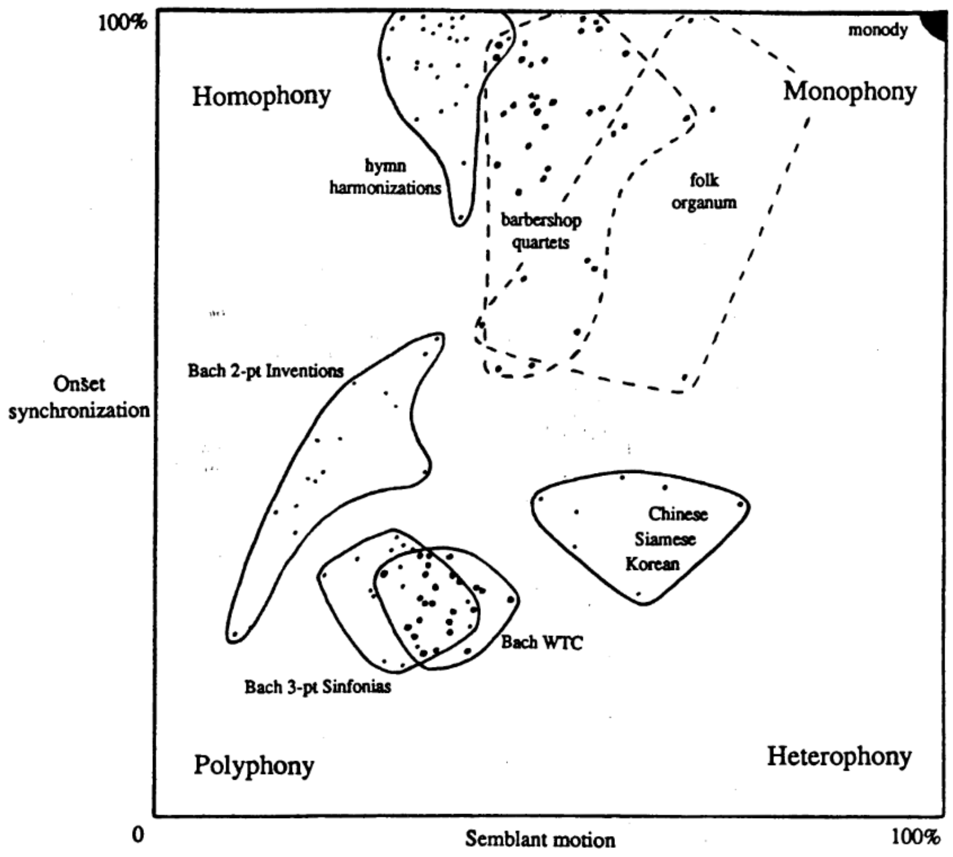


Figure 2.1.: Huron (image reproduced from [122]) represents the four characteristic textures (Homophony, Monophony, Polyphony, and Heterophony) in a texture space along two axes, which can be described as *onset synchronization* (or *density/volume*) and *semblant motion* (or *diversity*). Different styles of music find their place in this texture space.

they support. A typical example is a part moving a third or a sixth below the main melodic line.

- *Static support* parts are of two types: sustained tones or chords, which are often pedal notes, and repeated melodic and rhythmic figures or ostinati.

- *Harmonic support* and *rhythmic support* elements typically serve the role of accompaniment in an evolving homophonic texture. They are often combined, but they can be separate. They are not to be confused with static support, which is used with constant harmony or to highlight unchanging pedal tones in an evolving harmony.

2.1.2. Timbre and Orchestration in Music Perception

In orchestral music, texture is highlighted by the use of different instruments, that can be more or less similar in timbre. Timbre is a characteristic that is rooted in the physical properties of the instrument emitting the sound and on the medium of transmission, but it is a fundamentally perceptive phenomenon. In Music Perception and Cognition, studies on orchestration have focused on the processing mechanisms of sounds, and on the essence of timbre. While orchestration has long been considered a purely artistic practice ("art and not a science" [211, p. 356]), perception scientists have begun advocating for the integration of principles from perception studies into music theory and composition practice, arguing against dismissing scientific approaches to studying orchestration as completely unattainable or undesirable. In 1979 McAdams and Bregman [176] suggested that

> *" composers and music theorists should thoroughly examine the relationship between the "musical" principles they use and espouse, and the principles of sensory, perceptual, and cognitive organization that operate in the human auditory system. [. . .] To ignore the evidence from the real life system in developing a theory of music or a musical composition is to take the chance of relegating one's work to the realm of what might be termed "paper music." "*

Since then, perceptual studies have shown how science can be useful to composers and orchestrators. Studies have focused on how instruments combination affects timbral response, on *blending* qualities of instruments (based on their own and combined timbres), and on how composers can make use of timbral consonance and dissonance to create new sounds [139, 234, 238]. Researchers have attempted, using statistical techniques such as multi-dimensional scaling, to embed instruments and sounds in a *timbre space* where the dimensions are human-friendly descriptors [276, 130, 72]. Most of these descriptors remain correlated to quantifiable acoustic measures [177]. Using this space it is possible to draw trajectories between sounds, to browse instruments more efficiently, and, when linked to a synthesis model, to perform sound morphing [224, 76, 137]. Thanks to these studies the importance attributed to timbre has shifted from being relegated as a secondary parameter in western music culture [268] to being considered and studied as a "structuring force in music" [64, 175].

Music perception has studied the auditory grouping mechanisms related to orchestral music. Those phenomena are correlated to certain relations between the spectra of the instruments (timbre) [152], to the organization of notes in the vertical (harmonicity and parallelism) and time axes (synchrony) [24, 62], but also to performance related parameters, such as room acoustic, spacial position of the sound sources, and musicians' intentions [57, 151]. They are at the basis of the formation of music streams[2] and orchestral textures [91].
Three orchestra related auditory grouping mechanisms have been studied by music

[2] *Stream* is a term that is used to describe a group of instrumental parts that are perceived as a whole [31]. We discuss this concept more in detail in Section 2.3.1

perception researchers and are detailed by McAdams, Goodchild, and Soden in the Taxonomy of Orchestral Grouping Effects (TOGE) [178]:

- *Concurrent grouping* is about the blending or contrast of instrumental sounds. Blends can be of two types: timbral augmentation, when the sound of one instrument is modified by the timbre supporting instruments, and timbral emergence, when a new timbre emerges from the mixture. All concurrent grouping phenomena can be punctuated or sustained, and, if sustained, they can be stable or transforming.
- *Sequential grouping* connects these events in time (by integration or segregation) and gives raise to the perception of different instrumental streams.
- Finally, *segmental grouping* acts on a higher level by enhancing the perception of musical structure through orchestration variation in time (timbral contrasts).

F. Levy proposed functional orchestration [164], an alternative analytic framework. In functional orchestration, orchestral music is described through *functions* (the musical aim or goal), *techniques* and *effects* (the perceptual aim or result). A study comparing analyses of the same piece (Ravel's *Alborada del Gracioso*) with different taxonomies, among which the TOGE and functional orchestration, has been carried out in the ACTOR[3] project [277].

Some tools have been specifically developed to study orchestration and music perception. The Timbre Toolbox [210] is a library allowing to easily compute a set of acoustic descriptors related to timbre. OrchView[4] is a software to annotate scores, supporting the TOGE. Orcheil[5] is a web application conceived and developed by Dolan and Newsome to visualize graphically the orchestration of a piece [64]. A visualization of the first movement of Beethoven 9th Symphony produced with this tool is displayed in Figure 2.2.

2.1.3. Orchestration is a Multi-Scale Problem

Orchestral music is a rich and complex art form that captivates audiences with its intricate interplay of melodies, harmonies, textures, timbres, and effects, that together contribute to the emergence of the *sound of the orchestra*. To achieve a successful orchestration, it is essential to master the balance between low level choices (like the voicing and instrumentation of a certain harmony or the playability and coherence of a single

[3]ACTOR (Analysis, Creation, and Teaching of ORchestration) is a 7-year project (2018-2025) funded by the Social Sciences and Humanities Research Council of Canada (SSHRC). Its aim is to bring timbre and orchestration to the forefront of scholarship, practice, and public awareness through collaborations among world-class artists, humanists, and scientists. See `https://www.actorproject.org/about-actor`. I have joined the project as a student member during my first year as a PhD student, and I have participated in the three last editions of the summer workshop. In the the edition of 2024 I have presented my work in the plenary session.

[4]`https://www.actorproject.org/workgroups/orchview`

[5]`https://orcheil.ca/`

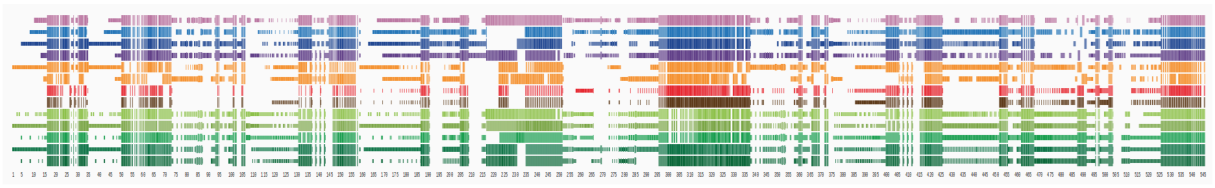


Figure 2.2.: First movement of Symphony No. 9 by Beethoven analyzed by Orcheil. Each colored row represents an instrument of the orchestra. Instruments are displayed in the order in which they appear on the first page of the score, from top to bottom. Bigger rectangles correspond to the segments in which the instrument is playing with louder dynamics.

part) and high-level composition goals (including desired perceptual effects). Any attempt to perform orchestration, or to understand it and explain it, must deal with the fact that orchestral music has a delicate hierarchical and multi-dimensional structure.

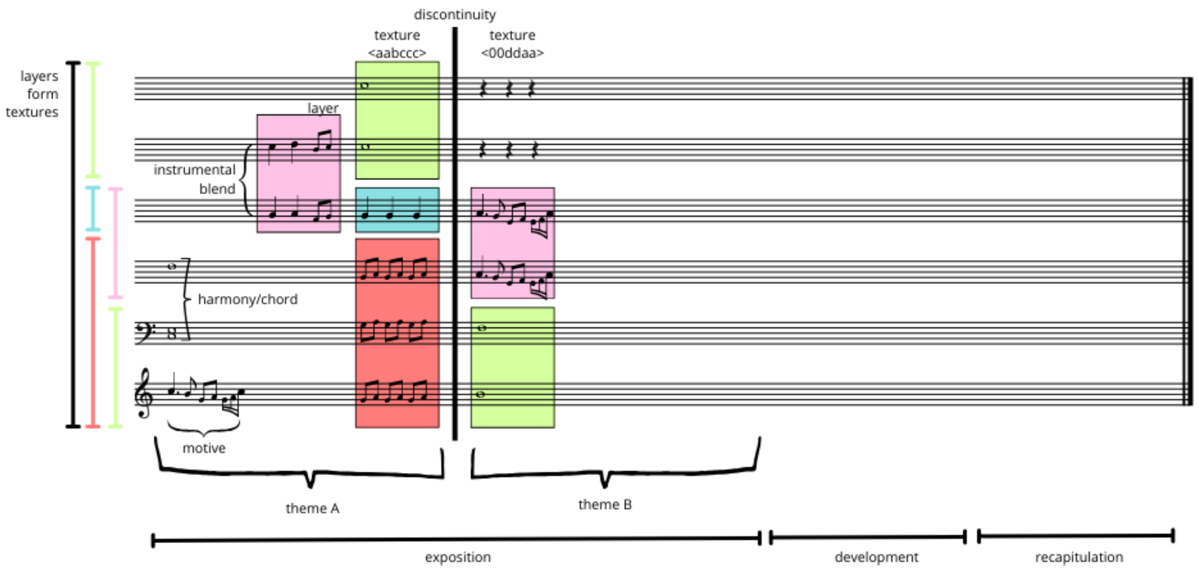


Figure 2.3.: Orchestral music is organized with horizontal and vertical hierarchical structures. Along the time dimension, notes are combined in motives and themes; then motives and themes are combined to structure music, as here in the the sonata form. Along the vertical dimension, note pitches are combined into harmonies and chords, note timbres are combined into blends. Blends give raise to streams and layers, which are juxtaposed to create different textures. On the largest scale, textural and timbral discontinuities highlight the form, as here between theme A and theme B.

In orchestral music, hierarchical structures develop along a horizontal (temporal) and a vertical (harmony and instrumentation) dimension, as illustrated by Figure 2.3. The horizontal and vertical structures are intertwined, and interact at different scales. Some

of these structures are common to other styles of music, while some others are related to orchestration concepts like texture (Section 2.1.1) and timbre (Section 2.1.2), and are specific to orchestral compositions.
The smallest unit is a note, with its pitch and duration, but also its timbre. Notes are stacked with their pitches along the vertical dimension to form harmonies and chords. In orchestral music, the different timbres of the instruments playing the notes are also involved. As observed by McAdams et al. [178], when different sounds are superimposed, their timbres can merge to form one orchestral blend or not. We have seen in Section 2.1.2 that this phenomenon is at the lowest level of the hierarchical organization of perceived orchestral grouping effects. At a higher level, instrumental blends which are synchronous and have some degree of parallelism form streams and layers by sequential grouping (more on this is presented in Section 2.3.1). The nature of layers and the way in which they are combined on the vertical dimension constitute different orchestral textures. We have seen that texture is an elusive concept, that is difficult to define. Nevertheless attempts to analyze it computationally have been made, and will be presented in Section 2.3.2.
Along the time dimension notes are combined to form motives and themes. On a larger scale, motives and themes are further arranged in time to produce a structure, such as that of the sonata form (exposition-development-recapitulation) that we typically find in first movements of classical sonatas, but also of symphonies [39].
The interconnection between the vertical and horizontal dimensions happens at the lowest scale in the formation of streams and layers from the temporal succession of synchronous notes of blending instruments. Another interaction between the two dimensions happens at a larger scale: the structure and form of a piece are constructed horizontally by combining motives and themes, and are highlighted by the choice of the instruments' combinations and other vertical characteristics like texture and timbre. In the orchestral graph displayed in Figure 2.2 we can see a visual representation of the instrumentation of the first movement of Beethoven 9^{th} Symphony. The structure of the piece can be observed in the graph, that is constructed using only the information about the instrumentation and the dynamics of the piece.

We have seen that orchestral music is a rich, multidimensional art form. According to one's taste and training, the attention can be captured by structures at different scales, both during listening and analysis. Focus may fall on intricate details, like a motive or a particular sound, or on the overall form of the piece. Understanding orchestration involves studying how these hierarchical structures interact both horizontally (over time) and vertically: how the harmonious blending of instruments gives rise to distinctive layers, how the juxtaposition of those layers crafts diverse orchestral textures, and how the organization of textures over time, produces the segmentation of the composition into distinct sections, ultimately culminating in the creation of a cohesive narrative. We study some of these interactions in this thesis: we model the vertical organization of instrumental parts in orchestral layers and textures in Chapter 3. Their temporal organization is also described by the model. We do not directly study timbre itself, but we model the results of timbral interactions that originate orchestral blends in Chapter 6.

On the largest scale, a section is devoted to the interaction between texture and form in Chapter 5.

2.2. Symbolic Music Representations

Western classical orchestral music is historically notated in music scores with many staves. Digital representations of music scores come in different formats, which are not completely equivalent, as the amount and type of information contained in one format might differ from the other ones. We expect that the essential information is conveyed by most representations, but still it is essential to chose the right format for the specific application. However some formats are more widespread than others and some compromises in terms of format are sometimes necessary to collect enough data. In this section we discuss the representations that were used in this thesis, without any claim of being exhaustive.

Standard MIDI Files The MIDI (Musical Instrument Digital Interface) standard[6], which was introduced in 1983, constituted a revolutionary development in the field of electronic music, establishing a unified communication protocol for digital instruments and computers. Prior to the advent of MIDI, electronic instruments manufactured by different companies were unable to interact in a seamless manner, which is the problem addressed by MIDI. MIDI transmits standardized messages that direct instruments on performance details such as note events (which indicate for example the onset or release of a note, the pitch, and the velocity), control changes (which modify parameters such as volume, modulation, and expression), program changes (which switch between different instrument sounds), and system messages for synchronization and setup [160, 192]. In addition to the protocol, the MIDI file format (.mid) was developed to digitally store these musical instructions. MIDI files comprise a sequence of MIDI messages arranged on a timeline, representing musical performance data. This digital score can be interpreted and played back by any device or software that is compatible with the MIDI standard. While the MIDI protocol ensures real-time communication between devices, the MIDI file format allows for the preservation and manipulation of musical data, which is used for a variety of purposes, including composing, arranging, and transferring music across different platforms.
Although MIDI was originally developed as a communication protocol between devices, MIDI files have since become one of the most popular formats for exchanging symbolic music data online and in MIR datasets. However, the information they contain cannot be entirely related to that of a printed score.

Score Representations: **kern, MusicXML, and MEI In comparison to MIDI, score representations incorporate additional features pertaining to the music score, while simultaneously losing features related to performance. For instance enharmonic equivalent notes such as C♯4 and D♭4 are distinguishable in a score representation, and

[6]`https://midi.org/`

expressive indications are marked in a manner analogous to a printed score. Conversely, MIDI does not permit the distinction of enharmonic equivalent notes but does allow for precise control of expression at each instant [197]. We present here three different formats for score representation.
The **kern music format[7] is an encoding system proposed by Huron [123] in 1997 and developed as part of the Humdrum Toolkit for representing musical scores. It uses symbolic notation to capture musical elements such as pitches, rhythms, articulations, and dynamics. Additionally, it supports polyphonic music by aligning multiple voices in parallel columns. The **kern format is flexible and extensible, allowing for various annotations and metadata. It is a plain-text encoding system, and it is primarily used in music research and analysis due to its readability and ease of processing.
The MusicXML[8] format, first release in 2001, was developed with the intention of establishing a universal standard for music scores [90]. Based on the Extensible Markup Language (XML), it is designed to serve as a music engraving software capable of representing the full range of visual elements present in a music score, including clefs, time signatures, notes, rests, accidentals, and dynamics.
The Music Encoding Initiative (MEI)[9], started in 2002, offers a distinct standard for musical scores. While MEI and MusicXML both encode music notation elements such as notes, staves, rests, and clefs in XML format, they are based on different philosophies. MusicXML was first intended to be used to share music content between music editors and it is therefore primarily focused on representing the visual aspects of a score. In contrast, MEI provides similar functionality for page layout but also focuses on encoding the intellectual content and structure of the music notation, meaning that it directly represents the musical relations between the objects of a score [228]. MEI supports various notation systems beyond the standard Common Western Notation, including mensural (Renaissance-era) and neume (Medieval) notations. MEI does not only represent the visualization for these notations, but it preserves their structure and semantics.

Music Representations for Machine Learning: Piano Roll and Music Tokens In many cases, MIDI files and digital scores are not directly suitable for Music Information Retrieval (MIR) applications employing machine learning models. Instead, lower-level representations are necessary. Two primary representations used in music machine learning models are the piano roll and sequences of music tokens. Both of these representations can be derived from MIDI files or digital scores. The selection of a representation is closely aligned with the choice of the machine learning model.
Digital piano rolls are inspired by piano rolls for player piano (see Figure 2.4). Those were perforated papers, that were controlling a mechanical system. The paper moves through a system that, in correspondence of a hole, activates a mechanism that plays a note on the piano. The note is then released at the end of the hole [241]. Some holes are dedicated to control expressive performance, but in their most basic form

[7]`https://www.humdrum.org/`
[8]`https://www.musicxml.com/`
[9]`https://music-encoding.org/`

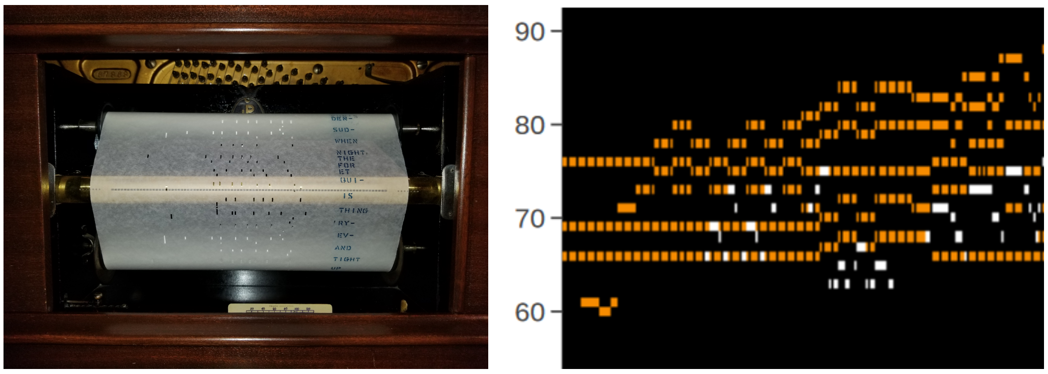


Figure 2.4.: On the left, a piano roll of *Parade Of The Wooden Soldiers* by Leon Jessel is being read by a performer piano. Image by Draconichiaro - Own work, CC BY-SA 4.0, `https://commons.wikimedia.org/w/index.php?curid=82604752`. On the right, a digital piano roll from Liszt's *Ihr Glocken von Marling*, reproduced from [244].

each hole track corresponds to a note. In computer music, a piano roll is represented as a matrix with a pitch and a time dimension. Non-zero values are usually associated to the presence of a note at the given time and pitch. Different variants have been proposed to extend the encoding to multi-track music, to allow the representation of velocity, and to better distinguish between repeated and sustained notes. For example, Nabeoka et al. [198] represent a multi-track score as a collection of matrices, with two matrices for each instrument: one encoding the onset times (when a note starts) and the other encoding the activation times (when a note is playing). Piano roll representations are analogous to the encoding of images and have been used successfully with models adapted from computer vision [244, 27, 51].
The representation as sequences of music tokens is influenced by the success of Large Language Models (LLMs) in Natural Language Processing (NLP). In this approach, music is encoded in a textual format through a sequence of word-like tokens derived from a predefined dictionary. Numerous studies have highlighted the similarities between music and language, suggesting that text-based models can be effectively adapted for musical applications. However, several critical differences distinguish symbolic music from text and language. Music is inherently structured in the time dimension with precise rhythms; it can be polyphonic, featuring simultaneous notes; it employs a multimodal notation system that includes various symbols (such as notes, measures, dynamics, and tempo); it is challenging to segment objectively; and it lacks a universally accepted "grammar" [148]. Different tokenization strategies have been developed by researchers to represent music in a textual format [83] (see Figure 2.5 for two examples). Some of these representations directly replicate a subset of the messages found in a MIDI file [204, 102]. However this approach is problematic since it results in excessively long sequences that exceed the context window of certain models. Additionally, generative models must learn to produce valid MIDI files, ensuring, for example, that every note onset is matched with the corresponding note offset. Other

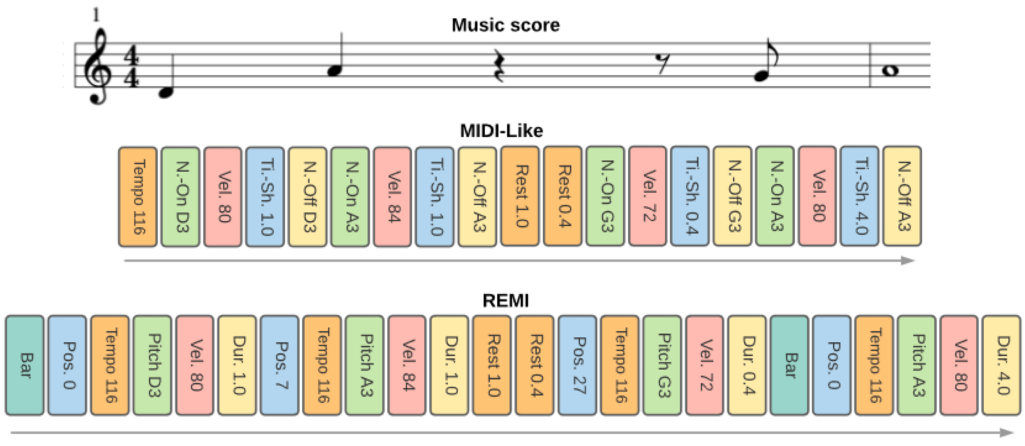


Figure 2.5.: Example of tokenization with two different strategies: MIDI-Like (directly replicating MIDI messages) and a REMI (encoding time with bars and positions) . Figure reproduced from [83].

representations try to group together MIDI messages that correspond to a musical unit of meaning [121, 116, 282, 220]. The choice of representation significantly influences the outcomes of learning processes, but no single representation is universally superior; the optimal representation depends on the specific task at hand [84].

2.3. Computational Music Analysis and Orchestration

In this section we introduce the research field of Computational Music Analysis and the recent advancements in topics related to orchestration. Computational Music Analysis (CMA), Computational Musicology (CM), and Music Information Retrieval (MIR) are related fields of research that aim at extracting information from music through algorithms. Even though their end goal might be different, they have a wide area of superposition in the data and methodologies used. There is also a certain overlap of research goals and interests with “traditional” musicology and music analysis, even though the disciplines remain separate to this day.

Music Analysis According to Bent’s definition from the Groove Music Dictionary, an analysis is an attempt to answer the question ‘how does it work?’ [17]. Therefore, it cannot be just a mere description but it must be an explanation of a piece of music [172]. In the Western classical tradition, music notation plays an important role in the creation and analysis of music. Computational approaches are frequently also based on scores. However, a variety of perspectives on the ontology of a *piece of music* exists and the identification of a score with *the music itself* is debatable [246, 43, 172].
We are aware of these problems, but we do not treat them in this thesis. We choose instead to adopt an analytical approach to music that is based only on the score itself and that is agnostic to the ontology of a piece of music. This is the proposal of Nattiez,

which he referred to as the "neutral level". An analysis should be considered neutral in the sense that it does not attempt to study the composer's intentions or the listener's cognitive mechanisms and emotions [201]. The concept of neutrality has been heavily criticized, with the argument that the analyst's perception and cultural background cannot be completely disregarded [191]. Despite the persistence of biases in algorithmic design, some authors have argued that computational methods can facilitate an analysis that is closet to this concept of neutrality and scientific rigor [8].

Computer Science and Music Sentiment with respect to employing computational tools to analyze music has been and still is divided [74, 43, 170]. The symbiosis between computer science and musicology remains relatively uncommon in contemporary academic circles [22]. This can be partially explained by the skepticism of the traditional musicological community towards digital humanities [147], and by the fact that computer scientists often lack a musicological background and encounter difficulties in identifying relevant musicological inquiries to pursue [269]. Mor et al. [193] conclude their review by urging musicologists to develop more formalized theories of music, ready to be implemented in a computer program. However, we believe that it is already feasible for the MIR community to adapt existing theories to the mathematical formalism required by computational methods. This is the approach we take in this thesis to formalize and describe orchestral texture.

Goals and challenges of Computational Music Analysis Anagnostopoulou and Buteau discuss four aims of Computational Music Analysis [8]. One or more of these may motivate different researchers. The primary goal of CMA is to discover musicologically interesting results. Second, the formalization of a music task is an important result in itself. For example, researching encoding languages and representations for music data is a valid and essential topic of research [269]. Third, the goal could be to assist an analyst in studies that require intensive computation. A fourth goal is to test computational methodologies in a complex and unusual abstract domain such as music. This last goal introduces the risk of a system that has its end in itself but is of no interest for other researchers [74]. Indeed, a correct computational analysis that can match the performance of a human expert with no added value is somehow 'useless' [172].
Therefore, the challenges of computational music research come from the interdisciplinary nature of the subject, with the difficulty to combine the rigor of informatics with music [8]. The methods employed must address the multirepresentational, multicultural and multiexperiential nature of music [70]. It is essential that experiments are designed in a way that allows the deduction of valid conclusions without the use of extrapolation [251].

Examples of Computational Music Analyses Computational Music Analysis has used a variety of different methods and approaches. Some studies have tried to adapt existing reductionist theories such as Schenkerian analysis or Lerdahl and Jackendoff's Generative Theory of Tonal Music [154] to a computational framework, using hierarchies and generative grammars [171, 227, 1, 104]. Along the same line, other studies have focused on structure and hierarchy in music [28, 55] and have used graphs to represent music [174, 245, 136]. Computation can be used as an interactive supportive tool in a human-centered study [225]. Recently, approaches based on deep learning and pretrained models have started to appear [41, 282]. For an extensive list of studies we refer to [193]. In the next sections we present the main research topics in CMA related to orchestration.

2.3.1. Models and Algorithms for Voices, Streams, and Layers

The identification and separation of different voices is very relevant to the study of orchestral music. A seminal study on the problem is that of Cambouropoulos [31], that starts by surveying the ambiguous uses of the term 'voice' in different music contexts. He provides definitions and a few examples to better explain the concepts, and only later proposes an algorithm to computationally analyze voices according to the given definitions. Three main meanings are identified (see also the example in Figure 2.6).

- *Literally instrumental 'voice'*: the sound sequences produced by different musical sound sources such as individual choir voices or instrumental parts,
- *perceptual 'voice'*: musical streams in which multiple sound events are grouped together by perceptual principles (see Section 2.1.2), and
- *harmonic 'voice'*: voices implied by the voicing of the underlying harmony.

We will avoid using the ambiguous term "voice" alone in this thesis. We are mainly interested in the second definition that we are going to call stream or *layer*. In orchestral music a layer is composed of one or more instrumental *parts* (first definition), but its composition is not fixed and may vary in time. We propose a formalization of these concepts in Section 3.1. We will rarely refer to the third definition.

Two main tasks related to voices have been proposed and studied: voice separation and melody identification in polyphonic music. The problem of voice separation consists in dividing the notes of a given score or MIDI file into clusters that correspond to the perceptual streams. In his 2001 study, Huron identifies six perceptual principles that are related to good voice-leading: toneness, temporal continuity, minimum masking, tonal fusion, pitch proximity, and pitch comodulation [124]. Many methods for voice separation start from a piece for one polyphonic instrument, typically the piano, and apply rules that try to enforce some of these perceptual principles [135, 217, 218, 166]. For example the contig mapping algorithm segregates the score in blocks in which the number of simultaneous notes is constant, named *contigs*, and then links the boundary notes of each of the voices in the contigs using rules to enforce pitch proximity and to avoid voice crossings [38]. Others methods are still rooted in perceptual principles but involve also learning some parameters from a corpus (through genetic algorithms or machine learning) [96, 99, 60], or are based on other ideas such as comparing information values [267]. Other definitions of voices have been also studied, such as the proto-voice analyzed by Finkensiep et al. [81]. A proto-voice is an harmony-related

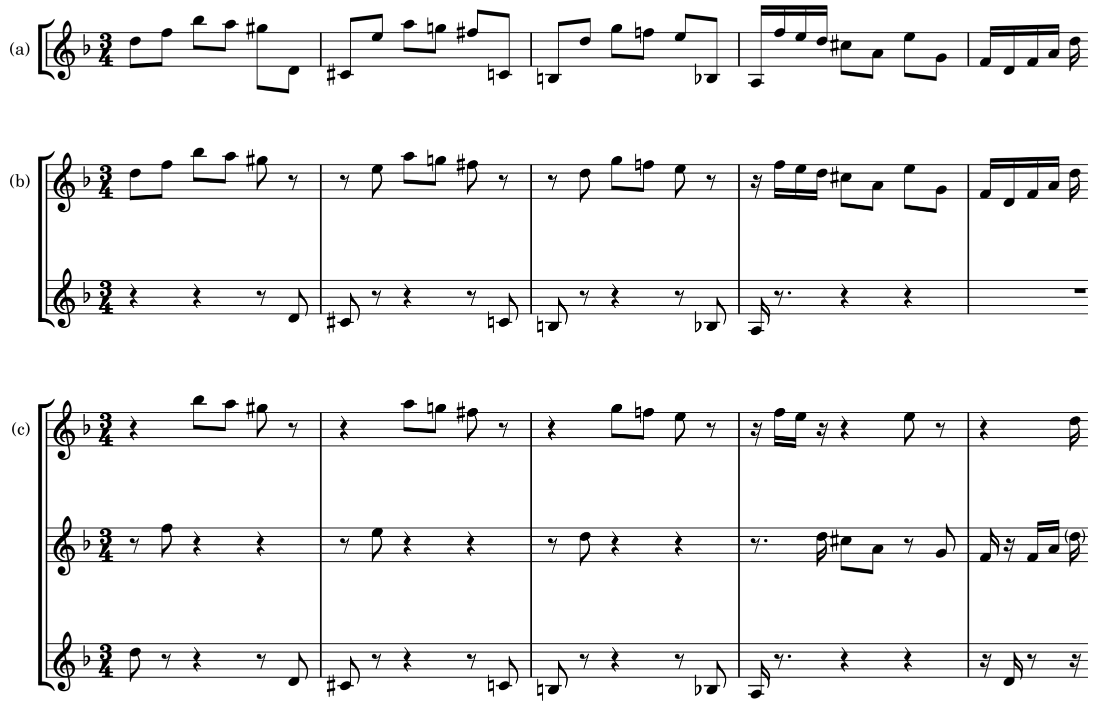


Figure 2.6.: *Chaconne* from the D Minor Partita for violin solo by Bach, BWV 1004, measures 33-36. Cambouropoulos [31] presents three possible analysis of voices: (a) one instrumental voice, (b) two perceptual voices or streams, and (c) three voices constructed following the implied triadic harmonic structure. The figure reproduces the same example from the original article.

'voice' that is perceived in certain pieces for solo monophonic instruments. The definition is related to the second and third categories indicated by Cambouropoulos.
The problem of melody identification is very similar, as it consists in separating notes from a given polyphonic score or MIDI file into two clusters: melody and not melody. In most of the cases the algorithms are looking for a monophonic line with no simultaneous notes. This means for example that they target at identifying only the top note in a melodic stream with octave doubling, but not every study adopted this approach. For example Soum-Fontez et al. [248] introduce texture and divide the notes into melodic and accompaniment layers in a way that allows multiple instruments to belong to the same layer. In general this task is highly unbalanced, since the baseline 'skyline algorithm' achieves already extremely high accuracy results by selecting the note with highest pitch at every time as the melody [262]. The interesting metric is the recall, which indicates the performance in the special cases where the melody is silent or it is not the highest line. Methods have used statistical based classifiers (random forest, support vector machines, bayesian methods) [226, 85, 127] and neural networks [162, 244]. Some studies have approached the two related problems of voice separation and melody identification with one model [117]. The problem of identifying voices, streams, and layers in orchestral music is a bit different. Rather than

having to cluster notes from a single polyphonic instrument into multiple streams, we are interested in grouping together different monophonic parts of a multi-instrumental score. The obtained clusters should also be able to evolve with time.

2.3.2. Modeling and Analyzing Texture

The studies on voice separation and melody identification presented in the previous section can already be considered as examples of computational analysis of music texture. The identification and separation of voices is a first important element of texture analysis, but not all of it. The function that the voices serve in a piece, a description of their internal characteristics, and the relationships between them are part of the constitutive elements of texture. In Huron's texture space model (Figure 2.1) these characteristics contribute to the *density* and *diversity* of layers, giving raise to the four emblematic categories of texture (Homophony, Monophony, Polyphony, and Heterophony) [122]. In multi-instrumental music, timbral aspects also come into play, and have an important role in characterizing orchestral texture.

Computational analysis of music texture is still a niche area of research. The seminal work of Nordgren on textural patterns (see Section 2.1.1 for more details) already identifies certain textural parameters and uses them to compare the orchestration style of romantic composers. These parameters (instruments number, range, register, and spacing, the proportion and register of the gaps, doubling concentrations and the register of concentrations) are quantities that are related to texture, and that can be computed [203]. Far from being a complete formalization of the concept of texture, his work opened the road to further research.
Significant efforts have been made to formalize the theory of texture and to facilitate the application of computational methods, in particular in the Algomus team. Giraud et al. have proposed a formalization of texture layers in string quartets [89]. They describe layer roles as melodies or accompaniments and the relations between the parts forming a layer. Moreover, they release a ground truth annotated corpus and propose a layer detection algorithm. The work of Soum-Fontez et al. is a step forward from classical voice separation and melody identification in the direction of texture analysis, studying the detection of melodies and accompaniments with the help of textural features [248]. Couturier at al. give a systematic description of texture in piano music, considering many textural characteristics such as diversity, density, functions, and internal organization of layers [49]. All those aspects are described with precise fine-grained annotation on a corpus of Mozart piano sonatas [47]. From the same authors is also a study that builds a distance metric to compare textures [48].
The concept of music texture has also been used in other studies in symbolic MIR, which avoid to give a precise characterization or description, but that want to control it in music generation. Texture then becomes a parameter that encodes, for example, "everything that is not harmony" and that can be controlled to generate music [271].

2.3.3. Analyzing Orchestral Texture

To go back to orchestral music, after the work of Nordgren, other researchers have applied computational methods to analyze orchestral texture. Guigue and de Paiva Santana propose a method based on the mathematical theory of integer partitions to analyze orchestral texture and released an implementation of the method for the *OpenMusic*[10] software [97]. Their implementation allows to analyze both the score and a concrete result in the form of a recording. Their model proposes a formal strategy to study the role of orchestration in musical structure and perception and in shaping musical form. They have successfully applied this methods to Webern's *Variations* Op. 30 [207]. Deep Learning techniques have also been used by Chu and Su to classify orchestral texture [42]. Their model was trained on the annotated corpus that we released as part of the research presented in this thesis (more on that will be presented in Chapter 5).

2.3.4. Analyzing Orchestration

Computer scientists have been studying also other aspects of orchestration with computational methods, including orchestral blends. Antoine et al. experimented with regression and classification models to predict timbral and perceptual characteristics of orchestral instrument combinations [12]. This approach aims to estimate instrument timbre fusions directly from abstract information, bypassing the need for extensive acoustic and psychoacoustic analysis. In another work, Antoine et al. proposed a purely score-based system for the identification of orchestral blends [11]. The model uses computational routines to filter blend candidates based on the principle of onset synchrony, harmonicity, and parallelism in pitch and dynamics. Darche analyzed timbral augmentation blends with network models [56].
In audio-based MIR, other works related to orchestral music and orchestration tackled melody extraction from symphonic recordings [23], as well as score-to-audio alignment [189] and instrument source separation [188]. Moreover, arbitrary audio spectra – even actual sounds – may be reconstructed by combining timbres from orchestral instruments [75].

2.4. Music Generative Models

The field of research in Music and AI has achieved important success during the years, thanks also to the increasing computational power of modern hardware architectures [157]. In this section we briefly walk through some historical milestones in computer music generation (Section 2.4.1), we cover the recent advances in the field (Section 2.4.2), we discuss the ideas of machine creativity and collaboration between humans and creative computers (Section 2.4.3), and we conclude by presenting generative models related to orchestration (Section 2.4.4).

[10]`https://openmusic-project.github.io/`

2.4.1. Algorithmic Music Generation from Procedural Methods to the AI Era

The idea of generating music with an automatic procedure dates to way back before computers. An historical example are musical dice games, from the end of the 18th century. The first example of such games is Kirnberger's "Der allezeit fertige Menuetten und Polonaisencomponist" (The Ever-Ready Minuet and Polonaise Composer), published in 1757 [111]. Music dice game started to became popular in the subsequent years. Hedges reports that at least 20 games have been published between 1757 and 1812 [106], among which Mozart's *Musikalisches Würfelspiel*. The idea behind dice games is that a few alternatives are written by the composer for every measure of the piece, and that players choose between them by chance by throwing dices. The possible outcomes are not limitless, but several paths between the composed measures exist and can give raise to very different pieces.
Ada Lovelace, a pioneer of computer programming, is considered the first one to imagine music made by AI in the first half of the 19th century. In a quote from her *Note A*, she imagines music as a possible application of computers besides computation.

> *"Supposing, for instance, that the fundamental relations of pitched sounds in the science of harmony and of musical composition were susceptible of such expression and adaptations, the Engine might compose elaborate and scientific pieces of music of any degree of complexity or extent."*

Her prediction has begun to come true in the 1950s, when computers started being programmed to make music. The "Illiac Suite for String Quartet", is often considered the first piece of music created with an algorithm running on a computer. It was composed by Lejaren Hiller in 1957 in collaboration with mathematician Leonard Isaacson at the University of Illinois. The fourth movement, in particular, was generated with a probabilistic Markov chain [113, 114]. In 1963, it is again Hiller, together with Robert Baker, to develop MUsic Simulator Interpreter for COmpositional Procedures (MUSICOMP). MUSICOMP is described as probably the fist system for computer-aided composition, a program that can be used with any compositional logic supplied by the user [112]. Koenig's Project 1, developed in 1964 to attempt to test the compositional rules of serial music, is an early example of machine-based music generation, contributing to the advancement of algorithmic composition techniques [109].
The idea of aleatory music (also know as chance music), which includes probability and randomness in compositions, has been explored by John Cage, Charles Dodge, Iannis Xenakis, and other avant-garde composers [111]. For example, John Cage's *Atlas Eclipticalis*, was composed by randomly placing translucent paper on a star chart and tracing the stars as notes [213]. Iannis Xenakis composed *Analogique A* and *B*, using Markov models to determine the order of musical sections [278].
In the 1980s algorithmic composition started to get more attention, with models that attempted to compose music in different styles and ways. Experiments include Markov Models, Generative Grammars, Cellular Automata, Genetic Algorithms, Transition Networks, and Chaos Theory [109]. David Cope began *Experiments in Musical Intelligence* (EMI) in 1981 to fight a composer's block. He started by modeling his own style,

and later moved to the styles of other composers. Some of these compositions became fairly successful [44, 45].
The late 1980s also marked the first experiments with Neural Networks and AI in Music [109]. Lewis in 1988 used a training and creation phase with a gradient descent approach [156]. Todd proposed in 1988 a model based on a sequential network using memory with feedback connections [258, 259], and Mozer in 1994 used the CONCERT network by Elman, which can continue a sequence of notes given the probability over the possible note candidates [195].
This brief history covered some of the main milestones in algorithmic composition and does not claim to be complete. For a more exhaustive review of early AI music composition systems we refer the reader to Fernandez and Vico's survey [79]. New AI approaches are considered better for generalization than rule-based approaches and opened the door to modern advances in the field [109]. In the next section we are discussing the latest advancements in music generation with Transformers and other modern deep learning methods.

2.4.2. Machine Learning for Symbolic Music Generation (Transformers and Other Modern Architectures)

Current state of the art models for music generation are based on Machine Learning and Deep Neural Networks. These models have been introduced with success in the field of Music Information Retrieval (MIR) for different tasks related to music analysis and generation, both in the audio and in the symbolic domain. Classical applications of these models are solving the tasks of *classification* and *regression*. An extensive review of their usage in a musical context is given by Liebman and Stone [157].
In this section we focus on recent models that aim at generating music in the symbolic domain. The current trend is to take inspiration from successful Deep Learning models in other domains, such as image and text generation, and to adapt them to music data. This poses a challenge of music representation, as music notes are placed on a pitch and on a time axes that are fundamentally different between each other. The challenge is to represent music as text or as an image in a smart and efficient way for the model architecture (see Section 2.2). These models offer a unique capability that is not available with grammar-based models or rule-based systems: the ability to automatically learn a style from an arbitrary music corpus [25]. However, the use of original compositions by human artists as training data, often without their explicit consent, raises significant ethical and legal concerns, particularly regarding the degree to which these works could be replicated and plagiarized in the outputs generated by such models [194]. Another concern with Deep Neural Networks is the environmental impact of their large energy consumption [222].

A first family of models is based on optimization techniques and constraint programming [255, 110]. These models are in a neighboring category to Machine Learning, but have been included here because of the many compositions that have been created using these kind of systems [9]. Statistical models have also been used, for their ability to

mimic a style by reproducing certain statistical properties of a given corpus [103]. Some researchers have tried to model the multi-level structure of music, combining models hierarchically. For example, Tsushima et al. [261] proposed a music arrangement model that combines generative Markov models and probabilistic grammars with a latent tree structure.
Recurrent Neural Networks (RNN) models, like Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU), have been very popular in music generation. These models treat music like a sequence. A tokenization strategy to represent music is usually required (Section 2.2). We list here a few examples of remarkable models. FolkRNN [252] is a model that allows to generate symbolic music in abc notation in the style of Irish folk songs. Oore et al. [204] proposed a model that can generate performance MIDI with expression, mimicking MIDI recordings of a human performance. DeepBach [101] combines a LSTM with a pseudo-Gibbs sampling method to generate chorales. Jaques et al. [125] proposed to use a LSTM in combination with a reward mechanism to train a generative model in a reinforcement learning fashion.
The adversarial training paradigm has also been employed to generate music. The idea behind Generative Adversarial Networks (GAN) is to pair a generator network with a discriminator network. The two models are trained at the same time: the discriminator is trained to distinguish training data from content generated by the generator, and the generator is trained to maximize the probability that the discriminator will make a mistake [92]. The GAN training protocol has been used for music generation in combination with Convolutional Neural Networks (CNN) [279, 68, 69], with Variational Autoencoders (VAE) [264], and more recently with Transformers [196, 128].
VAEs are another wide category of models that have been employed to generate music. When training VAEs the input is fed to an encoder network that maps it to a lower dimensional latent vector. Noise is then added to it, and the noisy vector is fed as input to a decoder network that maps to the output. The whole network is trained so that the output is the reconstruction of the input from the latent space [140]. With this technique one can affect the output of the generation by modifying the latent vector, allowing for some control to the generation [27, 243, 271, 270, 284]. Recently Wang et al. [273] have proposed a complex VAE model with multiple encoders and decoders that can exploit multi-modal training using both audio and symbolic data to generate symbolic music.
The model that has had the biggest impact in the last years is the Transformer. First introduced by Vaswani et al. [265], the original Transformer is a encoder decoder network based on self-attention. As opposed to previous sequence models like RNNs, the Transformer processes and encodes input data in parallel, rather than sequentially, allowing it to capture long-range dependencies more effectively. Encouraged by the substantial impact of the transformer in Natural Language Processing (NLP), Huang et al. [119] have proposed the music Transformer. The extraordinary results of this model have inspired many researchers in proposing alternative Transformer models for single-track and multi-track music in a wide variety of styles, adopting different tokenization, data representation and embedding strategies. Some of them are bound to generating one or a specific number of tracks because of their training method [65, 73, 116] while some

others adopt strategies to overcome the problem and generate music with any number of tracks [205, 158, 66, 215]. Some models try to incorporate music ideas in the encoding like phrases and themes [200, 280, 242]. And some authors proposed Transformers to solve different tasks than free music generation, like infilling [102, 167], accompaniment generation [220], style-controlled generation [40, 237], or expressive performance generation [257, 153].
The use of pretrained models, like the Bidirectional Encoder Representations from Transformers (BERT) model [63] and the Generative Pretrained Transformer (GPT) [216], has also been experimented. The idea of transfer learning and pretraining is to exploit the knowledge learned from training a machine learning model on one generic task and apply it to a different but related downstream task. For the downstream task, the learning process does not start from scratch, but the model is initialized with pretrained weights that come from the first model. Those already encode features that are hopefully useful for the downstream task as well [208, 274, 256]. This approach is particularly effective when there is limited labeled data since unsupervised pretraining can be used. For example, BERT [63] is pretrained for the tasks of masked language modeling and next sentence prediction. For the first task, a certain proportion of tokens in the sequence is replaced by a `[MASK]` token, and the model is trained to reconstruct the original token. For the second task, the model needs to predict if two spans of text that are presented to it are following each other in the original corpus or not. GPT [216] is instead pretrained autoregressively for next token prediction. The advantages of these architectures and training strategies are that the same pretrained model can be used for autoregressive generation, inpainting [34], and for other analysis tasks like classification [272]. Moreover the use of a pretrained model significantly reduces the amount of task specific labeled data that is required for the downstream task for which the model is finetuned.
Combining language models and music generation models, MuseCoco is a model that is now capable of directly generating music from a text prompt [161].
A recent alternative to Transformers is the Diffusion model. Its adoption is motivated by the success in generating high quality images. It uses an image-like representation for music, and it has been tested for single-track and multitrack generation [190, 186].

Four main challenges of music generation were identified by Briot and Pachet [25]: control, structure, creativity, and interactivity. They are still relevant today, also because some of them are difficult to evaluate and there is no agreement between researchers on how to measure them. For creativity in particular, its definition and evaluation is a complex topic. No statistical metric can be directly used in this regard. Herremans et al. [111] mention that music generation systems should find the right balance between similarity and novelty. In the next section (Section 2.4.3) we expand on this problem by presenting more ideas and results on human creativity, machine creativity, and co-creativity between humans and computers. Narrative and long-term structure are also identified as the most overarching challenges [111].

2.4.3. Computational Creativity and Music Generation

The spread of systems for AI-based music generation studies has contributed to drive a growing interest in machine creativity [185]. Many questions emerge regarding the definition, evaluation, and uses of *creative* machines [132, 131]. Esling and Devis analyze the creativity of AI with the lens of social science studies and highlight the intrinsic limits of AI for self-contained creativity. They suggest a move towards co-creativity in which generative AI algorithms are used as creativity-enhancing tools [77]. This proposal finds its place in Lubart's classifications of modes of human-machine interaction in a co-creative process [163]. Among them, the *computer as a colleague* mode implies a direct involvement of algorithms in the creation of the final output, rather than a mere assisting role *(algorithm as a tool)* for the creator. Similar categories are found in Kantosalo and Jordanous's description of the roles of AI in creative processes. They divide them in the categories of *co-creative colleague* and *creativity support tool* [134].
Some studies focused on human creativity in music [221], and on the way it can be enhanced with Machine Learning (ML) models [173] through modes of interaction between the artist and the model [88, 13]. Difficulties emerge in the evaluation of the creativity of such systems [6]. Louie et al. have also highlighted that a good interface for *steering* AI has an impact on the ability of users to express musical ideas and *"own"* the resulting creation [159]. Co-creative systems in music generation [79, 111, 25, 26, 126] can be divided into two categories: for live performance improvisation [13, 80, 209, 266], and for composition and production [16, 3, 237, 231]. The AI song contest, a competition specifically focused on co-creation in songwriting, inspired different uses of AI and discussions on them [118]. Some members of Algomus have been part of such a team, involving a composer right from the beginning of the system design, resulting in personalized AI models [61].

2.4.4. Generative Tasks Related to Orchestration

After having introduced music generation and co-creativity in general, we now move back to the orchestration problem and present a literature review on generative and co-creative models for orchestration. Crestel identifies four orchestral translation problems (Figure 2.7), two in the symbolic domain (projective orchestration and piano reduction), and two between the symbolic and the audio domain (transcription and orchestral rendering) [50]. In this thesis we expand on the symbolic domain side of this map (see Chapter 4, where we reformulate orchestration tasks in light of our modeling).
Piano reduction is the reverse operation of orchestration, it consists in rearranging an orchestral piece for piano solo. The piano introduces some limitations, due to the fact that it must be played by one musician with only two hands, rather than the many players in an orchestra. The timbre possibilities are also limited to the sounds the piano can make. Some computational studies have tackled this task, with different approaches to identify the essential elements of the music to keep in a piano reduction [120, 254, 199, 115].
Projective orchestration, as named by Crestel [50], consists in the transformation of a piano score into an orchestral score. After defining this task, Crestel proposes Live

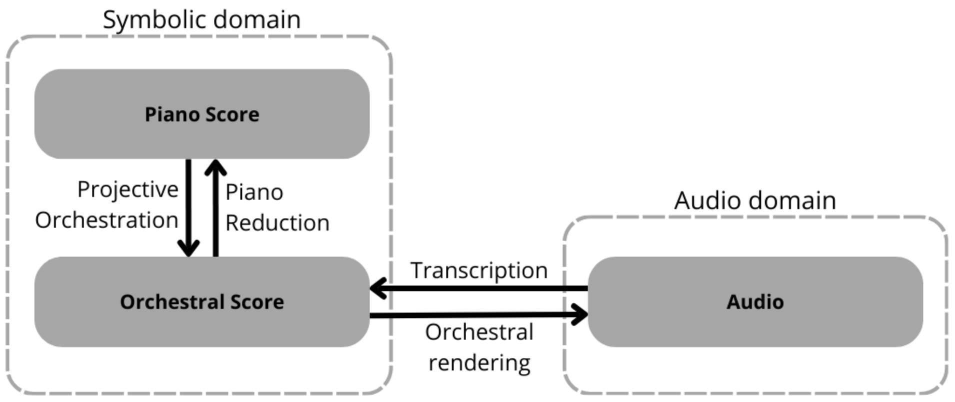


Figure 2.7.: Crestel identifies four orchestral translation problems: projective orchestration, piano reduction, transcription, and orchestral rendering [50].

Orchestral Piano (LOP), the first system to perform projective orchestration. It is a model based on the Restricted Boltzmann Machine (RBM) and it can assign orchestral sounds to notes played in real time on a digital piano, therefore modeling "live" orchestration of piano music [51]. This model however does not focus on the playability of the generated score by orchestral instruments. Before this, we cite the experiment of Handelman et al. in which z-chains are used to encode the information about music, and perform the instrumentation of an existing composition [105]. Others have dealt with instrumentation by learning to recognize the instrument for every note of music in which all the MIDI tracks had been mixed together [67]. Similarly, Kamada et al. have used a pretrained model to classify the instrument of notes for which the instrument information had been masked [133]. Nabeoka et al. proposed a system for wind band orchestration that uses a U-Net encoder-decoder model with a matrix-based representation [198]. The system can effectively generate scores for wind band, starting from a piano version of a piece. Q&A by Zhao et al. [284] is a multi-purpose multitrack model based on VAEs that can perform track separation, rearrangement, and orchestration of pop music.
Transcription here refers to those tasks that try to construct an orchestral score starting from an audio signal. It does not only include those systems that want to do an actual transcription of an orchestra playing, but also systems that aim at supporting composers in recreating existing sounds through blends of orchestral instruments. Several approaches have been used in this context, including spectral analysis, linear algebra methods, and genetic algorithms [214, 33, 229, 32, 75, 169, 2, 30, 29]. The IRCAM institute in Paris has been very active in this field of research, developing a range of "Audio-targeted Orchestration" software that are part of the *Orchid** family. More recently Cella proposed Orchidea [36, 37], a model for *dynamic* audio-targeted orchestration, in which the target sound is described as evolving in time.
The last transformation mentioned by Crestel is orchestral rendering, which is not really related to orchestration. It concerns the process of simulating realistic orchestral

sounds with computers[11].

Some orchestration-related models do not fit in any of the four translation problems, but will find their spot in the classification we give in Chapter 3. Miranda et al. proposed a system that generates voicing and orchestrations of chords trying to match verbal timbre descriptors [187]. Close to orchestration is also the problem of arrangement. Pachet used several algorithms to produce re-arrangements of Beethoven's Ode to Joy in different styles [206]. McCloskey et al. have proposed an algorithm for automatic arrangement [179]. Zhao et al. have proposed AccoMontage-3 [283], a model for lead sheet arrangement. Recently, Le et al. have proposed METEOR, a model for orchestral music rearrangement, or style transfer, with a focus on melody fidelity [150].

Finally some have attempted to directly model orchestral music composition. SymphonyNet is a deep learning model based on the linear Transformer architecture, capable of generating orchestral music either unconstrained (autoregressive), or weakly-constrained by a chord sequence [158]. It uses both structural and note-related tokens, together with a form of Byte Pair Encoding [86, 240]. We will try to extend their tokenization strategy to include texture information in Chapter 8. The Beethoven X experiment is an AI-assisted composition of orchestral music that aimed to be a plausible Beethoven symphony, obtained by the collaboration between the composer Werzowa, musicologists, and computational methods including generative Machine Learning. Several tasks have been modeled in this experiment (with many steps and human co-creation). No code nor data are publicly available, but their co-creative methodology is interesting, divided into *continuation* (expanding melodic lines and themes, from the original melodic material from Beethoven's own sketches), *harmonization* (composition of accompaniments parts for the melodic ideas, including, for example, homophony, counterpoint, and fugue), *transition* (orchestral/polyphonic inpainting, to connect different ideas), and *orchestration* (organizing across the available instruments and instrumental families of the orchestra) [95].

In light of our formalization of orchestration, we will propose a reclassification of both analytical and generative MIR tasks related to orchestration that have been presented in this chapter in Section 4.1.

2.5. Orchestral Music Corpora

Studies have focused on the perception of orchestral music (Section 2.1.2), on the computational analysis of orchestration (Section 2.3.2), and on automated orchestral music generation (Section 2.4.4). Some of these studies have built and released open corpora with orchestral scores and, sometimes, audio recordings to help research on

[11]Several commercial software offer an orchestra simulator, with different levels of realism and control of the results. These include *Sibelius*, *Finale*, *Dorico*, *MuseScore*, *Logic Pro*, *OrchSim*, and many more.

orchestration techniques and perception. These corpora are an invaluable resource to help research in orchestration techniques, automation, and perception.

OrchARD The taxonomy proposed by McAdams et al. describes perceptual phenomena arising from orchestral music, relying on auditory grouping mechanisms linked to orchestration techniques (refer to Section 2.1.2 and to the TOGE article [178]). The Orchestration Analysis & Research Database (OrchARD, as of 2024 not openly released)[12] contains annotations of the phenomena described in the TOGE in certain excerpts of orchestral music. These range from the low-level instrumental blends up to high-level auditory effects such as those of Segmental Grouping. The database contains annotations, scores (mostly in pdf format and some in MusicXML), and recordings of the excerpts. The annotations only target a few instruments at specific parts of the score, when the interesting phenomena occurs, and are not systematic on every measure of the score.

POD and SOD Crestel et al. published the Projective Orchestral Database (POD) gathering aligned MIDI files for both piano and orchestral versions of 196 pieces [52]. The dataset was constructed to research systems for 'projective orchestration', which is the name the authors give to the generation of orchestral scores starting from piano scores, much like how traditional orchestration is taught in conservatory and music school classes. The Symbolic Orchestration Dataset (SOD), by the same authors, extends the POD with a collection of MIDI and MusicXML files of orchestral scores only (without aligned piano versions).

SymphonyNet Dataset Liu et al. [158] proposed a Transformer model to generate multi-track orchestral music (more details in Section 2.4.4 and Chapter 8). Together with their generative system, they release the dataset on which it has been trained, composed of 46,359 polyphonic MIDI files. A portion of them (728 files) consists of western classical symphonic works, while the majority are in other styles, like pop and video game music.

Wagner Ring Dataset The Wagner Ring Dataset (WRD) is part of a long-term interdisciplinary research project conducted by research groups in Erlangen and Saarbrücken, Germany [275]. It is a multi-modal and multi-version annotated dataset built on Wagner's "Ring" cycle, comprising four operas organized into eleven acts and 21939 measures. The WRD provides score representations in various modalities and 16 recorded performances (3 publicly available). The WRD aims to support and advance research in music processing, Music Information Retrieval (MIR), and computational musicology. It has been successfully used to train a cross-version representation learning model for orchestral music (in the audio domain) [144].

[12]`https://orchard.actor-project.org/`, access can be granted by request to interested researchers.

Other sources of symphonic scores: IMSLP, MuseScore, and KernScores There exist many platforms that allow sharing and redistributing digital scores and MIDI files with a public domain license. Among them we mention the International Music Score Library Project (IMSLP)[13]. It mostly contains pdf scans of professional scores that have entered the public domain because of the elapsed time since the death of the author and the publication.
MuseScore[14] is an open source music notation software and a platform to share scores made with the software. An independent community of notation amateur is very active in transcribing and uploading scores from the western classical repertoire, and some academic-coordinated collective efforts have produced very high quality corpora of transcriptions [94].
KernScores[15] is an online library of scores in the **kern file format, native to the Humdrum Toolkit for Music Research [236]. The website also provides automatic translations into several other popular data formats for digital musical scores like MIDI. The collection has been created to assist projects dealing with the computational analysis of musical scores.
The scores in our corpus (see Chapter 5) have been originally collected from MuseScore and KernScores.

[13]`https://imslp.org/`
[14]`https://musescore.com/`
[15]`https://kern.humdrum.org/`

Part I.

Modeling Orchestration

3. Abstract Models of Orchestration

Through history, there is no singular "standard" orchestra as music featuring multiple instruments evolves in accordance with the style and the expectations of each era. Orchestras have evolved during time in the number of instruments involved, their characteristics, and their similarities and heterogeneity in sound and timbral properties. In the baroque era, orchestras were composed of a few strings, optionally woodwinds, and natural brass instruments. In the romantic and post-romantic period, instead, we have very huge orchestras with many instruments and players. Richard Wagner, for example, employs around 90 musicians for his Opera cycle *Der Ring des Nibelungen* (WWV 86). Big orchestras reach their peak with Berlioz, and 20th-century orchestras involve new instruments such as the english horn and various percussions [4]. In the contemporary period, there is a renewed interest for chamber music and composers prefer smaller ensembles in which they can explore instruments' extended techniques and new combinations of their timbres [10].
From this, the necessity to take a snapshot of a specific time, the end of the 18th-century, with a mature "classical orchestra" [249], with great influence on later periods. We focus on a selection of 24 first movements of symphonies in the western "classical style" [230] composed between 1779 and 1824 by the masters of the "first Viennese school" (Haydn, Mozart, and Beethoven). These symphonies usually employ orchestras consisting of strings, woodwinds in pairs, natural horns and trumpets [247], and a pair of timpani. The first movement typically follows the sonata form [39, 108], which holds true across all the 24movements considered, although some of them exhibit unconventional structures[1]. The interplay between primary and secondary theme sections, along with the development, offers composers opportunities to experiment with orchestral textures. Unlike earlier polyphonic music, which predominantly focused on contrapuntal textures, classical symphonies by Haydn and Mozart exhibit a homophonic texture, where melodic lines are supported by harmonies constructed from chords [107, p. 223]. Beethoven expanded the ensembles and pursued greater emotional expression through his compositional techniques, incorporating frequent and broader modulations [107, p. 227].

Orchestral music writing is an art whose "secrets" are traditionally transmitted through examples, practice and mentoring from one master to their students. As seen in Chapters 1 and 2, several books and treaties discuss orchestration techniques, but we are still far from a formalized theory of Orchestration. In this chapter we aim at expanding the understanding of such craft, by proposing three models of orchestration. We introduce

[1]In the Mozart 32nd symphony, a kind of sonata form is split among the first and the third movement [39, p. 93-94].

a taxonomy that uses both a perceptive and a music theoretical approach to describe how orchestral layers with different roles create orchestral texture in classical and early-romantic symphonies (Section 3.1). We then introduce the layer score (Section 3.2), an object with which we model an abstract composition with no instrumentation details. Finally, we present the orchestration plan (Section 3.3), through which all layers in a composition can be mapped to an instrumentation. The first object has been introduced in our DLfM publication [149], and the last two objects in our EvoMUSART publication [165], but here we present a more formal and unified model of these objects and their constituent parts.

3.1. A Taxonomy to Describe Orchestral Layering in Classical and Early-Romantic Symphonies

In this section we focus on describing the annotation syntax and process. We start by describing how auditory grouping principles contribute to the formation of orchestral *layers* (Section 3.1.1), and we formalize these ideas with mathematical language (Section 3.1.2). We then describe which *roles* layers can take in the texture, and the *relations* between instruments inside layers and between layers (Section 3.1.3). After, we present a few more mathematical definitions of properties of orchestral layers, and an attempt to define the orchestral texture of a measure (Section 3.1.4). We conclude the section by discussing similarities and differences between this model and the concept of orchestral blend from the TOGE [178] (Section 3.1.5).

3.1.1. Grouping Instruments into Layers

Western music frequently employs a *homophonic* texture, where the musical material can be divided into melody and accompaniment [143]. Figure 3.1 presents an excerpt from Symphony No. 41 by Mozart, where it can be said that the violins are playing a melody while the other instruments provide the accompaniment. However, it is reductive to talk about orchestral music only in terms of melody and accompaniment, as those may appear in various complex forms. Melodies and accompaniments can have different textures thanks to the involvement of a large group of instruments, and several passages might not even present a clearly "melodic" line. With our taxonomy we try to be more precise in describing also those cases. Orchestral texture can then be considered to be the assembly and organization of instrumental sounds into more intricate structures.

Based on studies in music perception, such as those presented in Section 2.1.2, we decide to divide the orchestra into *layers*, and we do so systematically, at every measure of the composition. In the Mozart example reported in Figure 3.1, the listener may perceive four layers following concurrent and sequential grouping [178]: the two violin sections (blended), the violas and contrabasses (also blended), the cellos, then the oboe. A layer might be formed by one, two, or more instruments whose sounds are blending together, and that are playing "the same part" from the music theoretical point of

	year	symphony (first movement)	key	bars	layers	av. len.	av. parts	
	1779	Symph. 32, K. 318	G Major	274	218	3.53	2.73	
	1779	Symph. 33, K. 319	B♭ Major	370	248	3.86	2.17	
	1780	Symph. 34, K. 338	C Major	264	269	3.11	2.47	
Mozart	1782	Symph. 35, K. 385, Haffner	D Major	204	170	3.95	2.56	
1756 – 1791	1783	Symph. 36, K. 425, Linz	C Major	287	280	2.70	2.76	
	1786	Symph. 38, K. 504, Prague	D Major	302	402	2.42	2.35	
	1788	Symph. 39, K. 543	E♭ Major	309	431	2.06	2.73	
	1788	Symph. 40, K. 550, Great G minor	G minor	299	278	3.33	2.50	
	1788	Symph. 41, K. 551, Jupiter	C Major	313	360	2.71	2.60	(Figure 3.1)
	1793	Symph. 99	E♭ Major	202	266	2.73	2.37	
	1793-94	Symph. 100, Military	G Major	289	250	3.16	2.66	
Haydn	1793-94	Symph. 101, The Clock	D Major	351	302	3.14	2.62	
1732 – 1809	1794	Symph. 102	B♭ Major	311	262	3.18	2.76	(Figure 3.9)
	1795	Symph. 103, Drumroll	E♭ Major	229	191	3.09	2.82	(Figure 3.11)
	1795	Symph. 104, London	D Major	294	250	3.20	2.69	
	1800	Symph. 1, op. 21	C Major	297	344	2.19	3.01	
	1803	Symph. 2, op. 36	D Major	360	411	2.59	2.71	
	1805	Symph. 3, op. 55, Eroica	E♭ Major	691	610	3.44	2.41	
Beethoven	1807	Symph. 4, op. 60	B♭ Major	498	353	3.46	2.85	(Figure 3.8)
1770 – 1829	1808	Symph. 5, op. 67	C minor	502	368	3.22	2.89	(Figure 3.10)
	1808	Symph. 6, op. 68, Pastoral	F Major	512	284	4.99	2.33	
	1813	Symph. 7, op. 92	A Major	450	434	3.05	2.72	
	1814	Symph. 8, op. 93	F Major	373	357	3.00	2.83	
	1824	Symph. 9, op. 125	D minor	547	603	3.29	2.54	(Figure 3.16)
				8528	7941	3.14	2.63	

Table 3.1.: The corpus of annotations contains 24 first movements of Haydn, Mozart, and Beethoven symphonies [149]. The last three columns give the number of annotated layers, their average length (in bars), and the average number of instrumental parts per layer.

view. In general, more layers can appear simultaneously in the same measure, but one instrument[2] can appear in at most one layer at a time[3]. Dividing the orchestra into layers can then be taught as an equivalent operation to the partition of a set. In the example of Figure 3.1 the ensemble

$$\{\mathsf{Ob}, \mathsf{Vln1}, \mathsf{Vln2}, \mathsf{Vla}, \mathsf{Vc}, \mathsf{Cb}\}\,,$$

composed of oboes, violin 1, violin 2, viola, cello and contrabass, is a set of instruments, that is partitioned as

$$\{\{\mathsf{Ob}\}\,, \{\mathsf{Vln1}, \mathsf{Vln2}\}\,, \{\mathsf{Vla}, \mathsf{Cb}\}\,, \{\mathsf{Vc}\}\}\,.$$

In measures 101 and 102, the oboe is silent, and it starts to play at measure 103. By simply partitioning the ensemble, we cannot distinguish between silent and playing layers. Clearly, partitioning the ensemble is not enough to describe an orchestral texture. We propose a less naive way to define layers, that also allows to describe several of their attributes, like the *role* among the simultaneous layers in the whole orchestra (Section 3.1.3.1) and the *relation* of the instruments inside a layer and possibly between layers (3.1.3.2).

[2]When we mention one instrument, we are actually referring to one instrumental section. Violin 1 is "one instrument", even though several musicians are playing that part.

[3]The only exception to this rule is the *divisi* indication, which splits one instrument of the orchestra in two. In Section 3.1.2 we also discuss this particular case.

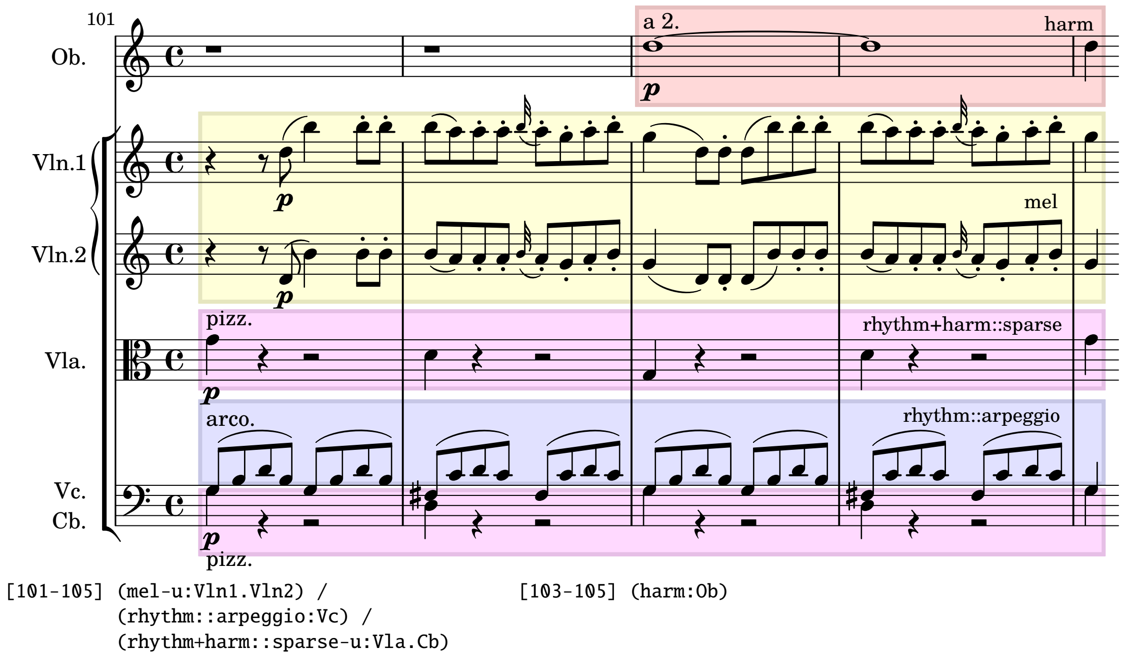


Figure 3.1.: *Allegro Vivace* in Symphony No. 41 *Jupiter* by Mozart, measures 101-105, from edition [100]. Four layers can be heard: one melodic (violins – Vln1 / Vln2), one bringing some rhythm (cellos – Vc), another one with sparse elements (violas and contrabasses – Vla / Cb), and, two measures later, an harmonic layer (oboe – Ob). In the layers containing several instruments, they are here in unison or octave doubling (-u).
http://algomus.fr/fm/mozart41-101.mp3

3.1.2. Formal Formulation of the Layers' Taxonomy

In this section we try to give formal definitions of the concepts of orchestral layers and annotations[4]. Music is made of notes, and notes can be described by their temporal position, their pitch, and their duration.

Definition 3.1 (Music Note)**.** Let $\mathcal{T} = [0, t_{end}] \subset \mathbb{R}$ be the time, $\mathcal{P} = \{\text{C-1}, \text{C}\sharp\text{-1}, \text{D-1}, \dots\}$ be the pitches, and $\mathcal{D} = \mathbb{Q}$ be fractions representing the durations of notes. Then a note can be described as a point

$$(t, p, d) \in \mathcal{T} \times \mathcal{P} \times \mathcal{D}.$$

When we place notes in an orchestral score, we have another dimension, the instrument. We introduce then the following.

Definition 3.2 (Ensemble)**.** An orchestral ensemble is a set $\mathcal{E}$ of distinct instrumental parts.

[4]The actual syntax used for the annotations, is detailed in Appendix A, and allows to describe at a whole the orchestral texture, including layers identifiers and concise expressions describing instruments joining a layer at a later measure, or layers continuing previous layers.

$$\begin{aligned}\mathcal{E} = \{&\mathsf{Fl1}, \mathsf{Fl2}, \mathsf{Ob1}, \mathsf{Ob2}, \mathsf{Cl1}, \mathsf{Cl2}, \mathsf{Fg1}, \mathsf{Fg2},\\ &\mathsf{Hrn1}, \mathsf{Hrn2}, \mathsf{Trp1}, \mathsf{Trp2},\\ &\mathsf{Timp},\\ &\mathsf{Vln1}, \mathsf{Vln2}, \mathsf{Vla}, \mathsf{Vc}, \mathsf{Cb}\}\end{aligned}$$

is a typical ensemble for a classical symphony, with two distinct parts for woodwinds (flutes, oboes, clarinets, and bassoons) and brass (natural horns and trumpets), one timpani part, and string parts (violins 1 and 2, violas, cellos and contrabasses).
With distinct instrument parts, we mean parts that are autonomous, even if the instrument is the same. For example, violin 1 and violin 2 both consist of violin players, but are two distinct parts. Distinct instrument parts roughly coincide with the staves of the conductor score, with some exceptions. Indeed, some instruments, like the piano and the harp, have their part written on multiple staves. Moreover, many scores display distinct parts written on the same staff with *divisi* indication. For example, it is typical to have two distinct flute parts in classical symphonies, despite the fact that they are normally written on the same staff and that they are occasionally playing in unison. It is also typical to divide one string section in two in some punctual moments of the score (for example the first violin or the cello parts can be divided)[5]. In that case one musician per desk would read the top line and one the bottom line. At times, the two parts might belong to different orchestral layers (for example we could find the viola divided in two parts of which the highest one is doubling the violins and the lowest one the basses). In more modern pieces, this division could go even further, splitting an instrument section into 3, 4, or more subsections. In any case we can consider these subsections to be the "individual instruments", and the collective name of the instrument to be a virtual alias to indicate the set that collects them all. For example, the viola (Vla) could be divided into two parts in certain measures of a symphony. Then Vla1 and Vla2 would be the individual instruments, which can appear only in a layer at a time, and $\mathsf{Vla} = \{\mathsf{Vla1}, \mathsf{Vla2}\}$ would be a collective name that is used when Vla1 and Vla2 are part of the same layer. All the reasoning that follows can be easily extended to this case using this arrangement.

Now we have all the objects we need to define an orchestral score, as a set of orchestral notes.

Definition 3.3 (Orchestral Note and Orchestral Score)**.** Let $\mathcal{T} = [0, t_{end}] \subset \mathbb{R}$ be the time, $\mathcal{P} = \{\text{C-1}, \text{C}\sharp\text{-1}, \text{D-1}, \dots\}$ be the pitches, $\mathcal{D} = \mathbb{Q}$ be fractions representing the durations of notes, and let $\mathcal{E}$ be an ensemble of instruments. An orchestral note is defined by $(t, p, d, ins) \in \mathcal{T} \times \mathcal{P} \times \mathcal{D} \times \mathcal{E}$. Then an orchestral score O is a set of orchestral notes, that is a subset $O \subset \mathcal{T} \times \mathcal{P} \times \mathcal{D} \times \mathcal{E}$.

There are other objects and other properties of notes that can be encoded in a score, such as dynamics (***f***, ***mp***, ***ppp***, ...) and articulations (*pizz.*, *staccato*, ...). We are ignoring those here, but they could be included. Basically here we are projecting the notes into

[5]Each annotation is bond to a score, so the annotation syntax (Appendix A) adopts a specific strategy for *divisi* and distinct parts that are on the same staff.

the 4-dimensional subspace $\mathcal{T} \times \mathcal{P} \times \mathcal{D} \times \mathcal{E}$.
According to a time signature ($\frac{4}{4}$, C , $\frac{6}{8}$,...), that might even change during a piece, time is organized into measures (Figure 3.2). Gotham et al. [93] discuss different music numbering conventions and propose a strategy to overcome their discrepancies. We use measures a lot in the thesis, but we will not go into the details of how to divide time into measures. We will assume that a partition of time into measures exists, and that it is correct.

Definition 3.4 (Measures)**.** Given the time $\mathcal{T} = [0, t_{end}]$, a partition of $\mathcal{T}$ into intervals that respects certain properties, is a set of measures $\mathcal{M} = \{m_1, m_2, \dots, m_N\}$, $N \in \mathbb{N}$.

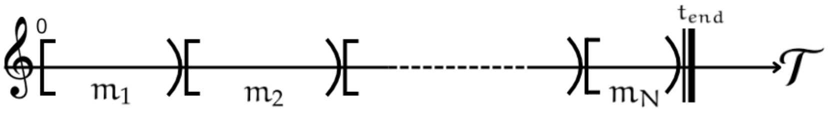
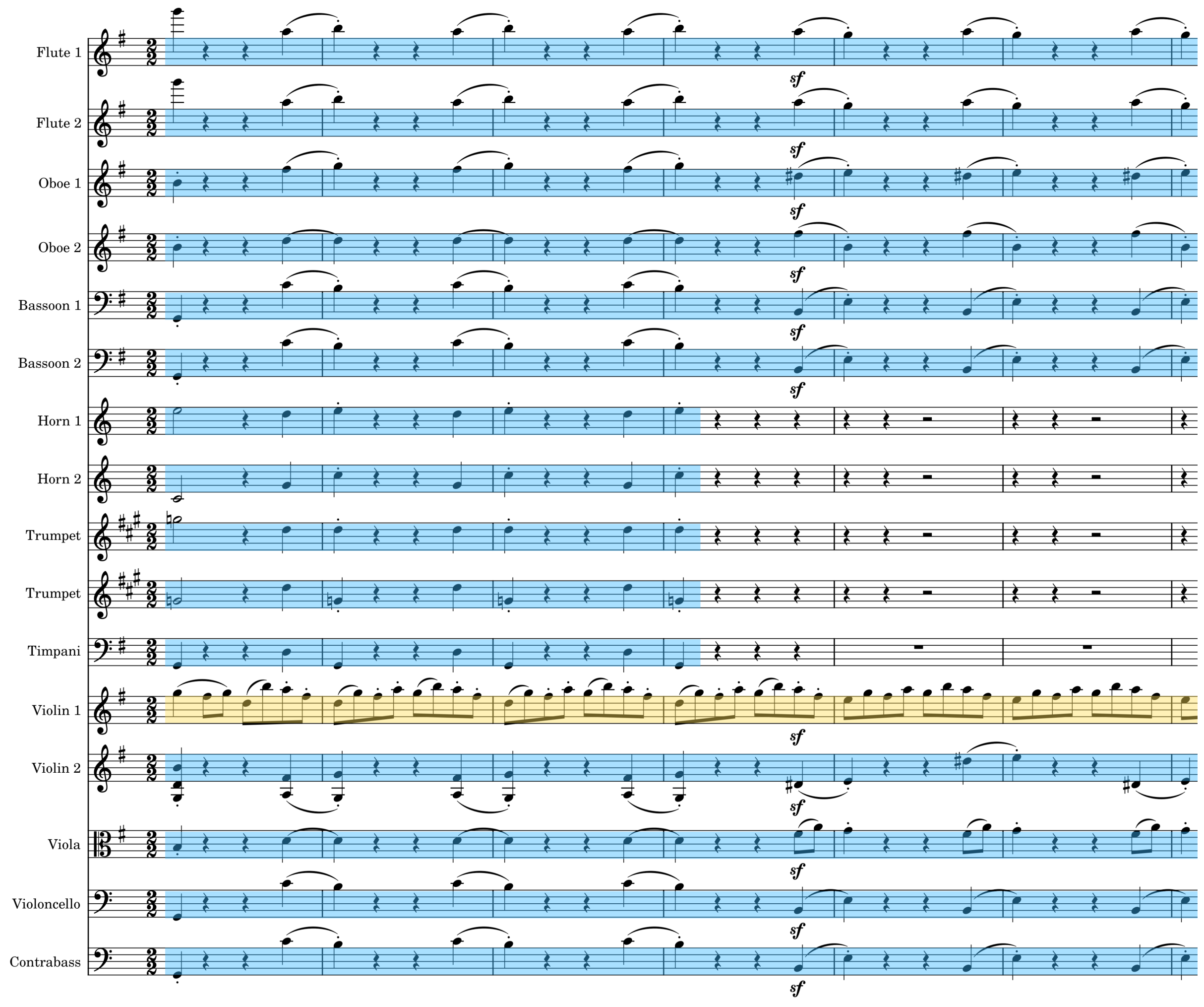


Figure 3.2.: Measures $\mathcal{M} = \{m_1, m_2, \dots, m_N\}$ are intervals that partition the time $\mathcal{T}$.

The choice of using measures as the unit of time for texture annotations is not casual. Orchestral texture is a high-level musical feature that cannot be understood note-by-note. A measure of music is an appropriate time window that is usually wide enough to analyze texture but not excessively wide, which would result in many texture changes inside the same window (as it would be for a formal section). Still the choice remains an approximation; there are many instances (for example in the presence of an anacrusis) in which a texture change does not coincide with the boundary of a measure. However, the problem of precisely identifying texture boundaries is complicated to define and formalize, since different layers beginnings might be shifted in time. Analyzing texture at measure level offers a better overview of the texture evolution in a piece. For every measure we decide to annotate the "prevalent" texture, *i.e.* the one that explains the largest part of the measure. This solution can be a limitation and it can sometimes prevent a precise description of the phenomena we want to study, but in most of the cases it proved to be a good approximation to describe texture.
We would be tempted to define layers as a partition of the ensemble at every measure, but we have seen at the end of the previous section that this would raise some problems with silent instruments. Ideally, instruments should not belong to any layer in the measures in which they are silent. In the example presented in Figure 3.3 the sparse layer spans several measures and its composition in terms of instrumental parts changes. At the beginning it is played by all instrumental parts except for the violin 1, and after four measures brass instruments and timpani stop playing. Then, rather then a subset of the ensemble $\mathcal{E}$, an orchestral layer is better defined as a subset of the cartesian product between measures and ensemble $\mathcal{M} \times \mathcal{E}$.

Definition 3.5 (Orchestral Layer)**.** Let $\mathcal{E}$ be an ensemble, and $\mathcal{M} = \{m_1, m_2, \dots, m_N\}$, where $N \in \mathbb{N}$, be a set of measures. Then, an orchestral layer ℓ is

$$\ell \subset \mathcal{M} \times \mathcal{E},$$



```
[49-54] (rhythm-u:Vln1)
[49-52] (rhythm+harm::sparse-h:Fl1.Fl2.Ob1.Ob2.Fg1.Fg2.Hrn1.Hrn2.Trp1.Trp2.Timp.Vln2.Vla.Vc.Cb)
[53-54] (rhythm+harm::sparse-h:Fl1.Fl2.Ob1.Ob2.Fg1.Fg2.Vln2.Vla.Vc.Cb)
```

Figure 3.3.: *Adagio - Allegro* in Symphony No. 100 *Military* by Haydn, measures 49-54. Two layers can be heard: a rhythmic layer (yellow, light) played by violin 1 (Vln1), and a sparse layer (blue, dark) played by all other instruments until measure 52, and by woodwinds and strings from measure 53. It is an example of a layer with changing instrumentation, since brass instruments and timpani stop playing after four measures.
`http://algomus.fr/fm/haydn100-49.mp3`

whose elements are couples of measures and instrumental parts. A layer can be identified with a layer identifier $\mathrm{id} \in \mathrm{ID}$.

An example of layer is $\ell_{\texttt{rhythm+harm::sparse-h}}$ in Figure 3.3[6], where horns (Hrn1, Hrn2) and trumpets (Trp1, Trp2) take part only in measures 49 to 52, and other instruments continue up to measure 54. Notice that both $\mathcal{M}$ and $\mathcal{E}$ are discrete sets, but $\mathcal{M}$ is ordered and $\mathcal{E}$ is not. The identifier can in principle be anything, for example a name:

$$\mathrm{id} \in \{\texttt{mel1}, \texttt{rtm1}, \texttt{mel2}, \texttt{harm}, \ldots\}.$$

However, for convenience, we will sometimes assume that $\mathrm{ID} = \mathbb{N}$ and that we have $n \in \mathbb{N}$ layers $\ell_1, ..., \ell_n$.
If we want to know the instrumentation of a layer $\ell \in \mathcal{M} \times \mathcal{E}$ at a certain measure m, we fix a certain measure $m \in \mathcal{M}$ and take all the instruments $i \in \mathcal{E}$ such that $(m, i) \in \ell$. This set is called an m-section of the layer ℓ.

Definition 3.6 (Layer Instrumentation at Measure m)**.** The layer instrumentation for any given layer ℓ at measure m, is the set of all the instruments $i \in \mathcal{E}$ such that $(m, i) \in \ell$, and it is called the m-section of the set ℓ.

$$\ell i^m \subset \mathcal{E}$$

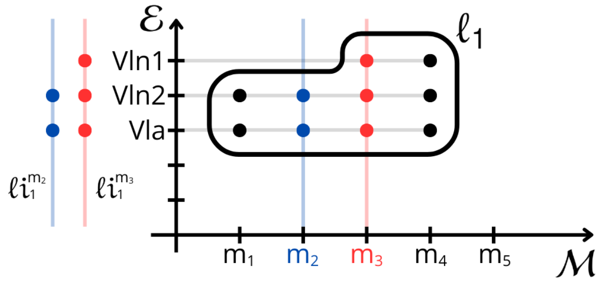


Figure 3.4.: Layer Instrumentation at measure m_2 and m_3 of layer ℓ_1. Layer ℓ_1 (points surrounded by the curved bold line) has, at measure m_2, an m_2-section including violin 2 and viola. At measure m_3 the m_3-section of ℓ_1 includes violin 1 and violin 2 and viola.

In Figure 3.4 we see a schematized representation of a layer. At measure 2 (blue) the instrumentation is Vln2 and Vla, at measure 3 (red) it is Vln1, Vln2 and Vla.
No instrument can belong to more than one layer simultaneously. We impose this with the following validity condition.

[6]The string `rhythm+harm::sparse-h` used to identify this layer has a meaning: it is a layer with a mixed rhythmic (`rhythm`) and harmonic (`harm`) role, that we call `sparse`. Moreover the parts composing it are in homorythmic relation (`h`). Layer roles and relations will be explained in detail in Section 3.1.3.

Definition 3.7 (Valid Collection of Orchestral Layers)**.** A collection of orchestral layers $\mathcal{L} = \{\ell_1, \dots, \ell_n\}$, where $n \in \mathbb{N}$, is *valid* if and only if all layers are pairwise disjoint, *i.e.*

$$\ell_i \cap \ell_j = \emptyset, \quad \forall i, j \in 1, \dots, n, \quad i \neq j.$$

When performing annotations, we must group instruments and output a valid collection of layers (Figure 3.5). Notice that the concept of validity does not directly rely on simultaneity, but rather on the fact that the sets in $\mathcal{M} \times \mathcal{E}$ do not intersect. The

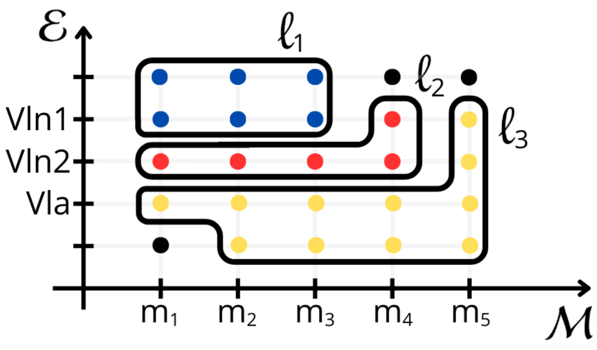


Figure 3.5.: A collection of layers is valid when they do not have any points in common.

annotation syntax (Appendix A) fully describes layers, even when the instrumentation is evolving, or when some instruments are divided (*divisi*). It describes the composition of layers through the instrumentation at measure m from Definition 3.6. In the scores shown in this chapter we report the composition of the layers with a list of instruments[7].

3.1.3. Layer Descriptions: Roles and Relations

Layers can be described through a function that assigns a description to every layer.

Definition 3.8 (Layer Description Function)**.** Let $\mathcal{L} = \{\ell_i\}_{i \in ID}$ be a valid collection of layers. Given a set of layer descriptions D, a layer description function is a function

$$\delta : \mathcal{L} \to D,$$
$$\ell_i \mapsto d,$$

that assigns to every layer ℓ_i, where $i \in ID$, a description $d \in D$.

In the definition above, we intentionally remained abstract, to allow the choice of different sets of descriptions D (Figure 3.6). In the following we specify the description of layer roles, relations inside layers, and relations between layers.

[7]For example, in Figure 3.3 the sparse layer `rhythm+harm::sparse-h` at measure 54 is composed of flutes 1 and 2, oboes 1 and 2, bassoons 1 and 2, violin 2, viola, cello, and contrabass, as indicated by `Fl1.Fl2.Ob1.Ob2.Fg1.Fg2.Vln2.Vla.Vc.Cb`

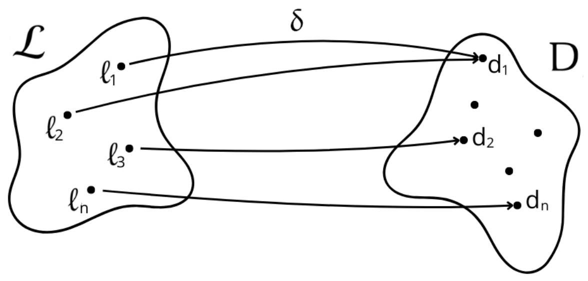


Figure 3.6.: Layer description: all layers $\ell_i \in \mathcal{L}$ can be mapped to a description $d_i \in D$.

3.1.3.1. Layer Roles

Benward and Saker describe music texture by establishing different types of textural elements. These are *melodies* (primary, secondary, or in parallel motion) and *supports* (static, harmonic / rhythmic or both) [18]. Grounded in this classification, and on other concepts commonly discussed in orchestration literature, we propose to assign zero, one, or several *roles* to each identified layer. The basic roles are Melody, Rhythmic Accompaniment, and Sustained Harmony, and can be further specialized into sub-roles that give a qualitative description of their content.
Formally, given a set of roles $\mathcal{R}$, such as $\mathcal{R} = \{\mathsf{mel}, \mathsf{rhythm}, \mathsf{harm}\}$, we can chose its power set $2^{\mathcal{R}}$ as the set of descriptions. Then, the description function

$$\begin{aligned}\delta_{roles} : &\mathcal{L} \rightarrow 2^{\mathcal{R}},\\ &\ell_i \mapsto R,\end{aligned}$$

assigns to every layer ℓ_i, where $i \in ID$, a set of layer roles $R \subseteq \mathcal{R}$. With this description, a layer can have zero, one, or multiple roles, allowing mixed roles (Figure 3.7).

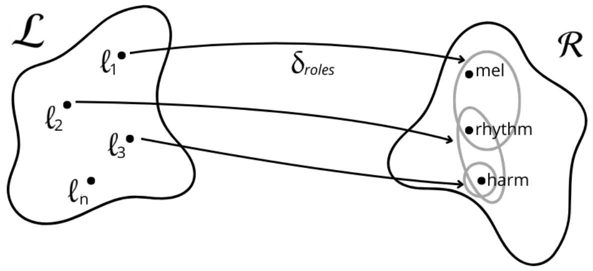
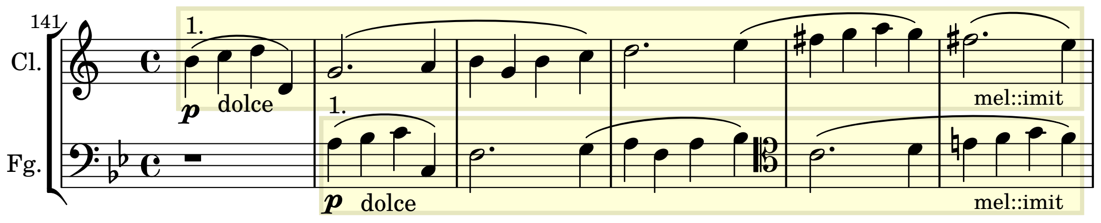


Figure 3.7.: Layer description: roles. All layers can be mapped to a set of layer roles $R \subseteq \mathcal{R} = \{\mathsf{mel}, \mathsf{rhythm}, \mathsf{harm}\}$. In the example $\delta_{roles}(\ell_1) = \{\mathsf{mel}, \mathsf{rhythm}\}$, $\delta_{roles}(\ell_2) = \{\mathsf{rhythm}, \mathsf{harm}\}$, and $\delta_{roles}(\ell_3) = \{\mathsf{harm}\}$.

In the following, we present each role and sub-role.

Melody (mel). A melody may be defined as a *thematic* music material perceived as the *foreground*, in opposition to the background accompaniment [253]. In most cases, but



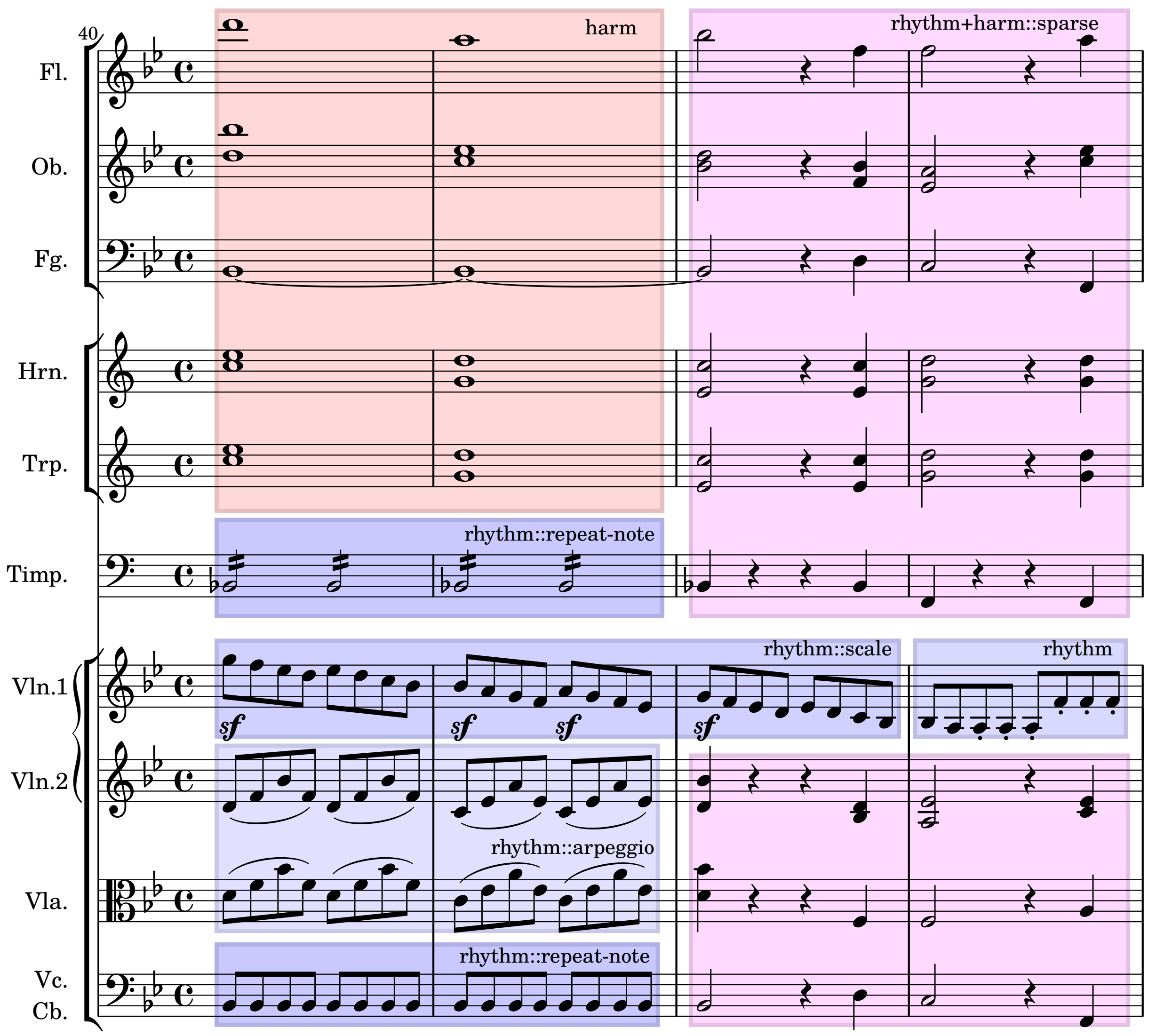
[141-148] (mel::imitation:Cl1)   [142-148] (mel::imitation:Fg1)

Figure 3.8.: *Allegro Vivace* in the first movement of Symphony No. 4 by Beethoven, measures 141-146, from edition [263]. The first clarinet (Cl1) and the first bassoon (Fg1) play a melody in canon, a type of imitation (mel::imitation).
`http://algomus.fr/fm/beethoven4-141.mp3`

not all, the melody is played by instruments of high register [4]. Melody identification in symbolic scores is an active field of research [244, 248].
One might expect that there is only one melodic layer, (as it is the case for Figure 3.1, where the violins are playing the melody), accompanied by an homorhythmic multiple-part harmony resulting in a chordal texture [182], but this is not usually the case in most symphonic excerpts. A polyphonic texture is also possible, by employing a writing style with counterpoints techniques (a specific sub-role mel::imitation[8] can be introduced to describe this, Figure 3.8). Sections in which "no single part has any melodic meaning" [223, p. 99] also exists. Those are sections that develop a predominantly harmonic and/or rhythmic discourse. We see an example of this in Figure 3.9, where the first violins were not annotated as melody on measures 40-42. This is subject to interpretation (subjectivity in the annotation process is discussed in Section 5.1.2.2). The instrument has been interpreted as fulfilling a mainly "rhythmic" role. We annotate melody only in case of clearly identifiable themes, leaving the possibility of having *no melody* in some segments. On the contrary, the annotation guidelines specifies that when there is a short rhythmic fragment *inside* a melody, that may last up to two measures, it is still annotated as melody, in order to favor longer layers.

Rhythmic Accompaniment (rhythm). The references describe a lively orchestral accompaniment as a *fluttering accompaniment* [4, p. 632], that adds *"movement"* [142, III, p. 53] and a *pulsating* [202] effect. Through the employment of certain techniques, it can create *dynamism* [211, p. 363], or *agitation* [19, p. 12], by adding instruments and/or playing "more notes" with the existing instruments. Benward and Saker [18], categorize these elements as "rhythmic support", and we identify such layers as fulfilling the role of "rhythmic accompaniment". The description can be further refined through the sub-roles described below. In the following, a *significant part* of the measure means "at least half of the measure", that is at least two beats in 4/4 or 3/4 and one beat in 2/4 or 3/8.

[8]mel::imitation stands for the imitation sub-role of the melody (mel) main role.



```
[40-42] (rhythm::scale:Vln1)              [42-43] (rhythm+harm::sparse-h:TUTTI)
[40-41] (rhythm::repeat-note:Timp) /      [43-43] (rhythm:Vln1)
        (rhythm::arpeggio-u:Vln2.Vla) /
        (rhythm::repeat-note-u:Vc.Cb)
```

Figure 3.9.: *Allegro Vivace* in the first movement of Symphony No. 102 by Haydn, measures 40-43, from edition [212]. In measures 40-41, four layers are heard: one sustained harmony (woodwinds and brass), one with scales (first violins, see text for discussion), one with arpeggios (second violins and viola), and one with repeated notes (cellos, contrabasses). From measures 42, the orchestra, in (almost) tutti, plays sparse chords. The annotated layers are rounded to measures.
`http://algomus.fr/fm/haydn102-40.mp3`

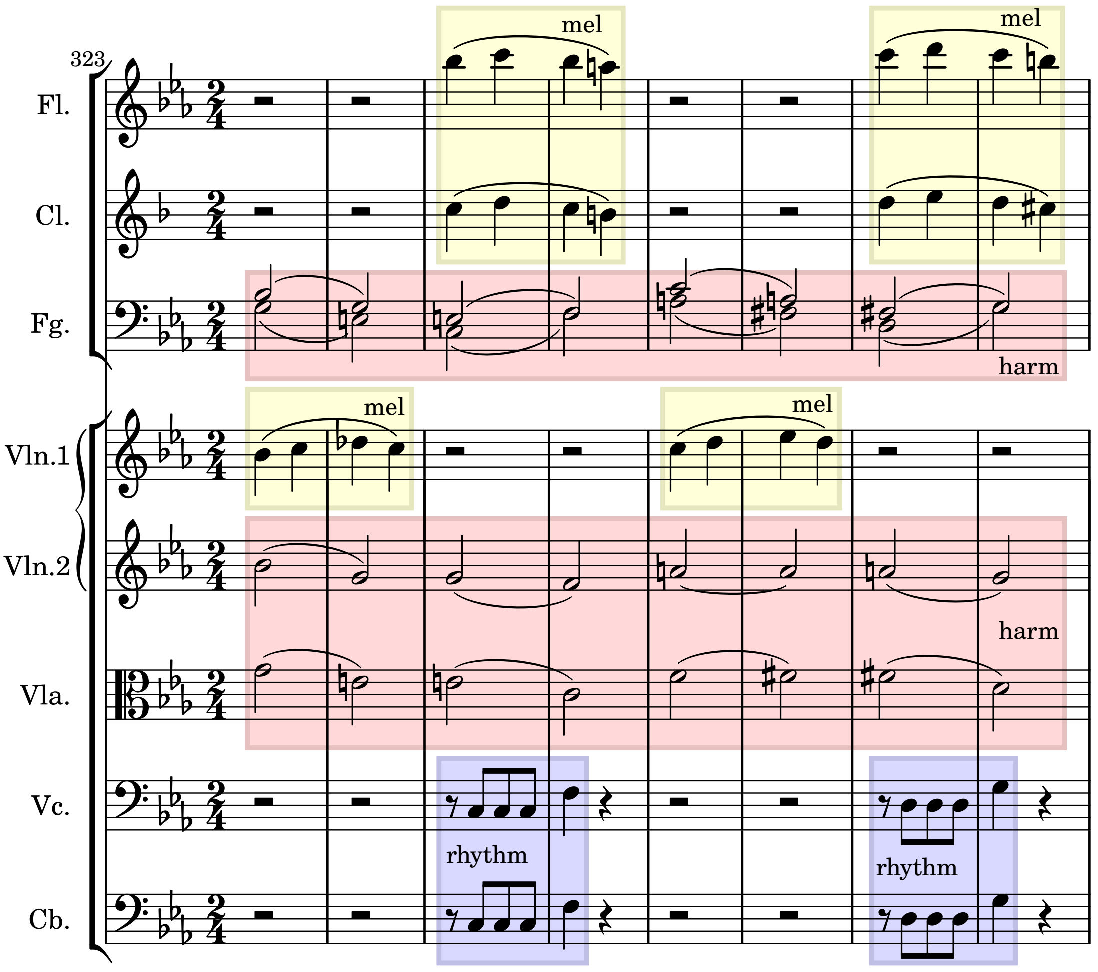
- rhythm::repeat-note. Repetition of the same note (as cellos and contrabasses in Figure 3.9) or tremolo. A significant part of the measure is composed of the same note in quarter notes or less.
- rhythm::oscillation. Oscillations around a chord note, with neighbor tones (*ondulation* [4, p. 632]) or between two chord notes (*"batterie"* [142, p. 59], *animated harmony* [211, p. 372]). A significant part of the measure is composed of at least two oscillations between the same notes in quarter notes or less, with the same rhythm.
- rhythm::arpeggio. A pattern that includes chord notes (as the cellos in Figure 3.1). A significant part of the measure is composed of notes from one chord, with possibly one non-chord tone.
- rhythm::scale. Diatonic or chromatic scales (as the first violins in Figure 3.9). A significant part of the measure is composed of sequences of more than four successive notes – ascending or descending.

The *repetition* of the pattern (same note or an oscillation/arpeggio/scale) underlines the rhythmic effect of those layers.
These conventions can be relaxed in certain cases to obtain more coherent annotations on the long scale (more about the annotation process is presented in Section 5.1.2.1). For example, we decided to annotate the famous pattern in the Beethoven 5th symphony (Figure 3.10) only as rhythm instead of rhythm::repeat-note, to differentiate it from other rhythm::repeat-note in the same movement.

Sustained Harmony (harm). Parts that are playing long sustained notes mostly in whole/half notes, occasionally quarter notes to form a chord. They are called "sustained tones" by Piston [211, p. 371] or "harmonic accompaniment" by Rimsky-Korsakov [223, p. 37]. In the examples already discussed we can find this realized through a single pedal pitch supporting the underlying harmony (as the dominant D pedal on the oboe in Figure 3.1), or by a multiple-part homorythmic harmony of several pitches (bassoons, second violins, and violas in Figure 3.10).

Mixed Roles. Melody, Harmony, and Rhythm are not independent. Most of the times, they are intertwined: melodic and rhythmic parts contribute to the harmony. Indeed, when one performs harmonic analysis, one should consider all the parts of a composition. Moreover, harmonic layers may also present some melodic or rhythmic features. We acknowledge that ambiguous cases exists, where it is difficult to characterize a layer as either melodic, rhythmic, or harmonic, and we introduce the possibility to annotate multiple roles for one layer. Two particular mixed roles emerge to be very important:

- rhythm+harm::sparse. Isolated chords (often homorhythmic) called "*secco*-like chords" by Adler [4, p. 590] (as the tutti in Figure 3.9), or single notes (often on low register instruments, such as in Figure 3.1), that may emphasize either strong beats

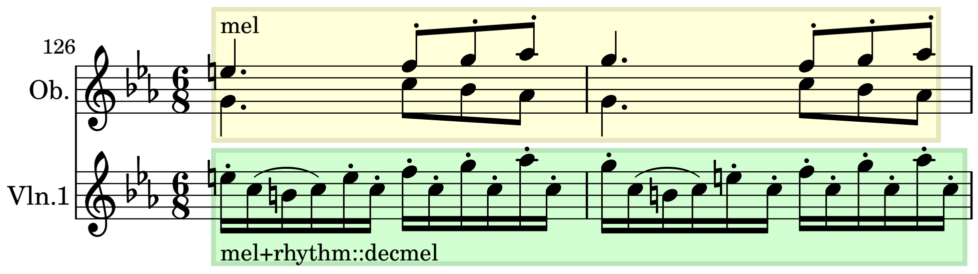


```
[323-330]               (harm-h:Fg.Vln2.Vla)
[323-324,327-328]       a:(mel:Vln1)
[325-326,329-330]       b:(mel-u:Fl.Cl1) / (rhythm-u:Vc.Cb)    {CR 2 ab}
```

Figure 3.10.: *Allegro con brio* from Symphony No. 5 by Beethoven, second thematic zone in the recapitulation, measures 323-330, from edition [263]. The melody in unison `mel-u` is played in a Call/Response scheme (`<CR>`) between the first violins `a:{Vln1}`, and the flutes and the first clarinet in B♭ `b:{Fl.Cl1}`. There is with the latter group a rhythmic layer in unison `rhythm-u` on the cellos and the contrabasses `{Vc.Cb}`, playing the famous rhythmic pattern of that symphony. There is a sustained harmony layer, in homorhythm, on the bassoon, the second violins, and the violas `(harm-h:Fg.Vln2.Vla)`.
`http://algomus.fr/fm/beethoven5-323.mp3`

or counter-beats. Such a layer should include at most two notes in a 4/4 or 6/8 measure, and one note in 2/4 or 3/8, separated by rests.

- mel+rhythm::decmel. Called "decorative variation of the melody" by Piston [211, p. 371], "harmonico-melodic phrases" by Rimsky-Korsakov [223, p. 99], or "heterophony" [182], such phrases combine at a single instrument the melody with a



`[126-127] (mel-h:Ob) / (mel+rhythm::decmel:Vln1)`

Figure 3.11.: *Allegro con spirito* in the first movement of Symphony No. 103 by Haydn, measures 126-127, from edition [212]. The main melody is played by the first oboe (Ob), while the first violins (Vln1) are playing a "decorative variation of the melody" [211], consisting in the same melody played in sixteenth notes with ornamentations on counter-beats.
http://algomus.fr/fm/haydn103-126.mp3

more (or less) "rhythmic" texture (Figure 3.11). We annotate such a layer when the "underlying" melody is found in another layer. Otherwise an ornamented melody is still annotated as mel.

Other combinations such as mel+harm and mel+rhythm+harm are possible, but rarely used. Parameters such as meter and tempo have an influence on the assigned role. On slow movements, for example, or on the slow introductions to first movements, a "significant part of a measure" or a "number of notes in a measure" can be rather evaluated on only a portion of the measure. That is to say, that the indications in the annotation guide must be musically interpreted.

3.1.3.2. Relations inside a Layer

The principle of rhythmic synchrony is fundamental for instrumental blending and the formation of layers. Then, (almost) every layer exhibit **homorhythm** as relation between its constituent parts. We annotate it with `h`, following the relations description in [89]. An example of this is the (almost) tutti chords in Figure 3.9. The relation can be stronger in some cases, and we can find **parallel motion** (`p`), often in thirds or in sixths, or in octave or **unison** doubling (`u`) (second violins and violas in Figure 3.9). We do not make distinction between octave and unison in our annotations. We annotate the prevalent relation, that does not have to be perfect. Such relations are tagged when they apply to most of the parts: a parallel motion can be partly in thirds, partly in sixths, and include other notes [89]. Relations go from the weakest (`h`) to the strongest (`u`), with all unison doublings being also parallel and homorhythmic. In our annotations we mark only the strongest relation that applies to a layer. So, `h` means that there is homorhythm without parallelism and unison. The **no relation** tag `0` is applied to layers formed by one instrument alone.

Formally, given a set of inside-layer relations ρ, such as $\rho = \{\mathtt{0}, \mathtt{h}, \mathtt{p}, \mathtt{u}\}$, we can chose it as the set of descriptions. Then, the description function

$$\begin{aligned} \delta_{relations} :& \mathcal{L} \to \rho, \\ & \ell_i \mapsto r, \end{aligned}$$

assigns to every layer ℓ_i, where $i \in \mathrm{ID}$, a relation $r \in \rho$. With this description, each layer has an inside-layer relation (Figure 3.12).

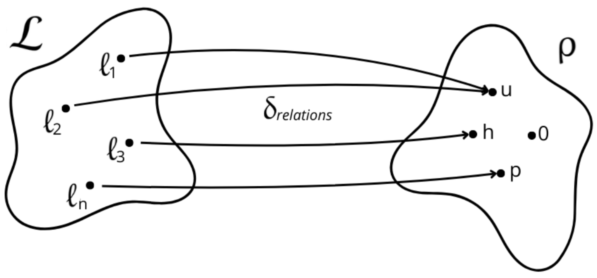


Figure 3.12.: Layer description: relations. All layers can be mapped to a relation from $\rho = \{\mathtt{0}, \mathtt{h}, \mathtt{p}, \mathtt{u}\}$.

3.1.3.3. Relation between layers: Orchestral Effects as Meta-Layers

The mechanisms of *segmental* grouping, can make several orchestral effects emerge from the succession of layers with varying orchestration [91]. In this study we focused on the formation of layers, but we have included some of these effects in our annotations. The syntax allows to encode them as Meta-Layers, *i.e.* relations between different layers. One example is "Call-and-response" (CR) schemes (see Figure 3.10). Called "transference of passages and phrases" by Rimsky-Korsakov [223, p. 107], or "antiphonal writing" by Adler [4, p. 273], they are formed by sequences of concurrent layers in which the instrumentation and/or the roles of involved instruments periodically change. We annotate call-and-response schemes with `CR(f)`, where `f` is the frequency of alternation between the two component layers, expressed in number of measures.

We can formally describe meta-layers relations as descriptions given to sets of layers.

Definition 3.9 (Meta-Layer Description Function)**.** Let $\mathcal{L} = \{\ell_i\}_{i \in \mathrm{ID}}$ be a valid collection of layers. Given a set of descriptions M, a meta-layer description function is a function

$$\begin{aligned} \mu :& 2^{\mathcal{L}} \to 2^{M}, \\ & L \mapsto d, \end{aligned}$$

that assigns a set of descriptions $d \subseteq M$ to every subset $L \subseteq \mathcal{L}$.

We are not really interested in detailing a meta-layer relation for every subset $L \subseteq \mathcal{L}$, so the function can take as value the empty set $\emptyset$.

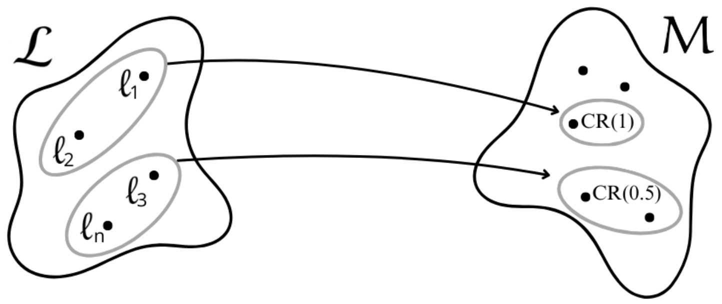


Figure 3.13.: Layer description: meta-layers. All sets of layers from $2^{\mathcal{L}}$ can be mapped to a point of 2^{M}, *i.e.* to a set of meta-layer relations from $M = \{\mathtt{CR(1)}, \mathtt{CR(0.5)}, \ldots\}$, or to the empty set $\emptyset$. Here, ℓ_1 and ℓ_2 are part of a call-and-response scheme with a frequency of 1 measure; the CR scheme between ℓ_3 and ℓ_n has instead a frequency of half a measure.

3.1.4. Properties of Layers and Orchestral Texture

In this section we define two properties of orchestral layers that can seem trivial at first, but that will be helpful to make analogies when defining abstract layers as components of Layer Score in Section 3.2. We can define the span and the total instrumentation of a layer, by projecting it on $\mathcal{M}$ and on $\mathcal{E}$.

Definition 3.10 (Span and Total Instrumentation of a Layer)**.** Given an orchestral layer $\ell \subset \mathcal{M} \times \mathcal{E}$ we can define its span as its projection on $\mathcal{M}$.

$$\mathrm{span}(\ell) := P^{\perp}_{\mathcal{M}}(\ell) \subset \mathcal{M}$$

The total instrumentation of a layer will be instead its projection on $\mathcal{E}$.

$$\mathrm{ti}(\ell) := P^{\perp}_{\mathcal{E}}(\ell) \subset \mathcal{E}$$

We can visualize the projection in Figure 3.14. We can now also attempt to give a definition of the orchestral texture of a measure.

Definition 3.11 (Orchestral Texture)**.** Given a valid collection of orchestral layers $\mathcal{L}$, the orchestral texture τ^{m} at measure m is the set of all orchestral layers $\ell \in \mathcal{L}$ such that $m \in \mathrm{span}(\ell)$.

3.1.5. Orchestral Layers and OrchARD Blends

In Section 2.5 we introduced a few corpora of orchestral music, among which the OrchARD database, which contains orchestral scores on which several perceptual phenomena have been annotated, following McAdams et al.'s Taxonomy of Orchestral Grouping Effects (TOGE) [178] (see also Section 2.1.2). McAdams's orchestral blend is a very similar concept to the orchestral layer we have defined in this chapter. In fact,

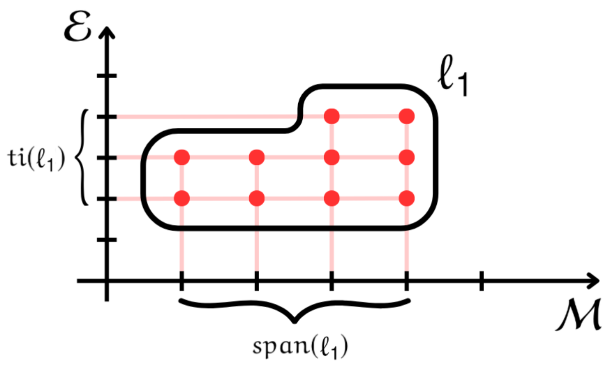


Figure 3.14.: Layer Projections: Span and Maximal Instrumentation of a Layer. The span of layer ℓ_1 is its projection on the measures axis $\mathcal{M}$ and its total instrumentation is its projection on the ensemble axis $\mathcal{E}$.

we can say that formally, they can be defined exactly in the same way, so, following Definition 3.5, an orchestral blend β is an orchestral layer

$$\beta \subset \mathcal{M} \times \mathcal{E},$$

whose elements are couples of measures and instrumental parts.
The difference between layers and blends is given by the criteria with which instruments are grouped. The model for layers annotations that we proposed here, and that we used to annotate a corpus of first movements of classical symphonies (Chapter 5) is rooted in ideas based on perceptual principles [91], including musical streams [176] and Cambouropoulos's perceptual voice [31], but also on music theoretical ideas on texture and on the roles that layers and voices serve in a composition in the classical style [18].
The orchestral blends annotations in OrchARD were build to study perception, so they focus on the timbre of the sounds, and they are based on a reference recording for which the score serves as support.
The biggest difference between the two models is that, in our model, onset synchrony or synchronicity [234, 31] is the primary factor that gives raise to homorhythmic layers. In the special cases when the different instruments present also parallelism and/or unison, they are always part of the same layer. This is not always the case for blends, because even instruments that are playing exactly the same part might not blend well because of timbre characteristics, or the musicians' performance, or the recording room and equipment. An example of non blend due to timbral heterogeneity is what happens between measure 183 and 186 of the third movement of Debussy's *La Mer*: the english horn (Eh.), the horn 1 (Hn in F 1 in this score), and the glockenspiel (Glock.) are playing the same part, but the glockenspiel's timbre does not blend well with that of the other two instruments (see Figure 3.15).
In the example in Figure 3.15 we have a blend $\beta_{\text{Figure 3.15}} = \{m_{183}, m_{184}, m_{185}, m_{186}\} \times \{\text{Eh.}, \text{Hn in F 1}\}$, but we might want to group together the english horn, the horn 1, and the glockenspiel in one layer $\ell_{\text{Figure 3.15}} = \{m_{183}, m_{184}, m_{185}, m_{186}\} \times \{\text{Eh.}, \text{Hn in F 1}, \text{Glock.}\}$. Then we have that $\beta_{\text{Figure 3.15}} \subset \ell_{\text{Figure 3.15}}$, because the layer includes all instruments

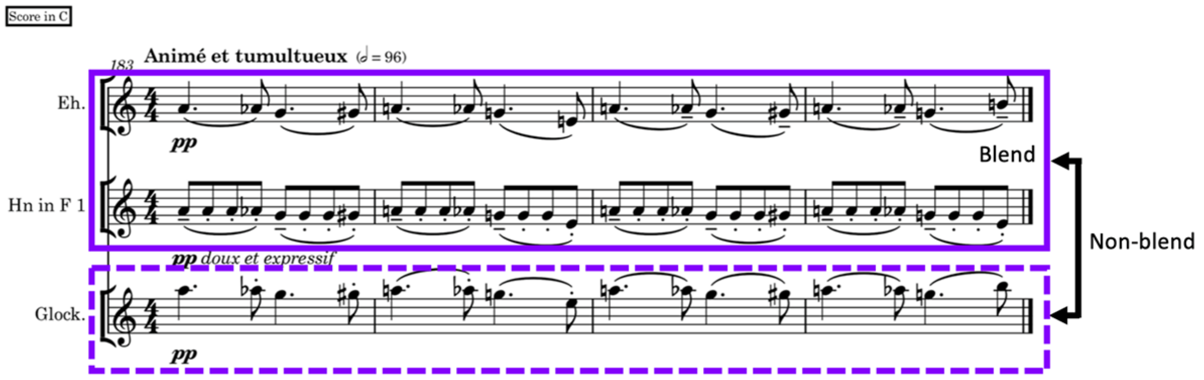


Figure 3.15.: Third movement of *La Mer* by Debussy, measures 183-186, figure reproduced from [178]. The glockenspiel (Glock.) does not blend with the english horn (Eh.) and the horn 1 (Hn in F 1), and its timbre is still well identifiable in the mixture.
`http://algomus.fr/fm/debussy-la-mer-iii-183.mp3` (audio from [178])

that are playing the "same part", even though there is timbral heterogeneity.
To further link these two models we could describe the blending components of a layer through a new layer description function that would map every layer to a set of blends. Conversely, we could introduce blend description functions in a similar way. They could be used to describe the blend type from the TOGE, and to map each blend to its superset layer.
Another difference is that we annotate also layers constituted by one single instrument. Instead, a group of one instrument is never considered as a blending, since this is a property of a group of two or more instrumental parts. The result is that layers span the entire duration of a score, while blends are only annotated when two or more instruments are fusing their timbres, giving rise to more sparse annotations. At every measure, our layer annotations on the classical symphonies form a partition of all of the instruments that are not silent. The same does not happen with blends in OrchARD, as not all playing instruments contribute to a blend.
Blends in OrchARD also have their specific description, different from the layer role and relationship descriptions given by our model. For example a timbral augmentation blend α can be seen as the union of two sub-blends: a dominating component $d \subset \mathcal{M} \times \mathcal{E}$ and an embellishing component $e \subset \mathcal{M} \times \mathcal{E}$, *i.e.* $\alpha = d \cup e$.
In the rest of the thesis, we mainly adopt the textural point of view in defining and studying layers, except for Chapter 6, which takes the timbre and perception point of view to study blends in OrchARD.

3.2. The Layer Score

An orchestral score can be analyzed at different levels, like other music objects and concepts. Taking an orchestral score as a "neutral level" [201], the analyst can examine orchestration techniques. We have focused on *textural/instrumental* aspects, and we have presented in Section 3.1 a taxonomy to analyze how orchestral music can be split into a number of layers with a predominantly melodic, rhythmic, or harmonic role, or a mix of them. Layers are highlighted by combinations of instrumental timbres [149], and auditory grouping processes favor the emergence of orchestral effects [178]. Describing and modeling such high-level abstract concepts is a challenge for (digital) musicology and music analysis. In this section we propose a model that may be used as an intermediate step when composing or orchestrating music, in addition to being a useful concept in music analysis.

We start by presenting some ideas that inspired this model in Section 3.2.1, then we present the layer score (Section 3.2.2), and finally we formalize it in Section 3.2.3.

3.2.1. Behind Orchestral Music: Composer/Orchestrator Sketches

Any music, any score, may be seen as a rendering of high-level *musical ideas*, that may be used as intermediate steps when composing or improvising music, or may serve analytical purposes. Orchestral layers and effects are no exception, but are they planned by the composer? Were these layers and effects present in the composer's – or the orchestrator's – mind when envisioning the piece? It is known and documented that some composers develop sketches at some point of their workflow. These can contain rough ideas, themes and motives, incomplete melodies and harmonies, that can be further developed and serve as inspiration for writing more complete and elaborate music [233, 281]. In particular, Beethoven has left us sketches that have been extensively studied. They contain precise musical ideas such as patterns, themes, and sometimes orchestration drafts [129]. De Sousa [59] presents the idea of *textural* spaces, arguing that composers are encouraged to explore the options and possibilities offered by the orchestra by moving in these spaces. The model is powerful, but it is unclear weather it might correspond to conscious or unconscious ways in which composers model complex structures with intricate layers in their mind.

These raw sketches can thus contain, in some cases, germinal ideas of concurrent musical parts and layers. It is a rough plan that preexist the orchestral score, in which the composer expand these initial plans. There are some MIR studies that have explored the idea of construction from sketches. In the realm of symphonic music Gotham et al. [95] presented a co-creative experiment in which they created a plausible version of a Beethoven's 10th Symphony, letting a composer interact with AI models to build the score starting from original sketches by Beethoven himself[9]. In this process they

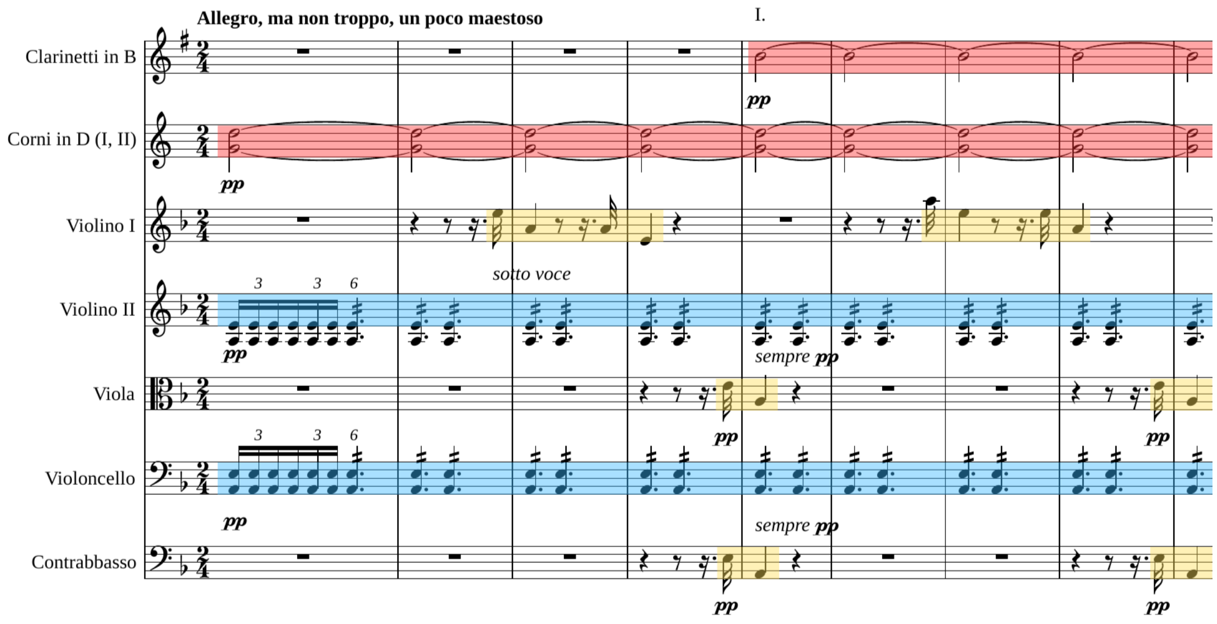
[9]Digitized versions of some of Bethoven's sketches preserved at Beethoven-Haus in Bonn can be consulted at `https://www.beethoven.de/en/archive/list/node/5179594084712448/Sketches`

introduce a *short score* as a set of raw materials before orchestration. Somehow, in jazz/pop styles, *lead sheets* with melodies and chords may also be seen as a condensed version of a music piece – or could correspond to a sketch of the final song. MIR and AI methods aiming at (co-)creating such music often generate such lead sheets at first, then proceed with accompaniment generation [118], even if it is debatable whether this way of proceeding can be considered good practice in pop music composition, and an end-to-end generation is sometimes preferred [5].

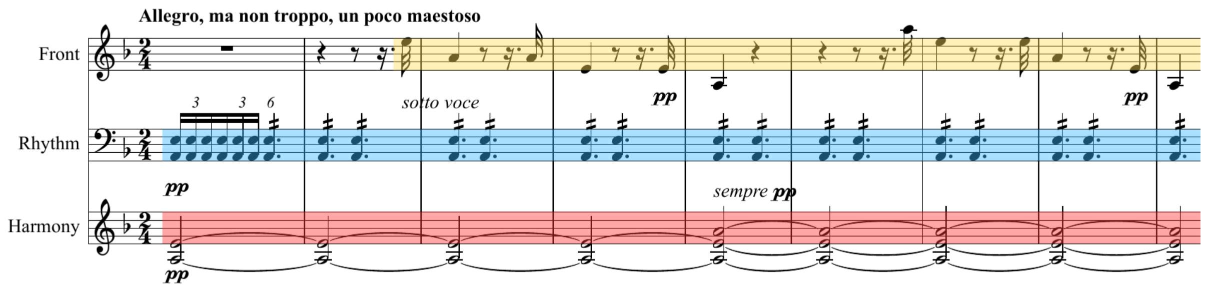
3.2.2. Defining and Modeling a Layer Score

Inspired by such hypothetical intermediary "sketch scores" and by analytical considerations, we define the *layer score* [165] as a score with a variable number of *layers*, each one with a given *role*, following the description of roles we introduced in [149] and Section 3.1 for classical-romantic orchestral music. Such a layer score has no indication on the instruments that should play the layers, making it an abstract version of a music piece, for which many realizations are potentially possible in terms of instrumentation. On the opening of the Beethoven's 9th Symphony, we analyze three layers with different roles, ℓ_{front}, ℓ_{rhythm}, and ℓ_{harmony} (Figure 3.16a). Those layers are distributed on several instruments in the full orchestral score. Here, one part in the layer score (Figure 3.16b) roughly corresponds to one layer in the orchestral score, but the actual music in the orchestral part could differ more from the content of the layer score, possibly depending on the capabilities of the instruments chosen to render a layer. For example, rhythmic motion could be rendered with a different density of notes if reproduced through a timpani roll, or through tremolo strings *sul ponticello*.
At the opposite, if we consider now a *piano reduction* of an orchestral score, the same layers will be blended together into the two staves of the piano reduction – a single pianist should be able to play it. In the layer ℓ_{rhythm} of the opening of the Beethoven's 9th Symphony, the lower strings *repeat* chords made of A and E, whereas the piano reduction by Liszt *alternates* between the same pitches (Figure 3.16c). Again, the actual music in the piano reduction could be different, in the rhythm organization for instance. In practice, the octaves often differ in piano reduction, the pianist hands not having the same ambitus as the orchestra. In some cases, the actual pitches may even differ between instrumentations, for example including patterns or scales in some of them.
In terms of size we can observe that the layer score is in between the orchestral score and its piano reduction, it has generally fewer staves than a full orchestral score and more staves than a piano reduction. But we can rather interpret it (anachronistically) as a *common ancestor* between the two. We can speculate, exploring uncertain territory, that the layer score contains the most basic "essence" of the music, whatever that might mean, but not the actual music rendering. In any case, it includes structural, melodic, harmonic content, and some of the textural content, but it is disconnected from any aspect related to instrumentation. It is *potential music* that can be reduced to fit the piano, expanded to the symphonic orchestra, or adapted to any other type instrumentation. This concept of layer score allows thus to decouple composition from instrumentation and orchestration (although this decoupling may be artificial) and to

(a) **Full Orchestral Score.** 13 tracks/instruments (7 in this extract)
Transcribed by ClassicMan on `musescore.com`
Textural labels from [149]. *Front:* Vln1, Vla, Cb; *Rhythm:* Vln2, Vc; *Harmony:* Cl, Hrn



(b) **Layer Score.** 3 layers



(c) **Piano Reduction.** Arr. by Franz Liszt, ed. Breitkopf & Härtel

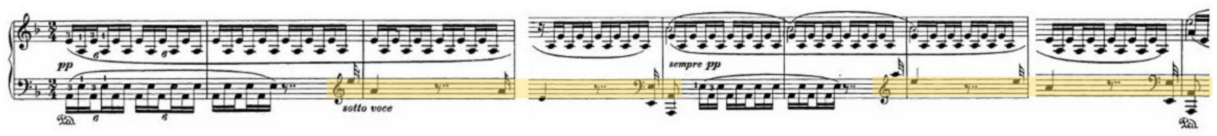


Figure 3.16.: First eight measures from the first movement of Symphony No. 9 by Beethoven, op. 125. (a) The orchestral score can be decomposed into three layers: one with an *harmonic* role (red, dark), one with a *rhythmic* role (blue, light), and a third one with a *front* role, that is nevertheless difficult to categorize as a melody (yellow, very light). (b) The layer score contains one part for each of these three layers, λ_{harmony}, λ_{rhythm}, and λ_{front}. (c) In the piano reduction, the harmonic and rhythmic layer are blended together and rendered with a *pianistic* texture, which is different from the orchestral version. The left hand alternates between that harmonic-rhythmic layer, and the front layer.

`http://algomus.fr/fm/beethoven9-orchestra.mp3`
`http://algomus.fr/fm/beethoven9-piano.mp3`

devote our attention mainly to the latter ones.

Back to the question of composer sketches, a layer score is not intended to reproduce an existing compositional practice and we do not claim either that writing such layer scores would be a desirable practice. Writing music for the orchestra (or for any other instrument) can be a non-linear process, with iterations between high-level ideas and actual music content. For example, the constraints given by the ambitus of each instrument influence the composition itself, forcing the composer to rethink some of the choices they have already made, and even to rewrite major sections of the composition.

What we are formalizing here is a *new* object that can help separate certain aspects of symphonic writing, and give new perspectives in understanding and modeling orchestration techniques, both for analysis and for computer-assisted creativity. For example, piano reduction and orchestration from piano are symmetrical tasks. Orchestration from piano usually requires "de-pianotizing" the piano music, including voice separation [166] or texture analysis [49] – as piano reduction requires "de-orchestrating" the orchestral music. With a layer score, we have an object with "de-pianotized" and "de-orchestrated" music, allowing to develop computer assisted processes with intermediate steps that can increase the possibilities for human intervention. The following chapters will detail these steps, in particular Chapter 4 illustrates a framework for the computer-assisted orchestration of a piano score passing through a layer score.

3.2.3. Formal Formulation of a Layer Score

Piano Scores and Layers

We have seen that a layer is rendered differently in an orchestral score, in a piano score, and in the *layer score*. In particular with the number of staves involved. We remind that an orchestral score is a set of orchestral notes in the space $\mathcal{T} \times \mathcal{P} \times \mathcal{D} \times \mathcal{E}$. We can give an analogous definition for a piano score, even though piano music is not the focus of this thesis.

Definition 3.12 (Piano Notes and Piano Score)**.** Let $\mathcal{T} = [0, t_{end}] \subset \mathbb{R}$ be the time, let $\mathcal{P}' = \{A0, A\sharp0, \ldots, C8\} \subset \mathcal{P}$ be the pitches of the grand piano keybord, and let $\mathcal{D} = \mathbb{Q}$ be fractions representing the durations of notes. A piano note is defined by $(t, p, d) \in \mathcal{T} \times \mathcal{P}' \times \mathcal{D}$. Then a piano score P is a subset $P \subset \mathcal{T} \times \mathcal{P}' \times \mathcal{D}$.

Now, we can define what is a piano layer, but we won't enter into difficult discussions about piano texture [49]. We do not even introduce measures here since, even if we impose layer boundaries to happen at measure changes, we cannot be sure that the measure is enough of a temporal coordinate to separate layers (see Figure 3.17). We take than the safest approach and define a piano layer as a subset of the score, which gives us the following.

Definition 3.13 (Piano Layer)**.** Let $P \subset \mathcal{T} \times \mathcal{P}' \times \mathcal{D}$ be a piano score, where $\mathcal{T} = [0, t_{end}] \subset \mathbb{R}$ is the time, $\mathcal{P}' = \{A0, A\sharp0, \ldots, C8\} \subset \mathcal{P}$ are the pitches, and $\mathcal{D} = \mathbb{Q}$ are

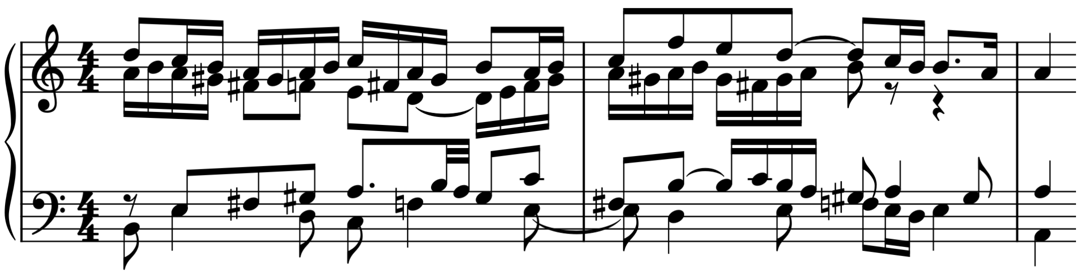


Figure 3.17.: Fugue No. 1 in C major from the Well-Tempered Clavier, by Bach, BWV 846, measures 12-13. Measure and pitch are clearly not enough to distinguish the four voices [98]: the highest note for the bass in the first measure (F3) is higher in pitch than the lowest note of the tenor (E3). Even with time and pitch, the four voices cannot be completely distinguished: the unison on the E3 (second note on the lower staff) is an example of a pitch at a certain time that belongs to two different parts (bass and tenor). In this case duration allows to distinguish the two notes belonging to different voices.

fractions representing the durations of notes. Then, a piano layer π is

$$\pi \subset \mathrm{P} \subset \mathcal{T} \times \mathcal{P}' \times \mathcal{D},$$

whose elements are tuples of time positions, pitches, and durations.

It is not sure, whether piano layers must be pairwise disjoint to be valid. It might be that a note belongs to multiple layers, even if it is not graphically repeated two times in the printed score. Once again, we leave the discussions about piano layers and validity to further studies. Whatever definition we take for a piano layer and its validity, once we have a valid collection of piano layers π we can connect layers in piano scores and orchestral scores for the same piece. Then, orchestration and piano reduction would be the following transformations.

$$\mathrm{P} \subset \mathcal{T} \times \mathcal{P}' \times \mathcal{D} \xrightarrow{\textit{orchestration}} \mathrm{O} \subset \mathcal{T} \times \mathcal{P} \times \mathcal{D} \times \mathcal{E}$$

$$\mathrm{P} \subset \mathcal{T} \times \mathcal{P}' \times \mathcal{D} \xleftarrow[\textit{reduction}]{} \mathrm{O} \subset \mathcal{T} \times \mathcal{P} \times \mathcal{D} \times \mathcal{E}$$

We prefer to introduce the layer score to better understand these operations.

Layer Score

In a layer score we want to make some abstraction, and avoid considering the instrumentation. How can we then define a layer in that case? One option is to define it simply as a set of notes.

Definition 3.14 (Abstract Layer)**.** Let $N \subset \mathcal{T} \times \mathcal{P} \times \mathcal{D}$ be a set of notes, where $\mathcal{T} = [0, t_{end}]$ is the time, $\mathcal{P}$ is the set of possible pitches, and $\mathcal{D} = \mathbb{Q}$ are fractions representing the durations of notes. We define an abstract layer λ as a set of notes:

$$\lambda \subset N \subset \mathcal{T} \times \mathcal{P} \times \mathcal{D}$$

A collection of abstract layers Λ as is a set $\Lambda = \{\lambda_i\}_{i \in ID}$.

The abstract layer has some properties, for example the span. The span of an abstract layer is similar to that of an orchestral layer (Section 3.1.4) (and it would be similar to that of a piano layer, if we had defined it) and it is an interval of measures,

$$span(\lambda) = [m_{beg}, m_{end}] \subset \mathcal{M},$$

such that $\forall (t, p, d$ that belong to λ, the onset $t \in [m_{beg}, m_{end}]$. Moreover there must exist notes whose onset is in m_{beg} and in m_{end}. It is possible then to identify each orchestral layer or piano layer with an abstract layer with the same span. We can also apply layer description functions to abstract layers. We are now ready to define the layer score as a score in which we replace the instrumentation information $\mathcal{E}$ with abstract layers.

Definition 3.15 (Layer Score)**.** Let $\mathcal{T} = [0, t_{end}]$ be the time axis, $\mathcal{P}$ be the set of possible pitches, $\mathcal{D} = \mathbb{Q}$ be fractions representing the durations of notes, and let $\Lambda = \{\lambda_i\}_{i \in ID}$ be a set of abstract layers. Then, we can define layer notes as $(t, p, d, \lambda) \in \mathcal{T} \times \mathcal{P} \times \mathcal{D} \times \Lambda$, and a layer score L as a subset $L \subset \mathcal{T} \times \mathcal{P} \times \mathcal{D} \times \Lambda$.

Basically, a layer score looks very similar to an orchestral score, with the difference that the staves of a layer score correspond to layers instead of instruments (Figure 3.18). The fourth dimension of an orchestral score is the ensemble $\mathcal{E}$, that of a layer score is the layers Λ, while a piano score has only three dimensions in our definition. Whoever is writing a layer score is responsible for creating meaningful layers, and they should be coherent in themselves. If the layer score is the first step of a composition, then layers could provide a plan for perceptual streaming. If the layer score is created from a piano score or an orchestral score, then the layers can be used to represent the original piano/orchestral layers.
Analogously to what we have done for orchestral texture, we can also define abstract texture.

Definition 3.16 (Abstract Texture)**.** Given a collection of abstract layers Λ, the abstract texture τ^m at measure m is the set of all abstract layers $\lambda \in \Lambda$ such that $m \in span(\lambda)$.

The difference between orchestral texture and abstract texture is in the nature of the layers, orchestral layers have information about instrumental parts, and abstract layers do not.

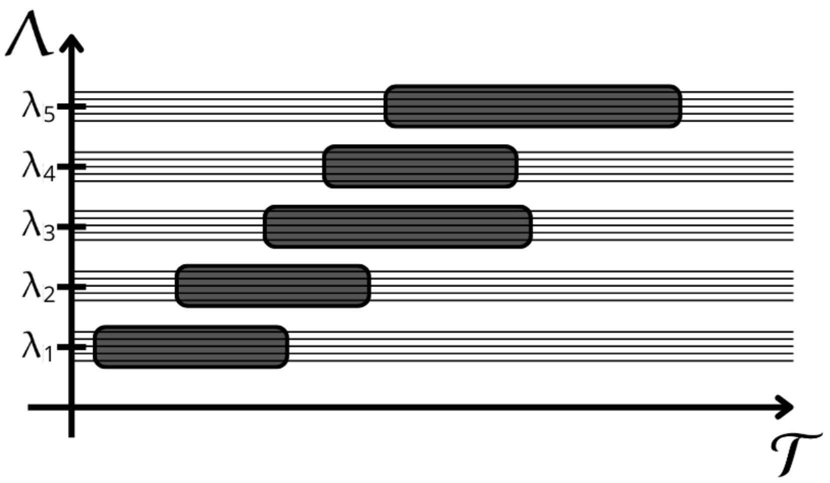


Figure 3.18.: In this schematized representation of a layer score, we can see that staves in a layer score correspond to the abstract layers. Black boxes indicate the portion of the staff corresponding to the span of the abstract layer, *i.e.* where there are notes.

3.3. The Orchestration Plan

Planning is omnipresent in music writing. In masterpieces nothing is left to the chance, but every element is meticulously curated to perfectly fit together with each other. The same goes for orchestration. In a good orchestration everything is perfectly balanced, and the instrumentation is evolving during the piece, telling a story, keeping the listener focused, and refreshing the ears with new interesting surprises. In this section we propose a model, the *orchestration plan*, which can be used to describe potential instrumentation and orchestration, favoring the development of computer assisted orchestration pipelines.

We start by discussing the inspirations from traditional practice (Section 3.3.1), then we formally define the orchestration plan (Section 3.3.2), we discuss the differences and similarities between layer annotations and orchestration plans (Section 3.3.3), and we conclude by showing how orchestration plans allow the transformation of abstract layers into orchestral layers (Section 3.3.4).

3.3.1. Planning in Traditional Orchestration Practice

Students in the orchestration class at the Conservatoire de Lille, which I had the pleasure of observing for the past two and a half years, have a project to complete at the end of their first year. For this project they choose an existing composition, usually for piano solo, and they write an instrumentation of the piece for an ensemble with 2-10 instruments. On their second year they have a similar exercise, in which they select a Beethoven piano sonata, and they orchestrate it for a classical orchestra, in the style of Beethoven. Before starting to write the actual arrangement/orchestration, the teacher, Vincent Paulet, is asking them to think ahead and present an *instrumentation/orchestration plan*. The form in which this plan is created is free, and can be more or less structured. I have observed that, in general, the students write some annotations on the original piano score, concerning, for example, the entrances of the instruments. These annotations naturally segment the piece into sections with different instrumentation. This allows them to have better control over the effects of novelty or contrast they want to achieve, and reserve some instruments for certain sections in which they would fit well. The choices are made on the basis of ambitus and timbre of the instruments. These plans might also go as far as highlighting certain voices in the score, similarly to what would happen in a layer score, and to connect them with arrows to one or more instrument names.

Belkin [15] identifies five key points that should be planned in an orchestral work: changes of sounds, accents, cadences, progressions, and gradation of climaxes. *Changes of sounds* and timbre must be logical in the context of phrases and motives. *Accents* that require to momentarily add sounds or change playing techniques, must do so in proportion to the desired effect and the context. Structure and *cadences* can be highlighted with orchestration. *Progressions* are evolutions that occur over a span of time and can correspond to an evolving texture. Finally, with *gradation of climaxes*, Belkin suggests that one climatic moment towards the end of the piece should carry more emphasis and "stand out" with respect to the other ones.
In the context of the functional orchestration model, F. Lévy [164] defines a *functional operation* as a "*musical aim or goal, verbalized or not verbalized by the composer, but clearly in their mind before starting to write, which helps them to choose a particular technique to achieve a particular perceptual effect*". This means that composers and orchestrators are planning things like contrasts, merging entities, layers, and other effects.

We take inspiration from all these ideas and we propose a formalization of the concept of *orchestration plan*, in which we explicitly divide the piece in segments, and we describe the instrumentation in each segment, allowing to be so precise as to indicate a group of instruments for each layer of the composition. Our model is thought to accompany a layer score to describe the *potential orchestration* that can emerge from them, but it is possible to use it without any reference layer score. It can be used at different levels of detail, by simply indicating the instruments in each segment or by going as precise as the "measure level", ultimately making the plan look like the layer annotations (Section 3.1) for a yet to create score. Those objects remain conceptually different, as discussed in Section 3.3.3.

3.3.2. Formal Formulation of the Orchestration Plan

Our first definition of an orchestration plan was given in [165]. In the context of that project, an orchestration plan was used together with an handmade layer score in order to make a co-creative orchestration of existing piano pieces from the suite *Angeles* by

Gissel Velarde. Here we give a slightly more formal definition of such an orchestration plan, that can in principle be implemented and generated in several different ways. We postpone to Chapter 7 the description of how the implementation and the generation were done in the *Angeles* project.
The goal of the orchestration plan is to assign instruments to layers. In order to define it we need to introduce some other objects before. We start by defining an orchestration segment, as a partition of time, alternative to measures, and less fine-grained.

Definition 3.17 (Orchestration Segment)**.** Let $\mathcal{T} = [0, t_{end}]$ be time, and let $\mathcal{M} = \{m_1, m_2, \dots, m_N\}, N \in \mathbb{N}$ be a partition of $\mathcal{T}$ into measures. Then time $\mathcal{T}$ can be partitioned into disjoint intervals $\mathcal{S} = \{s_i\}_{i=1,\dots,S}$, such that each interval is the union of one or more consecutive measures, *i.e.* $s_i = \bigcup_{j \in \mathcal{N}_i \subset \mathcal{M}} m_j$, and that their union is equal to the entire set $\mathcal{T}$, *i.e.* $\bigcup_{i=1,\dots,S} s_i = [0, t_{end}]$. We call the intervals $\mathcal{S} = \{s_i\}_{i=1,\dots,S}$ orchestration segments.

For our definition of segments all that we require is that the segmentation is a partition of time in wider parts than measures. The musical meaning comes from comparing the segmentation with a layer score. We can assume that a piece is made up of different segments where *texture*, which is the combination of simultaneous layers, does not change. This structure would look like the one shown in Figure 3.19. In fact, it can be difficult to fix a precise boundary between segments, since the transition from one texture to the next one can be smooth and progressive, and it can happen over several measures. The segmentation with rigid boundaries is indeed an approximation, but we consider that it serve our purposes, if we use it cautiously and we consider it more like a logical division than a time division in the applications. In a process of orchestration from a layer score, for example, the segmentation can be done at the relevant points where orchestration changes are wanted. So a segment is a set of measures on which the envisioned orchestration should not change.
The same segmentation can be used with a full orchestral score. As for the layer score, the orchestral layers are expected to remain constant during a segment. The instrumentation of each layer, instead, can evolve. We recall that the layer instrumentation at measure m is the set of the instrumental parts that belong to the orchestral layer at measure m (Definition 3.6). Over the measures of a segment, the layer instrumentation might be constant, but might also change, progressively getting bigger (orchestral *crescendo*) or smaller (orchestral *diminuendo*). We can then define the layer instrumentation at a segment.

Definition 3.18 (Layer Instrumentation at Segment s)**.** The layer instrumentation for an orchestral layer ℓ_{id} at segment s is

$$\ell i^{s}_{\mathsf{id}} = \bigcup_{m \subset s} \ell i^{m}_{\mathsf{id}},$$

i.e. the union of the layer instrumentations for ℓ_{id} at all measures m that belong to the segment s.

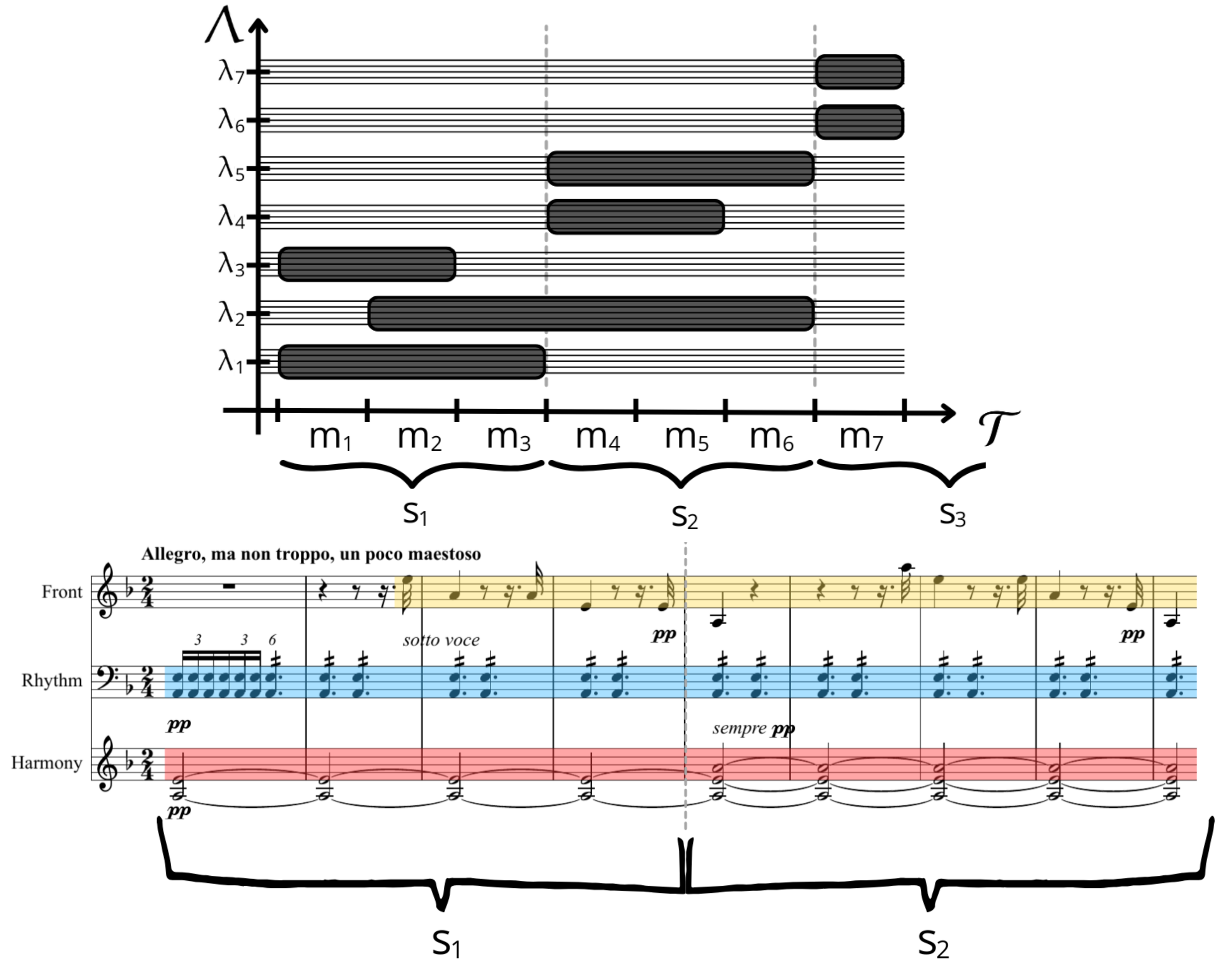


Figure 3.19.: On the top, a scheme of the segmentation of a layer score into orchestration segments. Segments are a less fine-grained partition of time than measures. On the bottom, a possible segmentation of the layer score for the beginning of Beethoven 9th Symphony. The second segment could start with the start of the upper A in the harmonic layer.

We can see an example of what this means in Figure 3.20.
The orchestration plan wants to mimic that for abstract layers in a layer score. In fact, through the orchestration plan we want to prescribe the layer instrumentation at segment s for all segments and all layers of a potential orchestral score that might be written starting from the given layer score.

Definition 3.19 (Orchestration Plan)**.** Let Λ be a collection of abstract layers, $\mathcal{S}$ be a segmentation of the piece, and let $\mathcal{E}$ be an orchestral ensemble. Then an orchestration plan ϕ is a function that assigns a set of instruments to every couple of layer and segment, *i.e.*

$$\begin{aligned}\phi : \Lambda \times \mathcal{S} &\to 2^{\mathcal{E}} \\ (\lambda, s) &\mapsto I \subset \mathcal{E}.\end{aligned}$$

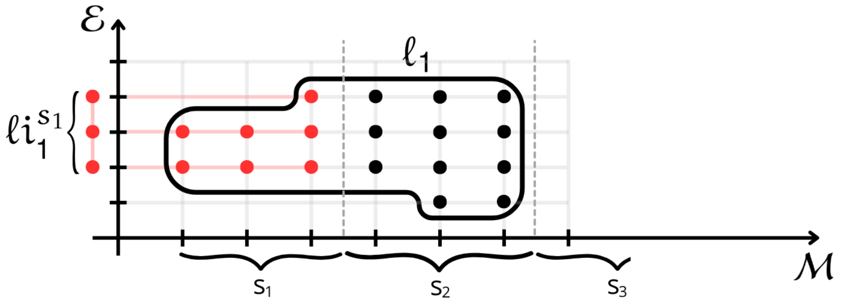


Figure 3.20.: The layer instrumentation $\ell i_1^{s_1}$ at segment s_1 of layer ℓ_1 is the union of the layer instrumentations ℓi_1^{m} for layer ℓ_1 at all the measures m of the segment s_1.

For example, an abstract layer λ_{mel} at segment s_1 could be assigned to cellos and contrabass $\phi(\lambda_{\mathsf{mel}}, s_1) = \{\mathsf{Vc}, \mathsf{Cb}\} \subset \mathcal{E}$. Notice that the set I can also be the empty set. For example the layer λ_{mel} might be silent at section s_5, and not be assigned to any instrument $\phi(\lambda_{\mathsf{mel}}, s_1) = \emptyset$.

Definition 3.20 (Segment Instrumentation)**.** Let Λ be a collection of abstract layers, $\mathcal{S}$ be a segmentation of the piece, and let $\mathcal{E}$ be an orchestral ensemble. Let $s \in S$ be a segment, and ϕ be an orchestration plan. Then a segment instrumentation SI^s at segment s is the set of the couples $(\lambda, \phi(\lambda, s))$ for all $\lambda \in \Lambda$.

$$SI^s = \{(\lambda, \phi(\lambda, s))\}_{\lambda \in \Lambda}$$

Defining the segment instrumentation for every segment of a piece, is equivalent to defining the orchestration plan, and in applications we have preferred listing segment instrumentations to describe the envisioned orchestration for the piece. An example from the *Angeles* project is the following. We have segment s_1 in which a set of simultaneous abstract layers $\Lambda = \{\lambda_{\mathsf{mel}}, \lambda_{\mathsf{rhy1+3}}, \lambda_{\mathsf{rhy2}}\}$ are present and playing in the layer score. We give the segment instrumentation SI^{s_1} in which each layer has its assigned set of instruments (see Figure 3.21).

$$SI^{s_1} = \left\{ \begin{array}{l} (\lambda_{\mathsf{mel}}, \{\mathsf{Vc}, \mathsf{Cb}\}), \\ (\lambda_{\mathsf{rhy1+3}}, \{\mathsf{Horn1}, \mathsf{Horn2}, \mathsf{Trp1}\}), \\ (\lambda_{\mathsf{rhy2}}, \{\mathsf{Fl1}, \mathsf{Fl2}, \mathsf{Cl1}, \mathsf{Cl2}\}) \end{array} \right\}$$

Figure 3.21.: Example of segment instrumentation from the *Angeles* project. Three layers $\lambda_{\mathsf{mel}}, \lambda_{\mathsf{rhy1+3}}$, and λ_{rhy2} are mapped to mutually disjoint sets of instruments.

We could also enforce the same validity condition as for the orchestral layers, and require the sets in one SI^s to be mutually disjoint. This is what has been done in the example of Figure 3.21, but it is not strictly required to obtain valid layers in an orchestral score.

3.3.3. Orchestration Plans and Texture Annotations: Differences and Similarities

The syntax of the orchestration plan is very similar to that of the orchestral layers annotations. It describes, for each segment, the layers using instruments in a predefined order and details the layers and roles (Figure 3.22). The orchestration plan can be less precise than the annotations: rather than assigning a layer instrumentation to every abstract layer at every measure, it will limit itself to assign a segment instrumentation to every segment, leaving more freedom to the instrumentation of each measure.

```
InstList: <Fl.Ob.ClBb.Fg|HrnF.TrpBb|Vln1.Vln2.Vla.Vc.Cb>
[p01] <2.2.|13|...mm>  1:rhy1:brass 2:rhy2:wood 3:rhy3:brass m:mel:mel2 (0)
[p02] <....|..|123mm>  1:rhy1:string 2:rhy2:string 3:rhy3:string m:mel:mel2 (4)
[p03] <....|13|.22m.>  1:rhy1:brass 2:rhy2:string 3:rhy3:brass m:mel:mel1 (0)
```

Figure 3.22.: An example of orchestration plan from the *Angeles* project. The plan starts by declaring the ensemble (`InstList`), where the instruments are separated by dots (`.`) and the instrumental families are separated by pipes (`|`). After that, each line represents a segment. For example, in the segment `[p01]` the segment instrumentation is comprised of four layers $\lambda_{\text{rhy1,brass}}$, $\lambda_{\text{rhy2,wood}}$, $\lambda_{\text{rhy3,brass}}$, and $\lambda_{\text{mel,mel2}}$. The layers are mapped to the instruments appearing in the order declared in `InstList`: For example, the "`<2.2.|`" bloc in the woodwinds refers to the layer instrumentation $\lambda_{\text{rhy2,wood}}$, with here flutes (Fl) and clarinets (ClBb). Dots (`.`) are here used as placeholder for silent instruments, while pipes (`|`) are again used to separate instrumental families.

We have seen that even if we don't have access to the *true* layer score, which also might have never existed, it is possible to construct a plausible layer score. The orchestration plan is another model, that together with the layer score, tries to describe orchestral music before orchestration. With those two models we separate the composer from the orchestrator, without concerning ourselves with being faithful to historical reality.
What would then be the orchestration plan for an existing classical symphony? This should be an object that relies on an existing segmentation of a layer score, and that assigns a group of instruments to every layer in every segment. The nature of this plan then depends on how we segment the piece. The annotations of the layers can be seen and interpreted as an orchestration plan, with segments of one or more measures. After all, there is a one-to-one correspondence between a group of instrument in a measure and the layer they are playing. So if the annotations can map the instruments to a layer, they can as well map the layer to the instruments. A segment in such orchestration plan correspond to a portion of the piece in which the instrumentation of each layer does not change.
While at this scale the annotations and the orchestration plan correspond and can

be represented by the same text file, they remain conceptually two distinct models, being one approximately the inverse of the other. The segmentation is chosen by the artist in the orchestration plan, while it is *imposed* by the score for texture annotations. The main focus of the annotations are the roles, and for the plan, the instruments. When annotating texture in an orchestral score we are answering the question "*what is there?*", an approach in which we analyze the score as the "neutral level" [201]. When creating an orchestration plan for an existing piece the question is "*how could the composer have created this?*", which is more open and subjective. The orchestration plan leaves more freedom to interpretation, while annotations aim at being more *objective* and quantitative.

3.3.4. Transforming Abstract Layers into Orchestral Layers

We have introduced all the objects that we propose as abstract models of orchestration, so we can now try to link them and to see what it means to perform orchestration with these objects. The basic transformation of orchestration that we can perform is that of *orchestrating a layer*. A piano layer $\pi \subset P$ is a set of notes. The same notes, taken in isolation, out of their original context in a piano score, constitute an abstract layer λ. The operation that we want to do in orchestration is to transform an abstract layer $\lambda \subset \mathcal{T} \times \mathcal{P} \times \mathcal{D}$ into an orchestral layers $\ell \subset \mathcal{M} \times \mathcal{E}$. The role of the orchestration plan function ϕ is to provide the missing information about the instrumentation I from the ensemble $I \subset \mathcal{E}$, at every segment $s \in \mathcal{S}$. A model can then collect this information and act on the notes to output an orchestral layer (see Figure 3.23).

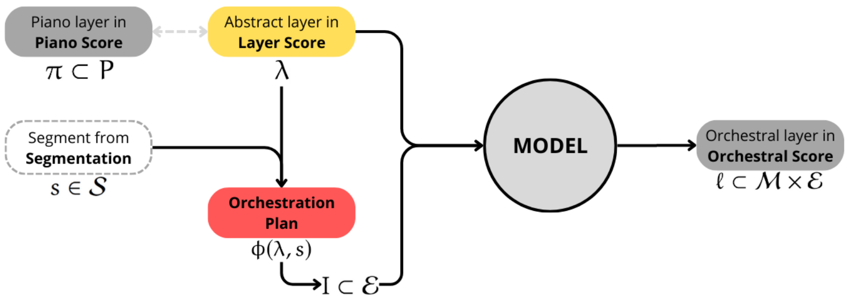

Piano layer in Piano Score
$\pi \subset P$
Abstract layer in Layer Score
λ
Segment from Segmentation
$s \in \mathcal{S}$
Orchestration Plan
$\phi(\lambda, s)$
$I \subset \mathcal{E}$
MODEL
Orchestral layer in Orchestral Score
$\ell \subset \mathcal{M} \times \mathcal{E}$

Figure 3.23.: An orchestration model is orchestrating the abstract layer λ into the orchestral layer ℓ following the instrumentation I decided by the orchestration plan ϕ for segment s.

In the next chapter we will discuss more in detail the possible transformations between abstract models and we will introduce a framework for computer assisted orchestration, using these transformation.

4. A Framework for Co-Creative Interaction in Orchestration

Traditionally, we think about composition as the act of creating music from scratch by a composer, a person sometimes described as *"a genius"*. However, this perspective oversimplifies the creative process. Composition is not always about creating something completely new but it often involves building on existing ideas, musical forms, or pieces. Music can be written for piano, for orchestra, for string quartet, for brass band, or for any other solo instrument and ensemble. Orchestration deals with transforming *existing music* from ensemble to ensemble. In traditional orchestration, two main transformations are typically considered, proceeding in opposite directions: orchestration from piano and piano reduction from orchestra, with some variants like reducing for piano four hands.

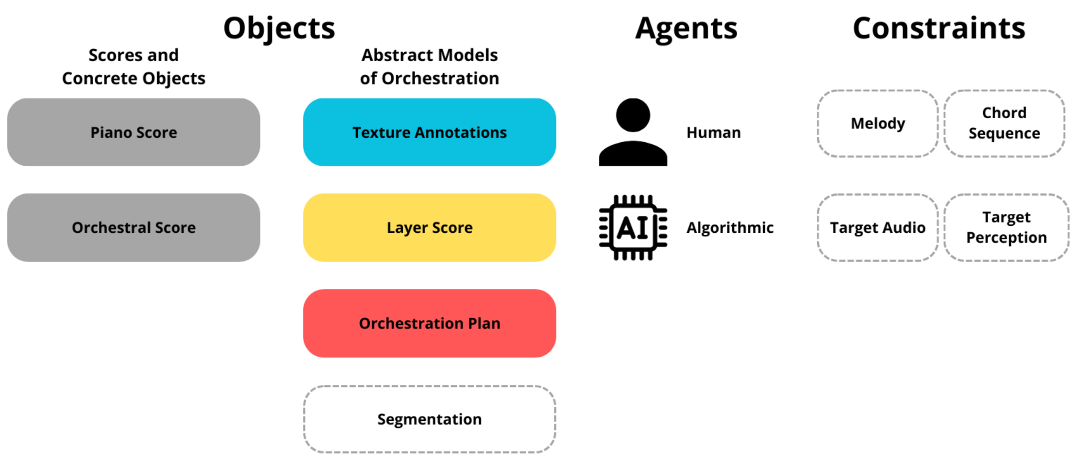


Figure 4.1.: The components of the framework: objects, agents, and constraints. Objects are divided into scores and abstract models of orchestration. Agents can be human or algorithmic. Some examples of constraints are shown in the figure.

With the help of the three abstract models just introduced we can reinterpret and classify human processes in traditional orchestration and tasks related to orchestration in MIR, and we can describe an environment in which human and algorithmic agents can interact to perform creative and analytic tasks. In this framework (Figure 4.1) there are some objects that serve as input, target, and intermediate steps. The agents can manipulate these entities and exchange information through the intermediate objects, to achieve their shared goal. Objects are scores (piano and orchestral score) and abstract models of orchestration (texture annotations, layer score, and orchestration plan). A segmentation can also be considered as an auxiliary object. Abstract models will be interpreted as intermediate objects in certain processes, but will also constitute the input and/or the target for new tasks. Other constraints are introduced for certain tasks. Some example of constraints that can be introduced for certain tasks are a given melody, a chord sequence, a target audio, or a desired perceptual effect.

This chapter starts by discussing and reinterpreting existing orchestration tasks in MIR with the help of the abstract objects introduced in the previous chapter (Section 4.1). At the same time new tasks will be introduced. Then we will focus on creation (Section 4.2) of orchestration with the help of computers, and we will introduce possible processes and interactions.

4.1. Reclassifying Orchestration Tasks in MIR

The art of orchestration involves a range of tasks, as summarized in Table 4.1. Our understanding of orchestration can be improved by considering tasks in light of the abstract models introduced in Chapter 3. Tasks can be identified and classified by their input and output. They sometimes receive different names when they are performed by humans or by MIR algorithms.
In the table, we mark the tasks modeled in this thesis with a black star (★). In particular, all generative tasks addressed by SymphonyNet [158] can be approached with our SymphonyNet-based transformer model (Chapter 8, Tasks U0, U1, and C1), we have implemented a model to generate orchestration plans for the *Angeles* project (Chapter 7, Task P3), and we have addressed the detection of orchestral blends (Chapter 6, Task A1b, which can be thought of as a kind of texture analysis).
We mark tasks that we have addressed but for which we believe our model is not mature enough with an half empty black star (⯪). These are constrained and unconstrained layer score orchestration tasks (Chapter 8, Tasks T6 and T7), the automated creation of layer scores from orchestral score for building our corpus (Appendix B, Task T4), and orchestration from piano (Section 4.2, Task T1, obtained through combinations of algorithmic and manual Task A3, T3, P3, and T7).
We identify those tasks that appear to be approachable with little modifications of our models but have not been addressed with a light bulb (💡). These are the constrained and unconstrained generation of layer scores (Tasks U3, C3, and C4), the melody and harmony constrained symphony generation (Task C2) that could be achieve by changing the sampling scheme of the existing models, instrument classification (Task P5), and unconstrained reorchestration (Task S1).
Finally, tasks that have been addressed by hand in this thesis are marked with a paper-and-pen symbol (✍). We have constructed layer scores from piano, we have performed taxture-constrained orchestration, and we have done segmentation for the *Angeles* project (Chapter 7, Tasks T3, T7 and A3). We have annotated texture in scores

(Chapter 5, Task A1a) and refined by hand some automatically constructed layer scores for our corpus (Appendix B, Task T4).

Task	**Input/Constraints**	**Output**	
Creation of New Music Material (Unconstrained) (U)			
U0. Multi-Track Music Generation *Polyphonic composition*	–	Multi-Track Score	[79][111][25] [109][148] (surveys) ★
U1. Unconstrained Symphony Generation *Symphonic composition*	–	Orchestral Score	[158] ★
U2. Piano Score Generation *Piano composition*	–	Piano Score	[79][111][25] [109][148] (surveys)
U3. Layer Score Generation *Abstract composition*	–	Layer Score	💡
Constrained Creation of New Music Material (C)			
C1. Harmony Constrained Symphony Generation *Harmony realization for orchestra*	Chord Sequence	Orchestral Score	[158][95] ★
C2. Melody and Harmony Constrained Symphony Generation *Melody and harmony orchestral arrangement*	Chord Sequence + Melody	Orchestral Score	[95][283] 💡
C3. Harmony Constrained Layer Score Generation *Abstract harmony realization*	Chord Sequence	Layer Score	💡
C4. Melody and Harmony Constrained Layer Score Generation *Abstract melody and harmony arrangement*	Chord Sequence + Melody	Layer Score	💡
C5. Audio-targeted Orchestration *Reproduce a target sound by layering orchestral instruments*	Target Audio	Orchestral Score	[214][33][229] [32][75][169] [2][30][29][36]
C6. Perception Controlled Orchestration *Produce a desired perceptual effect through orchestration*	Chord Sequence + Target Perception	Orchestral Score	[187]
Music Inpainting (Completion Tasks) (I)			
I1. Orchestral Inpainting *Generate missing parts of an orchestral score*	Orchestral Score (with gap)	Orchestral Score	[95]
I2. Layer Score Inpainting *Generate missing parts of a layer score*	Layer Score (with gap)	Layer Score	[95]
"Traditional" Orchestration Transformations (T)			
T1. Orchestration (from Piano)	Piano Score	Orchestral Score	[51](live) [198][179] [284] ☆+✎

T2. Piano Reduction	Orchestral Score	Piano Score	[120][254][199] [115][284]
Transformation Sub-tasks with a Layer Score (T)			
T3. Layer Reduction (from Piano) *Construction of a layer score (from piano)*	Piano Score	Layer Score	✎
T4. Layer Reduction (from Orchestral Score) *Construction of a layer score (from orch. score)*	Orchestral Score (+ Texture Annotations)	Layer Score	✎ ⯪
T5. Piano Arrangement (from Layer Score)	Layer Score	Piano Score	
T6. Layer Score Orchestration	Layer Score	Orchestral Score	[95] ⯪
T7. Texture-constrained Orchestration (from Layer Score) *Orchestration of a layer score following a plan*	Layer Score + Orchestration Plan	Orchestral Score	✎ ⯪
Planning Tasks (P)			
P1. Orchestration Plan Generation *Structure and timbre composition*	–	Orchestration Plan	
P2. Orchestration Plan Completion *Timbre transition and structure completion*	Orchestration Plan (with gap)	Orchestration Plan	
P3. Orchestration Texture Planning *Assign layers to instruments (one-to-many)*	Layer Score	Orchestration Plan	★
P4. Orchestration Planning (without Layer Score) *Assign layers to instruments (one-to-many)*	Piano Score + Texture Annotations	Orchestration Plan	
P5. Instrumentation (Instrument Classification) *Assign notes to instrument*	Multitrack Score (w/o instrument labels)	Instrument Labels	[158][67][105] [133] 💡
Analysis Tasks (A)			
A1a. Orchestral Texture Analysis	Orchestral Score	Texture Annotations	[97][42] ✎
A1b. Orchestral Blends Detection *Perception of orchestral blends*	Orchestral Score	Orchestral Blends Positions	[11] ★
A2. Piano Texture Analysis	Piano Score	Texture Annotations	[49][47][48]
A3. Segmentation	Piano Score or Orchestral Score	Segmentation	✎
Self-transformations (S)			
S1. Unconstrained Reorchestration *Propose an alternative orchestration*	Orchestral Score	Orchestral Score	[158][51](live) [284][206] [150] 💡
S2. Texture-constrained Reorchestration *Reorchestration following a plan*	Orchestral Score + Texture Annotations + Orchestration Plan	Orchestral Score	
S3. Re-arrangement for Piano *Simplification/Variations/Rearrangement*	Piano Score	Piano Score	

S4.	Layer Score Enrichment *Composition of new parts*	Layer Score	Layer Score

Table 4.1.: MIR tasks and sub-tasks related to orchestration and orchestration analysis. The algorithmic MIR task name is printed with normal style fonts and the corresponding traditional human task (or a description of the task) is reported in small italics. Tasks are classified by their input and output. In the last column we cite papers that have addressed or discussed the task. We also presented and implemented some tasks in [149], [165], and in this manuscript. Stars represent tasks modeled (★) or addessed (✫) in this thesis. Tasks that appear to be approachable with little modifications of our models are identified by a light bulb (💡), and tasks addressed by hand are marked with a paper-and-pen symbol (✍).

Creation of New Material and Inpainting (Tasks U, C, I, P) While some people may wish to generate complete symphonic orchestral music (Task U1 on Table 4.1), a more fruitful co-creation may involve the generation starting from existing material, such as chord progressions, melodies, or incomplete scores (Tasks C1, C2, and I1). Similar tasks exist for the *piano score* (see for example [102]), and can be introduced for the *layer score* (Tasks C3, C4, and I2). When it is performed by a human alone, it is called composition or arrangement. When it is performed autonomously by an algorithm, it is called constrained/unconstrained generation, automatic arrangement, or inpainting. In co-creative composition, the human and the algorithm collaborate in the creative process. A first strategy for co-creative collaboration consists is splitting the task into sub-tasks that can be performed by the different agents. Another one consists in implementing a feedback loop in which the human provides the input and reviews the output of the algorithmic generation. Alternatively a process in which both actors are involved with different roles could be designed.
Some particular constrained generation tasks are those that involve audio and specific target perceptual effect as control input. Indeed, late-romantic or modern styles have tried to reproduce timbral effects or actual sounds (bells in Wagner, massive texture in Debussy, birds in Messiaen), as modeled by the target-based orchestration tasks (Tasks C5 and C6). In this context, many studies have been proposed on "Audio-targeted Orchestration"[214, 33, 229, 32, 75, 169, 2, 30, 29, 36, 37]. Miranda et al. [187] have instead proposed an interactive Computer-Aided Orchestration system that generates orchestrations to match verbal timbre descriptors.
For the remaining objects, unconstrained generation of texture annotations disconnected from a score does not make a lot of sense, but the generation of an orchestration plan could be part of a composition process, in which planning timbres and sounds with respect to structure comes before harmony and melody (Tasks P1 and P2). In the table, we have placed them in the planning section. For example, it might be the case for ambient music, more based on the evolution of sonic material, or for modern and contemporary western classical music in which timbre is the core (see for

instance Schoenberg's Klangfarbenmelodie [239] and Anton Webern's *Concerto for nine instruments*, Op. 24).

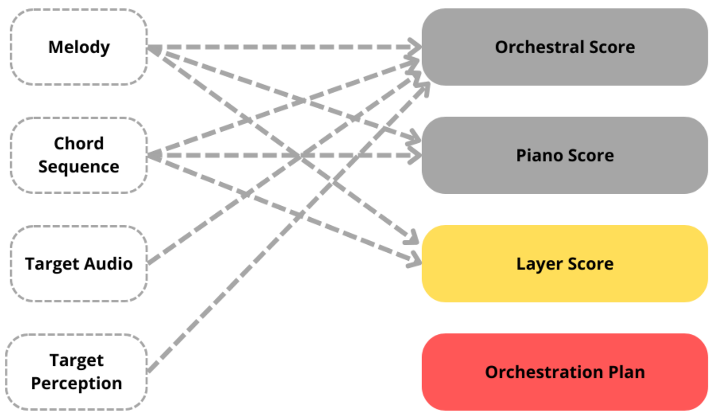


Figure 4.2.: Creation of new material: scores and plans can be generated freely or with constraints. Examples of constraints are a given melody or chord sequence, a target audio, or a desired perceptual effect. Table 4.1 reports the most important examples, but all combinations of constraints shown by the arrows in the figure could in principle be explored.

Transformation Tasks (Tasks T, P, S) We put in this category all tasks which produce one of the defined objects, having as input one or more of them (Figure 4.3). Piano reduction and orchestration from piano are symmetrical tasks in traditional practice (Tasks T1 and T2). Orchestration from piano usually requires "de-pianotizing" the piano music, including voice separation [166] or texture analysis [49], while Piano reduction requires "de-orchestrating" the orchestral music. In both cases, the tasks can be rethought with layer scores and orchestration plans as intermediate objects. One group of tasks involves the construction of a layer score from a piano or orchestral score (Tasks T3 and T4), or the opposite operation, with or without an orchestration plan (Tasks T5 - T7). Another group involves planning the orchestration (Tasks P3 - P5). Other tasks involve regenerating orchestral content over an existing piece, either partially (inpainting, Task I1) or completely (reorchestration), with or without constraints (Tasks S1 and S2). Other "self-transformations" (the transformation of an object into an object of the same kind) are those for the piano score, the layer score, and the orchestration plan (Tasks S3 and S4).

Orchestration Analysis Tasks (Tasks A) The last group of tasks contain those related to the analysis of orchestration (Figure 4.4), involving the identification of certain techniques in orchestral scores. The list that we provide is not exhaustive, but touches the main tasks explored in this thesis. One example is the analysis of orchestral texture (and, analogously, piano texture), producing annotations of layers as detailed in Section 3.1 (Task A1a and A2). We include piano analysis tasks here, as they can be useful

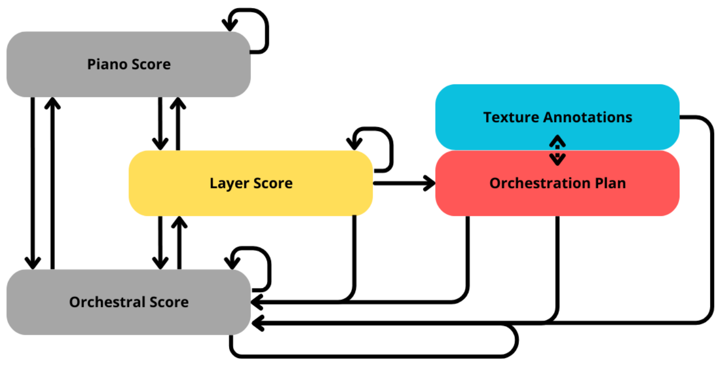


Figure 4.3.: Transformation tasks: transformations allow to pass from one or more objects to another one. Those tasks are divided into transformation tasks and sub-tasks (T), planning tasks (P), and self-transformations (S) that transform one object into an object of the same kind (Table 4.1).

steps in an orchestration process. The identification of segments and formal sections in piano or orchestral scores is another of these analysis tasks (Task A3). Chapter 6 will detail more the orchestral blend modeling and the related task of detecting possible perceived blends in scores (Task A1b).

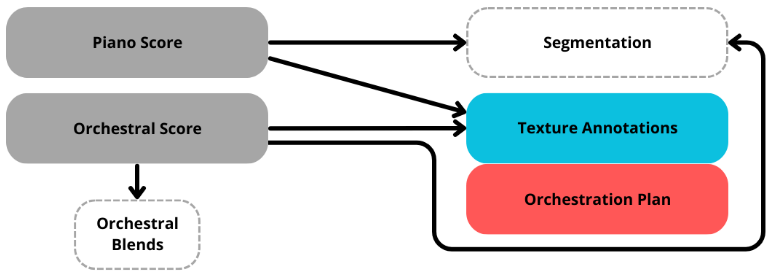


Figure 4.4.: Analysis tasks have the goal of identifying certain objects and techniques (like layers, segments, and orchestral blends) in scores. This is a non-exhaustive representation of some possibilities.

4.2. The Case of Computer-Assisted Orchestration (Task T1)

In this section we present a process for computer-assisted orchestration that combines ideas from music theory and computer science to facilitate the collaboration between artists and computational algorithms to orchestrate existing music pieces. The objective is to perform Task T1, going from a piano score to an orchestral score, in a co-creative and interactive way, allowing both humans and algorithm to take part in high-level and detailed decision-making. In order to achieve that, the task is split into sub-tasks, (Tasks A3/T3, P3, and T7) accommodating both human and algorithmic contributions in distinct roles. The different sub-tasks can be performed sequentially and independently by human and algorithmic agents, but it is more interesting to design each step allowing the possibility of interaction.

4.2.1. A Process in Three Steps

This structured approach unfolds across three pivotal steps (Figure 4.5) aimed at the creation of different intermediate objects and finally of the orchestral score. In this section we present the three steps, and how they can be separated into further sub-tasks, without assigning tasks to agents. The exact processes can vary a lot, based on which agent is taking which task, on the level of musical training of the humans, on the nature of the computational models employed, and on the ways of interaction between actors and objects.

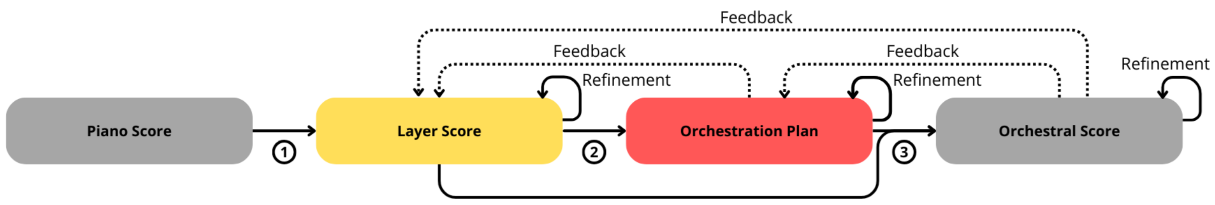


Figure 4.5.: Co-creative orchestration process in three steps: (1) constructing a layer score from the piano score, (2) planning the instrumentation of each layer, and (3) combining the layer score and orchestration plan to write an orchestral score. Iterative refinement is allowed for the layer score, the orchestration plan, and the final orchestral score, meaning that the agents can update the objects until a satisfying version is obtained. Feedback can also be implemented (dotted lines), allowing agents to modify previously taken decisions in a later phase of the process.

(1) Construction of the Layer Score Our proposed creative process starts from an analytical perspective, with an in-depth deconstruction of the composition to orchestrate. It is a is a *creative analysis* [7] task in which the analysis is a catalyst for the rest of the orchestration process. Particularly pertinent in this phase is the identification of the different voices and of the section boundaries between the evolving textures in the piece. The result of this phase is the layer score, the abstract score which contains all the separated voices (layers) present in the original piece, analyzed for their role in the composition (Task T3). The agents are also free to creatively complete the layer score by modifying some of the layers or by adding new ones while carefully respecting the style of the author. There are some analytic sub-tasks that are performed to obtain a layer score (Figure 4.6), like the segmentation of the piece (Task A3) and the identification of layers in the piano score. Those sub-tasks do not need to be completed explicitly (especially when they are done by an expert human), but a good layer score should contain all the information about the segmentation of the piece and, obviously, about the layers.

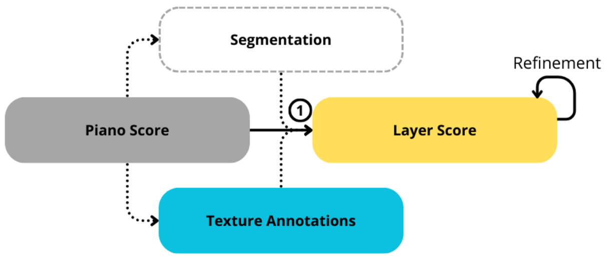
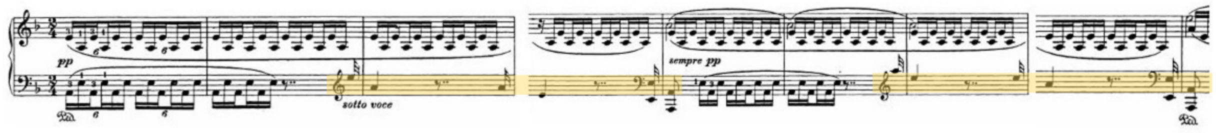


Figure 4.6.: The construction of the layer score involves analyzing the piano score to identify segments and layers, either implicitly or explicitly.

(a) **Piano Reduction.** Arr. by Franz Liszt, ed. Breitkopf & Härtel



(b) **Layer Score.** 3 layers

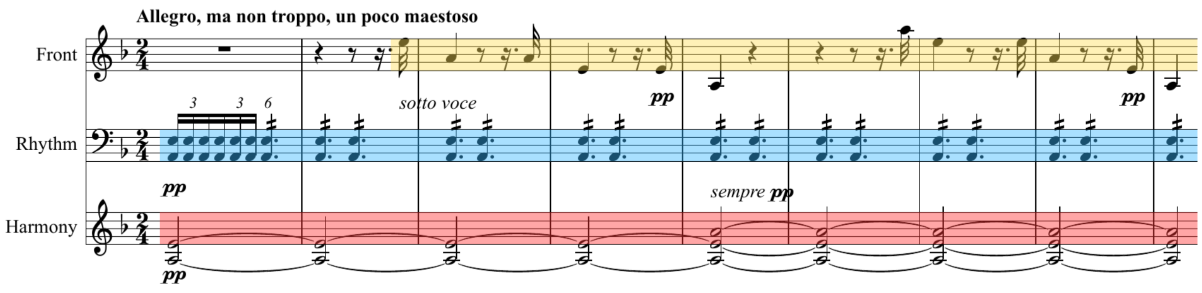


Figure 4.7.: First eight measures from the first movement of Symphony No. 9 by Beethoven, op. 125. Piano reduction and layer score.

As an example, we can imagine that we want to re-write the orchestration of Beethoven Symphony No.9, starting from Liszt's piano reduction. We begin from the first segment, which is the beginning of the symphony, already used as an example in the previous chapter and reproduced here in Figure 4.7. We identify two main elements: the alternated triplets, and the motif of the descending arpeggio. We can write these two voices that we have identified into separated staves in a score, and we can transform the alternated triplets into a figure with repeated notes. This would give us a first layer score, with only the *Front* and *Rhythm* layers. Since we are free to iterate, we can add other layers to this score if we are careful in respecting the style of the author. For example, here we can add some sustained notes that reproduce the same harmony as in the rhythmic layer, which gives the *Harmony* layer.

(2) Orchestration Texture Planning The second phase of the process consists in the creation of an orchestration plan, outlining the instrumentation of each layer across distinct sections of the piece (Figure 4.8, Task P3). A possibility is to let the human select the most suitable orchestration plan among a large collection of plans that are generated by a model for the same piece. Also, we can make plans with dependence on a given set of parameters (like overall loudness and instrumental density), so that they can be adjusted *a posteriori*. A model to generate orchestration plans is presented in Chapter 7.

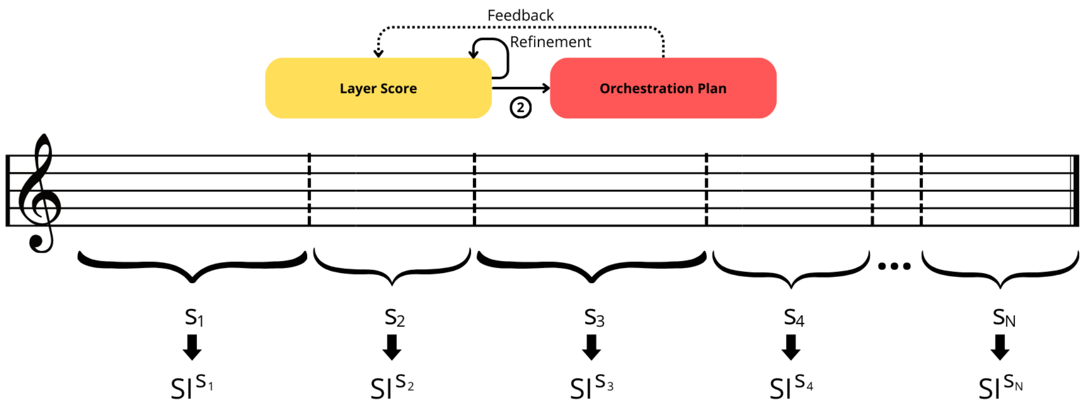


Figure 4.8.: An orchestration plan associates to every segment s_i a segment instrumentation SI^{s_i} containing the layer instrumentations for every layer in the segment.

$$SI^{s_1} = \left\{ \begin{array}{l} (\lambda_{\text{Front}}, \{\text{Vln1}, \text{Vla}, \text{Cb}\}), \\ (\lambda_{\text{Rhythm}}, \{\text{Vln2}, \text{Vc}\}), \\ (\lambda_{\text{Harmony}}, \{\text{Hrn1}, \text{Hrn2}, \text{Cl}\}) \end{array} \right\}$$

Figure 4.9.: Segment instrumentation of segment s_1 (first eight measures) of the first movement of Symphony No.9 by L. van Beethoven, op. 125. Three layers $\lambda_{\text{Front}}, \lambda_{\text{Rhythm}}$, and λ_{Harmony} are mapped to mutually disjoint sets of instruments.

As an example, Figure 4.9 shows the segment instrumentation for the first segment s_1 from Beethoven Symphony No.9.

(3) Texture-constrained Orchestration from the Layer Score following the Plan Finally, the orchestral score is constructed, by writing the notes contained in the layer score for the instruments indicated by the plan, taking care of the instruments ranges and playing techniques (Figure 4.10, Task T7). It is at this point that some of the decisions taken in previous steps might turn out as unreasonable and not feasible. A feedback operation is allowed, and the layer score and orchestration plan can be modified to solve the problems encountered during the creation of the final orchestral score. The final product is further refined to add or adjust some details such as dynamics and expressive indications that might not have been completely considered before.

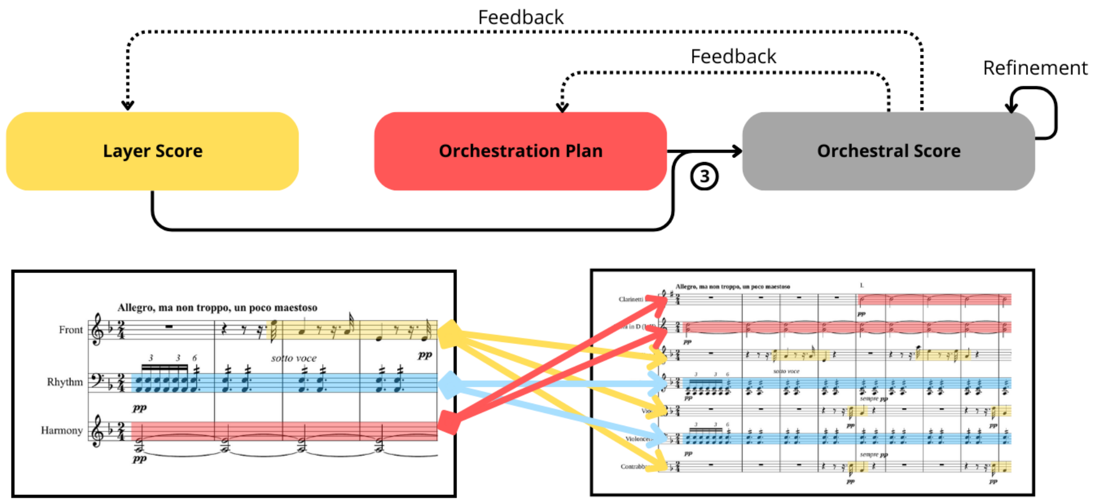


Figure 4.10.: The layers from the layer score are re-written for the instruments of the orchestra, following the prescriptions from the orchestration plan, and adapting them to the instruments' ranges and playing techniques. The lower part of the figure shows the construction of the orchestral score for Beethoven Symphony No. 9.

In the lower part of Figure 4.10 we see, for example, the construction of the orchestral score for the beginning of Beethoven 9th Symphony starting from the layer score and following the instrumentation prescribed by the orchestration plan in Figure 4.9.

4.2.2. Ways of Interaction and Specific Implementations

Thanks to the identification of these three steps with clear inputs and outputs, we can say that, independently from the specific implementations, the process enables the collaboration of a human artist with computational algorithms. The most immediate type of interaction is the use of material produced by humans as input for algorithmic processing (and vice-versa). The human can also interact by selecting and customizing machine outputs, by controlling generation parameters, or by mixing together different outputs. The opposite approach could be taken, with algorithms employed to review the material produced by the human, to check for obvious mistakes, and to propose alternative solutions. Finally, for some tasks, a process which requires the collaboration of the human and the AI as colleagues can be envisioned.

Almost every task in the framework could in principle be performed either by hand or by a computational model, from simple knowledge based AI to more sophisticated Large Language Models (LLMs). Some of these tasks are well studied in literature, such as voice separation in piano scores, some others have been explored and tested for western classical and romantic style orchestration in the context of this thesis (Chapters 7 and 8). Some of them remain to be explored in further studies (Chapter 9).
For the *Angeles* project (see [165] and Chapter 7) we have experimented with a probabilistic Markov model to generate orchestration plans that have then been used to complete a co-creative orchestration that has been played in a concert in Bolivia.
Also, several experiments have been made on token-based LLMs to perform several tasks, among which the constrained generation of orchestral scores. Those experiments have given mixed results and are documented in Chapter 8.

Part II.

Orchestral Music Analysis

5. A Multi-Modal Corpus of First Movements of Symphonies in the Classical Style

In this chapter we introduce a corpus consisting of first movements of Symphonies in the western classical style [230], composed between 1779 and 1824 by the masters of the "first Viennese school" (Haydn, Mozart, and Beethoven). The corpus collects a selection of 24 first movements of symphonies in sonata form [39, 108], together with the abstract models described in Chapter 3. In a first phase, a corpus of texture annotations at a measure level of melodic, rhythmic, harmonic, and mixed layers, has been created and openly released [149]. Those annotations follow the taxonomy presented in Section 3.1 and the syntax detailed in Appendix A. In a second phase, layer scores (Section 3.2), automatically produced from the annotated orchestral scores, have been added to the corpus. In this context, the texture annotations can be interpreted also as orchestration plans (Section 3.3) that prescribe the instrumentation to go from the layer scores to the original orchestral scores.

This chapter is divided into three sections: the first one is detailing the content and the format of the corpus (Section 5.1), including the challenges of visualizing it on the Dezrann platform, the second one is reporting results of statistical analysis on the corpus (Section 5.2), and the last one draws some conclusions (Section 5.3). The corpus is available under open-source licenses through a git repository at `http://www.algomus.fr/data/`, and deposit on the `recherche.data.gouv` data warehouse is underway.

5.1. Corpus Content: Scores, Annotations, and Recordings

This section presents the content of the multi-modal corpus. For each piece, the corpus contains different objects, even though not all formats are always available. A summary of the availability state of the corpus is presented in Table 5.1.
What follows is a presentation of the different material, including scores (Section 5.1.1), texture annotations (Section 5.1.2), sonata form annotations (Section 5.1.3), and layer scores (Section 5.1.4). We then show how we can visualize annotated scores with synchronized public domain recordings through the Dezrann web application (Section 5.1.5).

	symphony (first movement)	orchestral score	texture	layer score	recording	synch	
Mozart 1756 – 1791	Symph. 32, K. 318	-	✓	-	-	-	
	Symph. 33, K. 319	-	✓	-	En. Ch. Orch. (PD)	-	
	Symph. 34, K. 338	-	✓	-	-	-	
	Symph. 35, K. 385, Haffner	-	✓	-	Berl. Phil. (PD)	-	
	Symph. 36, K. 425, Linz	-	✓	-	Bamb. Symph. (PD)	-	
	Symph. 38, K. 504, Prague	MusicXML, mscx (CC0)	✓	-	Bamb. Symph. (PD)	-	
	Symph. 39, K. 543	MusicXML, mscx (PD)	✓	-	Bamb. Symph. (PD)	-	
	Symph. 40, K. 550, Great G minor	MusicXML, mscx (CC0)	✓	auto (MIDI), manual (expos.)	Bamb. Symph. (PD)	-	
	Symph. 41, K. 551, Jupiter	MusicXML, mscx (CC0)	✓	auto (MIDI)	Bamb. Symph. (PD)	-	
Haydn 1732 – 1809	Symph. 99	**kern (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	✓	
	Symph. 100, Military	**kern (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	✓	
	Symph. 101, The Clock	**kern (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	✓	
	Symph. 102	**kern (OTH)	✓	-	R. Phil. Orch. (PD)	✓	
	Symph. 103, Drumroll	**kern (OTH)	✓	-	R. Phil. Orch. (PD)	⚠	
	Symph. 104, London	**kern (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	⚠	
Beethoven 1770 – 1829	Symph. 1, op. 21	MusicXML, mscx (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	-	
	Symph. 2, op. 36	MusicXML, mscx (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	-	
	Symph. 3, op. 55, Eroica	MusicXML, mscx (OTH)	✓	-	R. Phil. Orch. (PD)	-	
	Symph. 4, op. 60	MusicXML, mscx (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	-	
	Symph. 5, op. 67	MusicXML, mscx (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	-	
	Symph. 6, op. 68, Pastoral	MusicXML, mscx (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	-	
	Symph. 7, op. 92	MusicXML, mscx (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	-	
	Symph. 8, op. 93	MusicXML, mscx (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	-	
	Symph. 9, op. 125	MusicXML, mscx (OTH)	✓	auto (MIDI)	R. Phil. Orch. (PD)	-	complete
		19	24	14	22	4	4

Table 5.1.: The corpus contains 24 first movements of Haydn, Mozart, and Beethoven symphonies. The table reports the availability of different material for each symphony, together with licenses. An updated version of this table is available in the corpus repository (`http://www.algomus.fr/data/`). The recordings of Mozart's symphonies are by different orchestras: the English Chamber Orchestra (En. Ch. Orch.) conducted by Colin Davis (1962), the Berliner Philharmoniker (Berl. Phil.) conducted by Karl Böhm (1960), the Bamberger Symphoniker (Bamb. Symph.) conducted by Joseph Keilberth (1956, 1959, 1963). The recordings of Haydn's symphonies come from the album *"The Salomon Symphonies"* (Volume Two) by The Royal Philharmonic Orchestra (R. Phil. Orch.) conducted by Sir Thomas Beecham (1960). The recordings for Beethoven's symphonies come from the album *"The Nine Symphonies Of Beethoven"* by The Royal Philharmonic Orchestra (R. Phil. Orch.) and The Beecham Choral Society conducted by René Leibowitz. Scores and recordings are redistributed with their original license as indicated in the table: Creative Common 0 (CC0), Public Domain (PD), or others (OTH).

5.1.1. Orchestral Scores

Scores for the 24 first movements of symphonies have been gathered from KernScores[1], MuseScore[2] and ClassicalArchives[3]. When allowed by the original license (which is not the case for all of the scores), they have been redistributed along with the corpus in the repository. Score files are in **kern, MuseScore, or MusicXML format. MIDI files converted from the scores, and for which conversion errors have ben corrected, are also made available.

5.1.2. Orchestral Texture Annotations

Texture annotations follow the taxonomy presented in Section 3.1, and describe how the instruments are grouped into layers at a measure-level. Layer roles are also described (either melodic, rhythmic accompaniment, harmonic accompaniment, or mixed) following Benward and Saker [18]. Specialized subcategories for rhythmic support layers like sparse, arpeggio, and scale are available. The relationship between the instruments inside a layer (unison, parallel motion, or homorhythm) is also described. Furthermore, we annotate some "meta-layer" phenomena, like call and response schemes. The annotations are stored both as text files (a description of their syntax can be found in Appendix A) and as Dezrann `json` files [87], that allow to display the annotations over the scores on the Dezrann[4] web platform.

5.1.2.1. Annotation Process

The annotation and reviews were done by hand on scores, both on paper and on the Dezrann web annotation platform [87], that was used to share annotations and to provide feedback to the annotators. Authors had access to recordings, during the process, even though no reference recording had been chosen, and the annotations were made in the spirit of privileging the score information. This is not a perception study, but the texture model describes the cues to certain perceptive phenomena (like blending and streaming) that are present (planned or not) in the score.
The few rules that form the annotation guidelines were designed after preliminary annotations and discussions between the annotators and the reviewers of the annotations, which are also the authors of the publication [149], to resolve some difficult cases. The strategy of doing a measure-level modeling introduces some "rounding" approximation, but at the same time helps avoiding difficulties in defining precise boundaries for each layer. More importantly, such measure-level annotations still imply looking also at the other instruments and at neighboring measures:

- The definition of layers can *be affected by other instruments*. As an example, the presence or absence of a high register melodic layer can affect the classification of

[1]`https://kern.humdrum.org/`
[2]`https://musescore.com/`
[3]`https://www.classicalarchives.com/`
[4]`https://www.dezrann.net/`

a certain pattern in the bass as rhythm or as melody. This, also considering that there might be no melody in some portions of a piece.

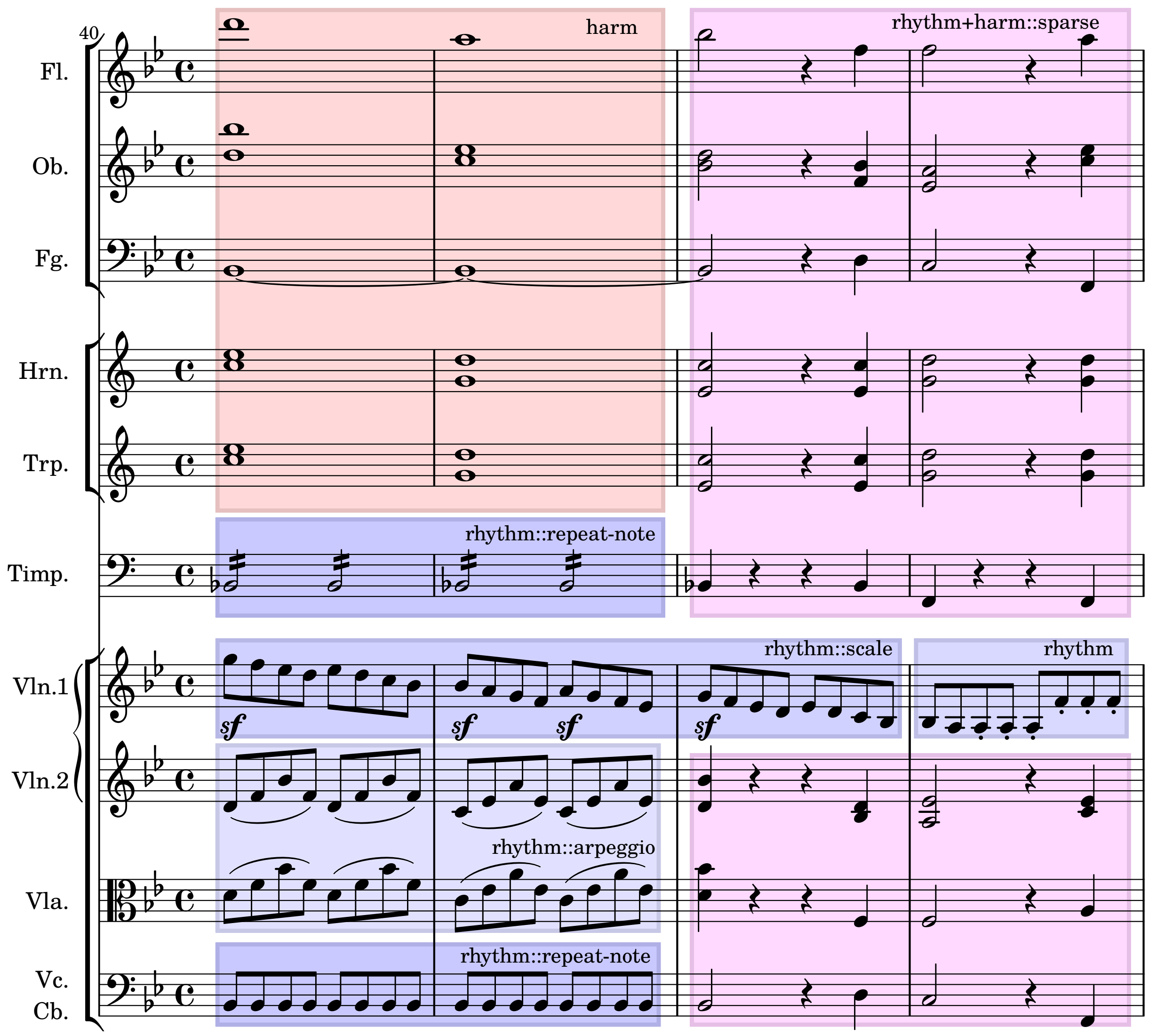
- Annotations *favor longer layers (several measures)*, considering layers as formed by sequential grouping. However, if a melody or a rhythm is logically finishing on the first beat of a certain measure, the corresponding layer is not extended to that measure (as the layers of measures 41-42 on Figure 3.9).

Annotations have been reviewed primarily to enforce coherence of each annotation, to correct obvious mistakes, and to discuss some groupings.

Annotators' Background The annotators and the reviewers of the annotations are my two supervisors (M. Giraud and F. Levé), another member of the Algomus team (D.-V.-T. Le), and myself. We are researchers and PhD students in computer music, with an academic background in computer science and with a music high-school training in flute, violin, piano, music theory, and music analysis. One of us has a prize in orchestration from the Conservatoire de Roubaix (France). We recognize that the fact that the annotators coincide with the corpus curators can lead to some bias, but we argue that the annotation of the corpus was a necessary process to the design and refinement of the annotation language itself. We are vigilant in controlling bias, and we decided to study annotator diversity (see next section) before the reviewing process.

5.1.2.2. Measuring Annotation Diversity

To measure annotation diversity, several annotators independently analyzed the first movement of Mozart Symphony No. 36 (two annotators) and the exposition in the first movement of the Haydn Symphony No. 99 (three annotators), and these annotations have been compared before any review.
The choice of the *grouping* strictly coincide for at least two annotators on 70% of the measures in Haydn 99th and 48% in Mozart 36th. The main causes of disagreement are splitting a layer in multiple ones or different boundaries identification for a layer. We can compare the pairwise combinations of the instruments in a measure, obtaining that at least two annotators agree on 92% of those pairs in Haydn 99th and on 67% in Mozart 36th.
Focusing on the main *roles* (gathering all the rhythm sub-roles), the three (resp. two) annotators strictly agree on the role of 64% of instruments within a measure for Haydn 99th (resp. 76% for Mozart 36th). Here the disagreement is often on the decision of considering a phrase as either a melody or a rhythmic accompaniment, like the first violin part in Figure 5.1. However, there may be ambiguities in identifying layers and roles, both in mixed roles (such as between mel+rhythm and rhythm) and in not clearly affirmed roles (no role annotation for one of the annotators, as in 13% of the corpus). Taking into account these cases, agreement reaches up to 77% for Haydn 99th and 82% for Mozart 36th.



```
[40-42] (rhythm::scale:Vln1)                [42-43] (rhythm+harm::sparse-h:TUTTI)
[40-41] (rhythm::repeat-note:Timp) /        [43-43] (rhythm:Vln1)
(rhythm::arpeggio-u:Vln2.Vla) /
(rhythm::repeat-note-u:Vc.Cb)
```

Figure 5.1.: *Allegro Vivace* in the first movement of Symphony No. 102 by Haydn, measures 40-43, from edition [212] (already shown in Figure 3.9). The role of the first violin is ambiguous: it could be annotated both as melody or rhythmic accompaniment.
`http://algomus.fr/fm/haydn102-40.mp3`

This study validates that our model and syntax are effective for identifying various viewpoints and ambiguous fragments. Additionally, our model demonstrates strong agreement across the majority of layers, facilitating corpus analyses.

5.1.3. Sonata Form Annotations

All 24 first movements of symphonies in the corpus are in sonata form, even if certain movements exhibit unconventional structures, like Mozart 32nd symphony in which sonata form is split among the first and the third movement. We therefore annotate primary and secondary thematic zones (P/S), conclusions (C), development (D), and the recapitulation (P′/S′/C′). These annotations are found in the same text files as the texture annotations, and are used to study the relationship between orchestration and form.

5.1.4. Layer Scores

A preliminary algorithm for the automatic creation of layer scores starting from annotated orchestral scores was developed during this thesis. The objective was to create a collection of layer scores aligned with the orchestral scores in the corpus, facilitating research on orchestration using AI (see the experiments with the Transformer model presented in Chapter 8). The current algorithm (Algorithm 1, reported in Appendix B) cannot be considered mature. We have decided to manually review and curate the output layer score for one excerpt (the exposition from Symphony No. 40 by Mozart) in order to identify the most important weaknesses and flaws. More details on the research in constructing layer scores are reported in Appendix B.
The corpus now contains automatically generated layer scores for 14 of the movements, together with the manually curated layer score for the exposition of Mozart's 40th symphony.

5.1.5. Synchronizing with Recordings and Visualizing the Corpus on Dezrann

Dezrann is an open-source web platform developed by the Algomus team that allows to share music and music analyses in the form of scores, images, audio/video/waveforms, and annotations [14]. It now contains over 10 curated music corpora of diverse origin, including the one presented here. Among other things, Dezrann allows to display different sets of annotations over music scores, to synchronize one or more audio recordings to a score, and to playback the recording while following the annotated score. All this is performed seamlessly in a web browser, without the need of any plugin or external software.
Regarding this corpus in particular[5], we display annotated orchestral scores, synchronized to public domain recordings. Figure 5.2 shows a screen capture of Dezrann

[5]`https://dezrann.net/explore/classical-symphonies`

displaying the annotated Symphony No. 101 *The Clock* by Haydn. Displaying such large symphonic scores with up to 15 staves was a big challenge for the development team.

The recordings have been gathered from Internet Archive[6] and IMSLP[7]. Regarding Mozart's symphonies, they come from different orchestras: the English Chamber Orchestra (En. Ch. Orch.) conducted by Colin Davis (1962), the Berliner Philharmoniker (Berl. Phil.) conducted by Karl Böhm (1960), the Bamberger Symphoniker (Bamb. Symph.) conducted by Joseph Keilberth (1956, 1959, 1963). The recordings of Haydn's symphonies are taken from the album *"The Salomon Symphonies"* (Volume Two) by The Royal Philharmonic Orchestra (R. Phil. Orch.) conducted by Sir Thomas Beecham (1960). The recordings for Beethoven's symphonies are taken from the album *"The Nine Symphonies Of Beethoven"* by The Royal Philharmonic Orchestra (R. Phil. Orch.) and The Beecham Choral Society conducted by René Leibowitz. The synchronization of the score with the recordings has been done (for the symphonies by Haydn, see last column of Table 5.1) through a collective effort of the Algomus team, using the synchronization tool from the platform.

Audio files and synchronization files are made available through the repository.

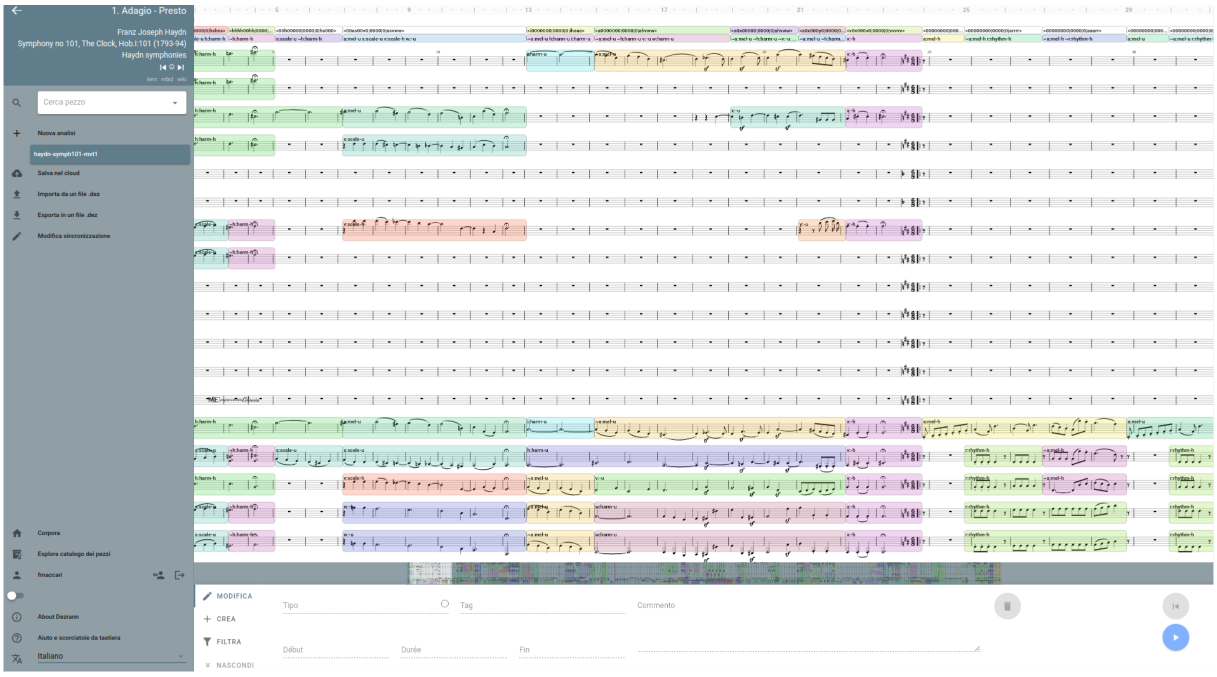


Figure 5.2.: Visualization of Symphony No. 101 *The Clock* by Haydn in Dezrann, measures 2-31. Colored boxes highlight different orchestral layers, while the full texture is described in the boxes above the score.

[6]`https://archive.org/`
[7]`https://imslp.org/`

5.2. Statistical Analysis of the Corpus

We now highlight and discuss properties of such a corpus of classical and early-romantic symphonies concerning layer roles and relations (Section 5.2.1), instrument association (Section 5.2.2), instrument roles (Section 5.2.3), and the link between orchestral texture and form (Section 5.2.4).

5.2.1. Layer Roles and Relations

In Table 5.2 we report the percentages of appearances of layer roles in the corpus. Those numeric values shall be read as the ratio of measures in the corpus where a layer with such role/sub-role is present in the annotations.

Roles and Sub-Roles	**Percentage of Measures in the Corpus**
Melody	68%
Rhythmic accompaniment	76%
Repeated notes	24%
Oscillation or *"Batterie"*	11%
Arpeggio	15%
Scale	8%
Sustained Harmony	66%
Mixed	
Decorative melody	4%
Sparse elements or Sparse chords	30%

Table 5.2.: Percentages of measures in the corpus where a layer with a certain role/sub-role is present. The total does not sum up to 100% since multiple layers can be present in the same measure.

Table 5.3 details the percentages of layers which have an inside-layer relation of homorythm, parallel motion, or unison/octave doubling. Regarding meta-layers, we recorded an average of 15.3 "Call-and-response" (CR) schemes per movement.

Relations	**Percentage of Layers in the Corpus**
Homorhythm	24%
Parallel Motion	9%
Octave or Unison Doubling	29%

Table 5.3.: Percentages of inside-layer relations on the total number of layers.

5.2.2. Instrument Association

In the corpus we observe that instruments of the same family (woodwinds, brass, strings) are more likely to play in the same layers (Figure 5.3, lower triangular part). Brass is more combined with woodwinds than strings: the string section is indeed rarely perceived as blended with the brass section [223, p. 61]. Timpani are more often associated with brass than with other instruments ([82, p. 46], [223, p. 117]). The pair of instruments that can be found most often together in the same layer are cellos with contrabasses (91% of the measures where both are playing).
The difference between instrument families is very significant in call-and-response (CR) schemes (Figure 5.3, upper triangular part). The alternation involving instruments of different families accounts for more than 70% of the annotated CR (including 48% of alternation between strings and woodwinds), whereas intra-family CR are below 25%.

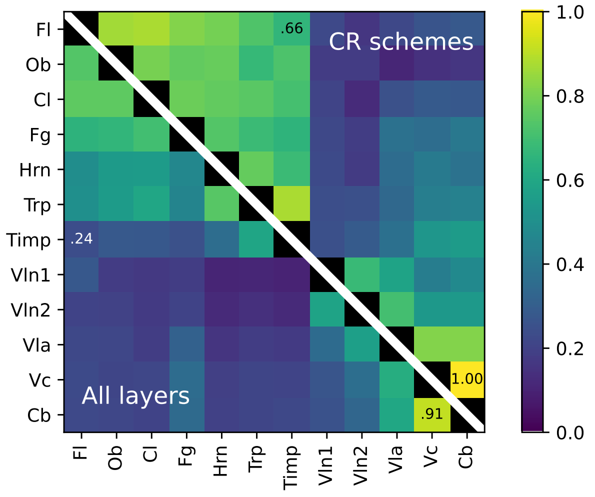


Figure 5.3.: Association of instruments on 19 movements in the corpus. For each pair of instruments, the cells show the ratio of measures when they are in the same layer to the number of measures when are both playing (bottom left) or when they are both playing in a call-and-response (CR) scheme (top right). As an example, flutes (Fl) and timpani (Timp) are in the same layer in 24% of the measures when they are both playing, compared to 66% when one considers only the measures in CR passages when they are both playing.

5.2.3. Instrument Roles

Strings play 84% of the measures in the corpus, woodwinds 57%, and brass 45%. The distribution of roles among instruments confirms common knowledge that the instruments and their families play different roles (Figure 5.4, left).
The melody is mostly played by strings (73% in Mozart (M), 66% in Haydn (H) 53% in Beethoven (B)) – especially, as expected, by first violins (Vln1, 28% (M), 26% (H), 18% (B)). Note that lower instruments also take a significant part in melody: second

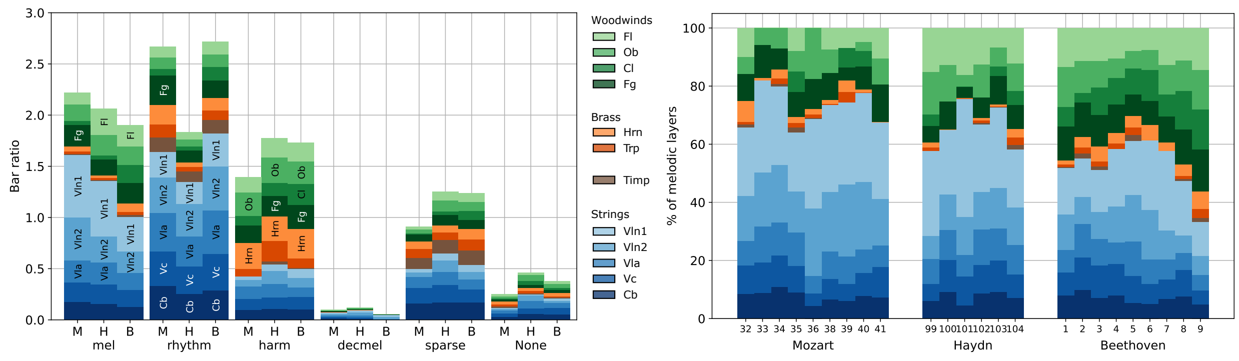


Figure 5.4.: Distribution of instruments by role, composer, and piece. In each family, instruments with highest register are represented with lighter shades. Each area represents the number of measures in which an instrument plays the corresponding role. (Left) All roles are normalized on the total number of measures from each composer (M: Mozart, H: Haydn, B: Beethoven), but each instrumental part is counted on its own. For example, first violins (Vln1) play in a melodic bar 61% of the measures in Mozart movements in the corpus, and the number of melodic measures (considering all instruments) is 222% of the number of measures in Mozart movements. (Right) Focus on the melodic roles on each movement of each composer, normalized on the total number of melodic measures for each movement.

violins (Vln2), viola (Vla), cellos (Vc), and contrabasses (Cb) account for 63% of the mel taken by strings. Most of the time, they play melody either in unison or multiple-part harmony with Vln1, but also, in 9% of the cases, they play the melody without Vln1. On the other hand, the harm role is mostly on woodwinds (46%) and brass (24%). The horn is particularly used in harm (47% of all layers with the horn, compared to an average of 21% for the other instruments) [219, p. 50]. Timpani play rhythm and sparse chords. They also play frequently in tutti passages (34% of layers, compared to 19% for the other instruments, data not shown).
The roles can also be studied among the composers or even the pieces. For example, Figure 5.4 (right) shows the distribution of *melodic* layers among the instruments. In romantic music, winds tend to have a more important place in melody [219, p. 70]: the proportion of wind instruments in melodic layers raises from about 30% for Mozart and Haydn to 47% (± 5%) in Beethoven. This rise is significantly greater than the rise of woodwinds and brass within all layers (42% M, 44% H, 48% B, data not shown).

5.2.4. Texture and Form

Instrumentation and texture are linked to form [260, 250]. Considering movements of the corpus that strictly follow a sonata form, we evaluated the similarities between sonata form sections based on their textural roles composition and their instrumentation. We consider the L_2 distance, normalized between 0 and 1, between two vectors containing the distribution of roles in two sections of the sonata form.

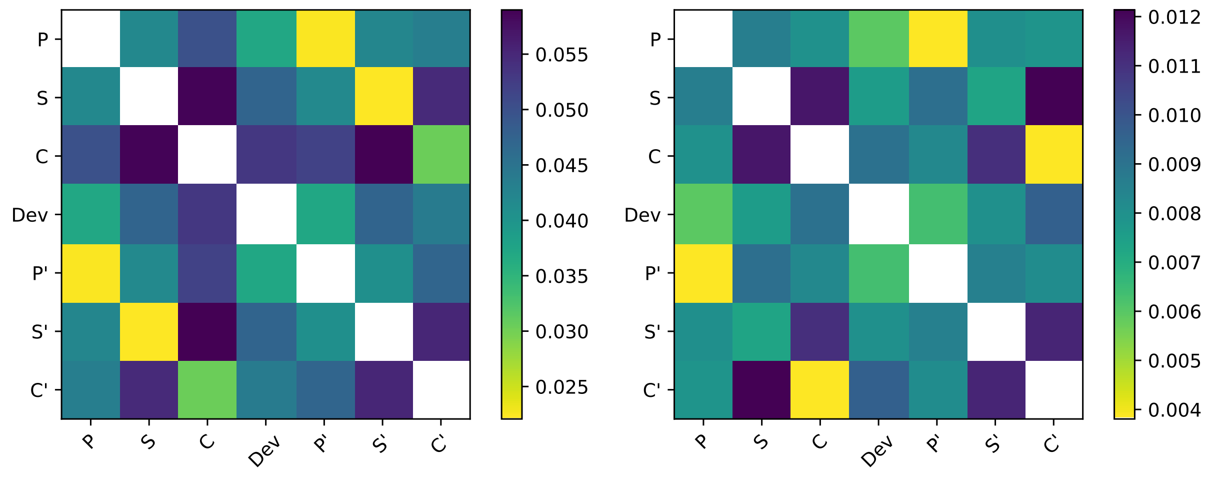


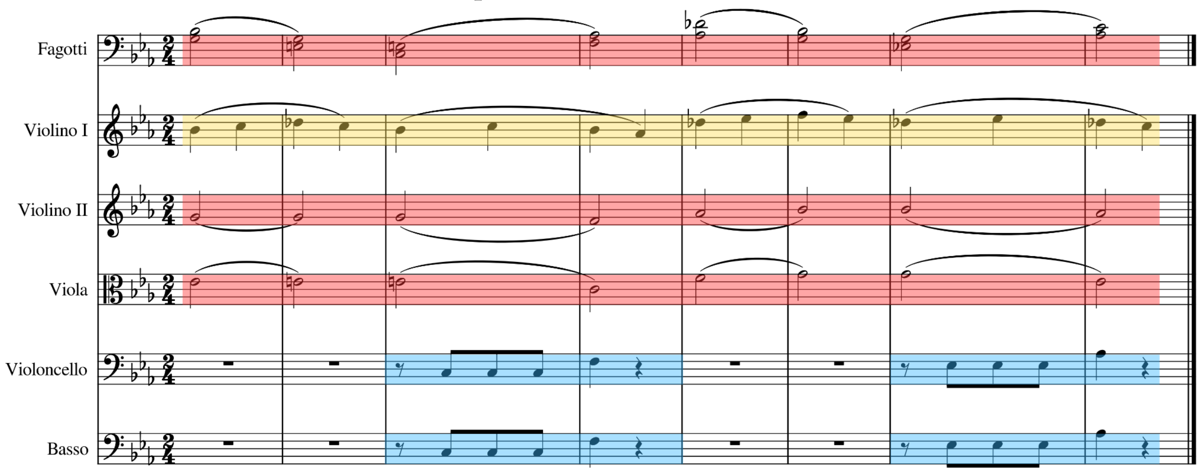
Figure 5.5.: Textural (left) and instrumental (right) disparity of main sonata form sections (P/S: Primary/Secondary thematic zones, C: Conclusion, Dev: development). P′/S′/C′ are in the recapitulation. The cells represents the mean of euclidean distance between vectors representing the occurrence of each role (left) or of each instrument (right) within each section.

The sections having the most similar textures are primary theme zones from the exposition (P) and the recapitulation (P′) (distance of 0.022 ± 0.005), and secondary theme zones from the exposition (S) and the recapitulation (S′) (0.022 ± 0.009). These P/P′ and S/S′ couples are significantly more similar that the other couples of sections (average distance of 0.046 ± 0.003, Figure 5.5, left). Remarkably, this *textural* similarity between the exposition and the recapitulation is more pronounced than the *instrumental* similarities (Figure 5.5, right).
The recapitulation may be indeed a re-instrumentation of the exposition, but still keeping similar texture. As an example, a fragment of the first movement in the Beethoven 5th symphony in the exposition (measures 75-82, Figure 5.6a) had three layers with the melody on the first violins, whereas the related fragment in the recapitulation (measures 323-330, Figure 5.6b) has the same layers but with a call-and-response on the melody between first violins and some woodwinds.

5.3. Conclusions

In this chapter we presented a corpus of 24 first movements of late Haydn, Mozart, and all Beethoven symphonies, with orchestral scores, 7900+ measure-level orchestral texture annotations, sonata form annotations, layer scores, and synchronized public domain recordings. This corpus has served as the testing ground for all ideas related to modeling orchestration (Chapter 3). The concept of orchestral layer has been developed during the annotation of this corpus, with categories and subcategories for layer roles emerging naturally throughout the process. The concept of layer score and orchestration plan have also been tested on the symphonies. A more detailed study on layer scores, on the characteristics they should have, and on their evaluation is still

(a) Exposition (measures 75-82)



(b) Recapitulation (measures 323-330)

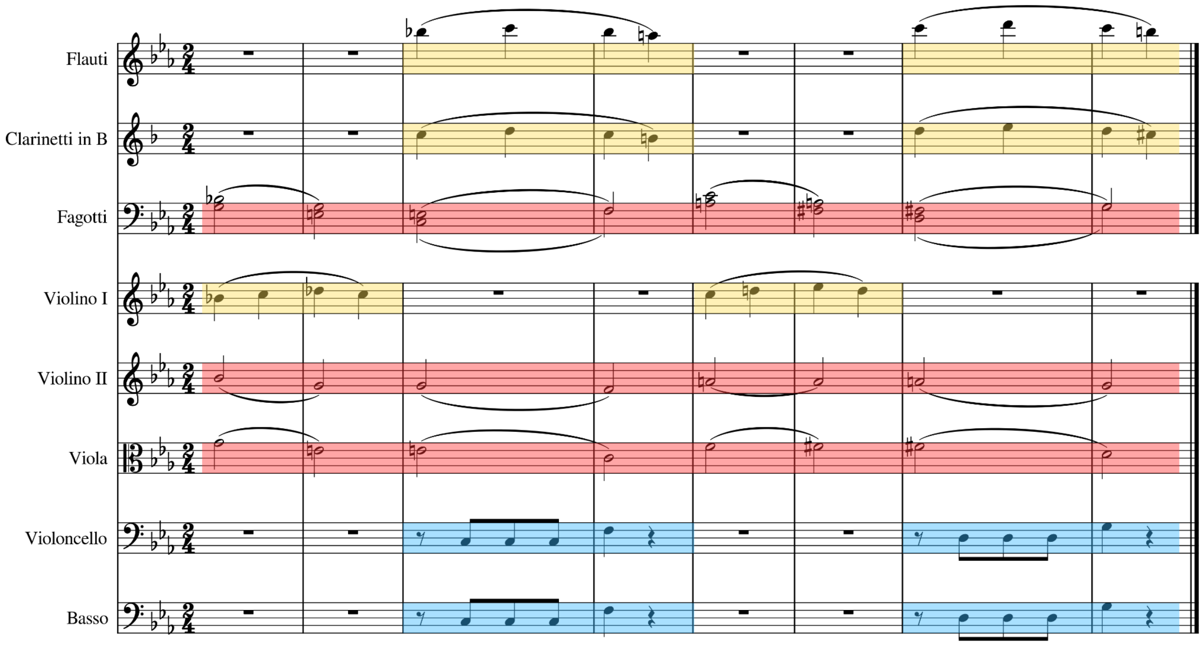


Figure 5.6.: *Allegro con brio* of Symphony No. 5 by Beethoven, second thematic zone in the exposition (a, measures 75-82) and in the recapitulation (b, measures 323-330). In the exposition, the melody (yellow, very light) is played by the first violins. In the recapitulation, it is divided to form a Call-and-response scheme (CR) between the first violins, the flutes and the first clarinet in B♭. The rest of the orchestration remains unchanged: an harmonic layer (red, dark) is played by the bassoons, the second violins, and the violas, while the basses (cellos and contrabass) play the symphony's characteristic rhythmic motive (blue, light).

http://algomus.fr/fm/beethoven5-75.mp3

http://algomus.fr/fm/beethoven5-323.mp3

necessary, and it is left to future work.
Our statistical analyses on the texture annotations confirm the different roles of the instruments and their families in this corpus, in particular, the use of strings and the increasing importance of winds in Beethoven's works, as well as the link between texture and form. These results may seem expected. As symbolic encoding of the full orchestra scores are available for these pieces, this corpus may enable further empirical studies on orchestration, involving scientists with more diverse expertise. Further studies could link measure-level texture to higher-level orchestral effects and style, or focus on other topics on melody, harmony, rhythm, and/or form, but also on organology and acoustics.
The corpus may also enable or improve studies involving Machine Learning, both in texture analysis from scores, in (semi-)automatic transcription from audio, or in music generation tasks empowered with texture (used, for example, as a constraint or a conditioning variable). Some of these ideas are examined in this thesis. In Chapter 8, we delve into several generative orchestration tasks with texture, layer scores, and orchestration plans, using this corpus as training data.

6. Detection of Orchestral Blends from Scores using Machine Learning

As we saw in Section 3.1.5, orchestral blends are related to orchestral layers. They both concern grouping instruments in orchestral music and they can be formalized using the same mathematical objects. In this chapter we investigate the detection of orchestral blends in music scores (Task A1b) using supervised machine learning techniques along with features motivated by perceptive studies. Instrumental blends happen when the simultaneous sounds produced by different instruments are perceived as one entity by a human listener, and they contribute to orchestral layering and the formation of textures. In the Taxonomy of Orchestral Grouping Effects (TOGE), the model of music perception developed by McAdams and collaborators, blends occupy the lowest level of the hierarchical structure, and are the first phenomena to be perceived and processed by the human listener (see Section 2.1.2 and [178]).
For this project we train a Machine Learning classifier on a subset of the OrchARD database (see Section 2.5) to distinguish between blend and non-blend for couples of instruments at a given measure of a piece.
In a previous study, Antoine et al. [11] were able to successfully identify in music scores an average of 81.60% of the instruments participating into blends in a subset of the OrchARD database (using a blend as a unit evaluation). They proposed a system that includes three successive filters, based on onset synchrony, harmonicity, and parallelism in pitch and dynamics, linked to three scoring algorithms. In this project, instead, we adopt a data driven approach, and learn the rules from the corpus using supervised Machine Learning.
The study presented in this chapter is the result of my collaboration with Prof. Stephen McAdams at McGill University, my visit to his Music Perception and Cognition Laboratory (MPCL), and the research project of Olivier Anoufa, student of the Master in Data Science at Centrale Lille that I supervised. A submission is being prepared.
The visit to the MPCL deeply influenced my way of thinking about orchestration: indeed, in this chapter we adopt the timbre and perception approach (see Section 3.1.5) to group instruments and we adopt the term "blend" from the TOGE.

This chapter is organized as follows. We start by introducing the problem in Section 6.1, and we then present the state of the art in orchestral blends detection in Section 6.2. We discuss data in Section 6.3 and we detail our modeling approach as a binary classification problem in Section 6.4. Finally, we conclude by discussing results and perspectives in Section 6.5.

6.1. Introduction and Goals

Orchestral Blends In the context of music orchestration, an *orchestral blend* refers to the process where multiple instruments are playing at the same time, and their individual sounds fuse together to generate a new distinct combined timbre [178, 21]. More precisely, this happens when the listener cannot identify or distinguish the different sources, that is to say, the different instruments that are playing at the same time loose their individuality. A blend can be intentional or not. For a composer or an orchestrator, achieving a good blend involves arranging the instruments in a way that their individual timbres complement each other, creating a unified and cohesive sound that often gives the impression to originate from a single source. Comparable to the blending of colors in the visual arts, where mixing different pigments creates new shades, orchestral blending is employed by composers and orchestrators to generate new timbres, expanding the expressive possibilities of music ensembles.
In their Taxonomy of Orchestral Grouping Effects (TOGE), McAdams et al. [178] place the blending process at the lowest level of a hierarchical organization of auditory phenomena stimulated by orchestral music. As detailed in Section 2.1.2, blends and non-blends are phenomena related to *concurrent grouping*, which concerns the human auditory processes grouping together simultaneous sounds. Blends are of two types: *timbral augmentation*, when the characteristic timbre of one instrument is enriched and amplified by the other blending instruments, and *timbral emergence*, when a completely new timbre is perceived. Finally, this phenomenon can be *sustained* or *punctuated*, depending on its duration, and if it is sustained, it can be *stable* or *transforming*. A stable blend is a blend with a specific group of instruments that does not change during time, whereas in a transforming blend, there is an evolution in time and some instruments can join or leave the blend.
In Section 3.1.5 we have shown that orchestral blends can be interpreted as orchestral layers, where a timbre and perception criterion has been adopted to group the instruments. The entire modeling process adopted in this chapter can therefore be adapted to detect texture-based orchestral layers using the annotated corpus of classical symphonies presented in Chapter 5.

Musical Cues that Affect Instrumental Blending Researchers have shown that several cues can have an effect on the perception of simultaneous musical elements, resulting in the formation of blends and streams. Those include onset synchrony, meaning that simultaneously starting sounds fuse better; harmonicity (or principle of tonal fusion), which means that sounds separated by unisons, octaves, and, to a lesser degree, other consonant intervals blend better; and parallel changes in dynamics and pitch (co-modulation) [11, 234, 31]. The spectral overlap between different sound sources has been identified as a contributing factor [234, 152]. The spatial disposition of the instruments, the characteristics of the performance room, and other factors associated with the musicians' performance can influence the degree of blending [57, 151]. These last three factors are dependent on a specific execution and are, therefore, typically beyond the control of the composer. Conversely, the initial four factors are compositional

choices that can be planned by the composer or orchestrator in the score[1].
These principles are of great importance. However, each of them alone can explain very little. As will be discussed subsequently, not all couplings of synchronous parts result in the component instruments blending together.
Additionnally, Cambouropoulos [31] considers two other principles that pertain to the arrangement of sounds in time: temporal continuity, which refers to the use of continuous or recurring sounds rather than brief or intermittent ones, and pitch proximity, which indicates that successive tones with close pitches better maintain the coherence of the stream.
In the TOGE [178], the concept of blend is distinct from that of stream. Indeed, the formation of blends (*concurrent grouping*) is at the origin of the second class of auditory grouping processes (*sequential grouping*) which determine the formation of streams and textures.

Objectives In this study, we hypothesize that the musical features that give rise to perceptual phenomena are already present and discernible in the score, regardless of whether they were intentionally planned by the composer. Our objective is to examine to which extent this assertion holds true for orchestral blends, to identify the insights that can be extracted about the characteristic attributes of blends, to improve the accuracy of previous investigations, and to focus on the peculiarities of misclassified cases.

6.2. Related Work

A computational model of orchestral blends was proposed by Antoine et al. [11]. In their paper, the authors describe an algorithm that exploits onset synchrony, harmonicity, and parallelism in pitch and dynamics to identify orchestral blends from scores (Figure 6.1). It selects the blending parts with a system that includes three successive filters linked to three scoring algorithms, applied to every measure of the score (see Figure 6.1).
First, parts are filtered by onset synchrony: the onset values for each active instrument in the measure are listed, and the instruments sharing the most onset values (up to a 30 ms window) are grouped together. A synchrony score is calculated with the cardinality of the intersection of the different sets of onset values. The synchronous groups are passed to the harmonicity filter. This second filter checks at every onset if the synchronous notes are in harmonic series, using the lowest pitch as root. If the instruments are not in harmonic series, the algorithm checks if they are part of a tonal chord from a set of templates. If there is no tonal chord including all pitches, the algorithm retains the largest group of notes belonging to a tonal chord. Then an harmonicity score is computed for all instruments that are either playing in harmonic series or forming a

[1]For the sake of completeness, we must report that some composers have tried to have some control also on the disposition of the instruments. For example, Charles Ives divides the orchestra into three instrumental groups that are placed in different spaces for his piece *The Unanswered Question*.

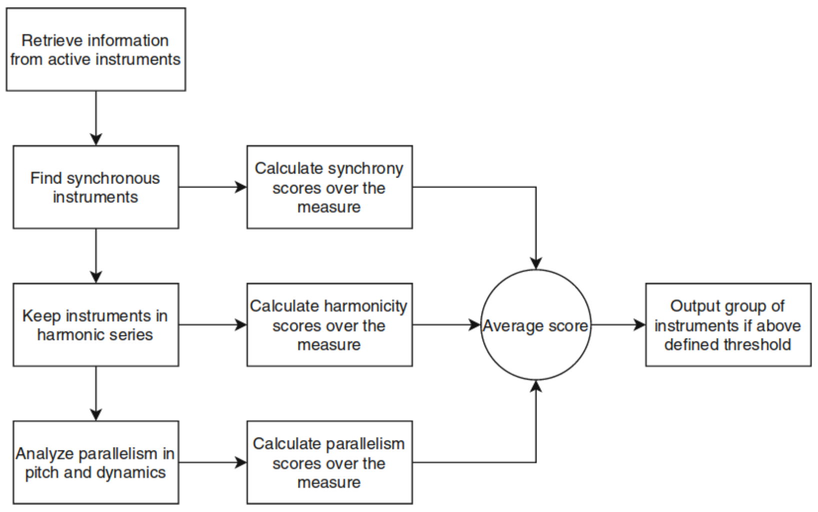

Retrieve information from active instruments
Find synchronous instruments
Calculate synchrony scores over the measure
Keep instruments in harmonic series
Calculate harmonicity scores over the measure
Average score
Output group of instruments if above defined threshold
Analyze parallelism in pitch and dynamics
Calculate parallelism scores over the measure

Figure 6.1.: The algorithm proposed by Antoine et al. [11] to retrieve orchestral blends from scores is based on three successive steps that filter groups based on onset synchrony, harmonicity, and parallelism in pitch and dynamics. Figure reproduced from the original paper.

tonal chord. Finally, the parallelism filter checks if the parts in the groups are parallel in pitch and if they follow the same dynamics curves. The algorithm generates a sequence for each instrument by comparing the pitch of each note to the first note of the measure. It assigns a value of $+1$, -1, or 0 to each note based on whether the pitch is higher, lower, or the same of the first note of the measure. It then compares sequences across instruments, adding 1 for matching values and 0 for differences. The parallelism score is calculated as the proportion of matching elements to the total number of notes. A similar procedure is followed for dynamics.
The final score is computed as the average of the three scores of the three steps, and a group is retained as a potential blend if the score is higher than a given threshold.
The algorithm is evaluated by comparing its output with the ground truth (human expert annotations from the OrchARD database). The evaluation is done in two ways, one regarding every blend as a unit, and the second one measure by measure. The output score corresponds to the number of instruments that are correctly retrieved for each target blend in the ground truth. The best results are obtained by fixing the threshold for the final average score to 60%. By considering a blend as one event, the model successfully retrieved 81.60% of the instruments involved in the effect. Considering each measure as one blend, the model successfully retrieved on average 80.80% of the instruments. This matching score corresponds to a sort of blend-level recall, as it corresponds to the ratio between the number of correctly identified instruments of the blend, true positives (tp), over the number of instrument in the target blend, true positives and false negatives ($\mathrm{tp} + \mathrm{fn}$). Remember that

$$\mathrm{recall} = \frac{\mathrm{tp}}{\mathrm{tp} + \mathrm{fn}}.$$

As a complementary measure, they report the average percentage of extra instruments, which is 13.8% when using the optimal output threshold value of 60%. The way this value is computed is not completely clear.

6.3. Data

The data comes from three sources that we have had the possibility to access thanks to Stephen McAdams. We have access to raw data from the OrchARD database, to the repository containing data, code and results from Antoine et al. [11], and to a database of spectral descriptors.

Expert Annotations from the *orchARD* Database The OrchARD database[1] contains expert annotations of orchestral effects, linked to scans of printed scores and recordings. The information is stored in a SQL database and the access to the online querying interface can be requested by interested researchers. We have been provided with a copy of the raw tables from the database that we have stored in `csv` format. The database contains many orchestral effects, including segregation, stratification, contrasts, and gestures. In our project we only focus on detecting blends and we use this dataset as ground truth to train and evaluate our model. Blends of different kinds are represented differently in the OrchARD database. Timbral emergence blends are represented as a set of instruments at a given location, where the location consists in the beginning and end measure numbers. Those can coincide if the blend happens in one measure only. Timbral augmentation blends are represented in a similar way, but with two sets of instruments, one containing the dominating instruments and the second one containing the embellishing instruments. The two groups share the same effect id. In this project, we choose to only focus on detecting whole blends, avoiding the distinction between timbral augmentation and timbral emergence, and avoiding treating as separate the dominating component and the embellishing component, that we merge in one single group. Except for the evolution type field, there is no difference in case of stable and transforming blend, the annotations list all of the instruments that join the blend between the first and last measure, and do not describe their evolution. In the study by Antoine et al. [11], a handmade post-processing has been done to know the exact components of transforming blends at every measure of a piece. We use this information as target blend.

MusicXML Scores from OrchPlayMusic We worked with a subset of the OrchARD database consisting of 27 extracts of scores, totaling to 1255 measures of music from a diverse range of composers and periods in music history, including Mozart, Berlioz, Debussy, and Vaughan Williams. Antoine et al. [11] used extracts of high quality MusicXML scores from OrchPlayMusic matched to OrchARD annotations to test their algorithm. We have access to the same collection of scores and to their code for the algorithm, allowing us to make precise comparisons with their result. We used a

[1]`https://orchard.actor-project.org/about/`

subset of the scores used by Antoine et al., from which we excluded 5 extracts because of processing issues.

Spectral Descriptors In this project we want to include as features some spectral descriptors that are connected to instrumental timbre. Those have been shown to affect blend perception [152]. For this aim, we use a dataset of spectral descriptors of different instruments that were computed on samples from the Vienna Symphonic Library[2] by Kazasis [138] using the Timbre Toolbox [210].

6.4. Modeling Approach

We frame our problem as a binary classification task, that consists in distinguishing between blend and non-blend for every pair of instruments at every measure of a piece. The choice of the measure as the time window for the analysis is coherent with what decided for this thesis (see Section 3.1). Moreover, it is further motivated by its successful use in the algorithm by Antoine et al. [11], and by the fact that blends are annotated with measure granularity in OrchARD. Reducing the frame of the calculations would result in a significant increase of the amount of data to process, while a longer analysis frame seems inadequate for fast evolving blends. The choice of the binary classification framework seems ideal to treat the problem with supervised Machine Learning.
Our pipeline consists of four phases: encoding the ground truth for binary classification, crafting features, training, and evaluating machine learning models. First we encode with a binary variable the pairs of instruments that are part of the same blend on every measure (Section 6.4.1); then we compute features that can be used as covariates in the classification model, starting from musicXML scores (Section 6.4.2); and finally we train and validate several variants of Machine Learning models for classification (Section 6.4.3) using a leave-one-piece out approach and several evaluation metrics, while holding a part of the dataset completely unseen for testing (Section 6.4.4).

6.4.1. Ground Truth Encoding

We build a ground truth table where each row corresponds to a pair of instruments at a given measure, and which contains the target value for the binary classification tasks (blend vs non-blend).
In order to encode the information from OrchARD in this format we proceed in the following way. We start by considering an orchestral blend β defined as a subset of the cartesian product between measures and ensemble, $\beta \subset \mathcal{M} \times \mathcal{E}$ (a blend has the same formal description of a layer, Definition 3.5). At every measure m each blend is characterized by a set of instruments, which is the m-section of the set β (Definition 3.6). Each measure can contain zero, one, or more blends. Detecting the blends in a given measure then consists in identifying a collection of disjoint sets of instruments. Our

[2] `https://www.vsl.co.at/`

modeling approach is based on a pairwise analysis. For every measure, we consider all possible pairs of instruments in the ensemble $\mathcal{E}$ and we assign to each of them a binary variable that takes value 1 if the two instruments belong to the same blend and 0 otherwise.
Consider for example

$$\beta_{45} = \{m_{11}, m_{12}, m_{13}, m_{14}, m_{15}\} \times \{\mathsf{Vln1}, \mathsf{Vln2}, \mathsf{Cb}\} \subset \mathcal{M} \times \mathcal{E}.$$

The couples that belong to blend β_{45} at any measure between 11 and 15 are

$$\{\mathsf{Vln1}, \mathsf{Vln2}\}, \{\mathsf{Vln1}, \mathsf{Cb}\}, \text{and } \{\mathsf{Vln2}, \mathsf{Cb}\}.$$

Then, assuming that no other blend is present on those measures, we would assign 1 to $(\{\mathsf{Vln1}, \mathsf{Vln2}\}, m_{11})$, and 0 to $(\{\mathsf{Vla}, \mathsf{Vc}\}, m_{11})$. In this way the problem is framed as a binary classification task between blend and non-blend for instrument pairs at every measure of the piece.

6.4.2. Feature Engineering

We compute several features that describe the content of the music score at the given measure and for the two given instruments. Those features are extracted from the musicXML scores using the python library `music21` [54].
The first three features are computed on a single measure for each instrument independently, and describe the distribution of the notes. They are

- note distribution statistics:
  - the number of notes in a measure,
  - the median pitch in the measure, and
  - the pitch range in a measure, which consist in the interval between the lowest and the highest pitch expressed in semitones.

Then we compute features that compare an instrument pair in a given measure. Those are

- note distribution statistics: the absolute difference between the values the first three features take for a considered pair of instruments at every measure,
- the onset synchrony score,
- the harmonicity score,
- the pitch parallelism score,
- the absolute difference between the first and second components of the cosine contours (comparing two parts on a larger time window, between measure m and measure $m + 1$), and

- spectral descriptors (centroid, spread, skewness, kurtosis, and flatness).

As discussed in Section 3.1.5, this model of blends does not have exactly the same focus as the rest of the thesis and as the analysis we did in Chapter 5. Somehow, many of the features that we encode here correspond to our criterion for analyzing layers, in particular onset synchrony and parallelism. However all features related to spectral descriptors are unique for this chapter, as they pertain to the study of timbre perception. The rest of this section presents further explanations on the computation of some of the features.

Note Distribution Statistics They are the first three features listed above and their absolute differences. No further explanation is required. The idea is that similar parts should have similar note distribution statistics.

Onset Synchrony Two parts are said to display onset synchrony if their notes start simultaneously. A stronger version of the synchrony property requires notes to also share the same duration. We implement the feature as the ratio between the number of synchronous notes between the two instruments (#synch) and the maximum number of note between the two instruments in the measure (#notes$_1$ and #notes$_2$):

$$\text{onset synchrony} = \frac{\text{\#synch}}{\max(\text{\#notes}_1, \text{\#notes}_2)}$$

We implement both a strict version of the feature that requires same note durations and a loose version of the feature without this request. We used the strict version in the final model. Figure 6.2 shows an example with perfect synchrony (score of 1). Figure 6.3 shows another example of a transforming blend with non perfect synchrony between the different parts.

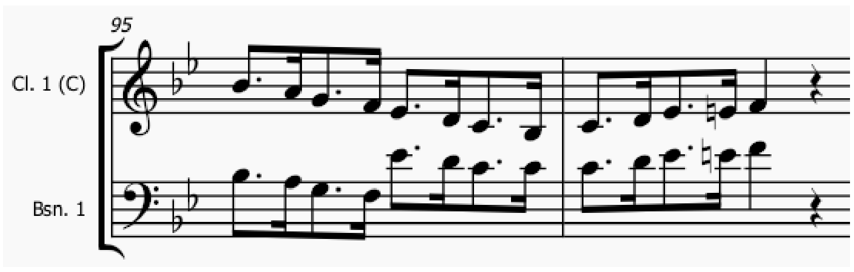


Figure 6.2.: Fourth movement from *Symphonie Fantastique* by Berlioz, measures 95-96. Perfect onset synchrony between Clarinet 1 and Bassoon 1.

Harmonicity Two instrumental parts are said to be in harmony if they play notes that are in a consonant interval. There is no scientific consensus on what are the consonant intervals and on why they are consonant [183]. We adopt what is commonly accepted

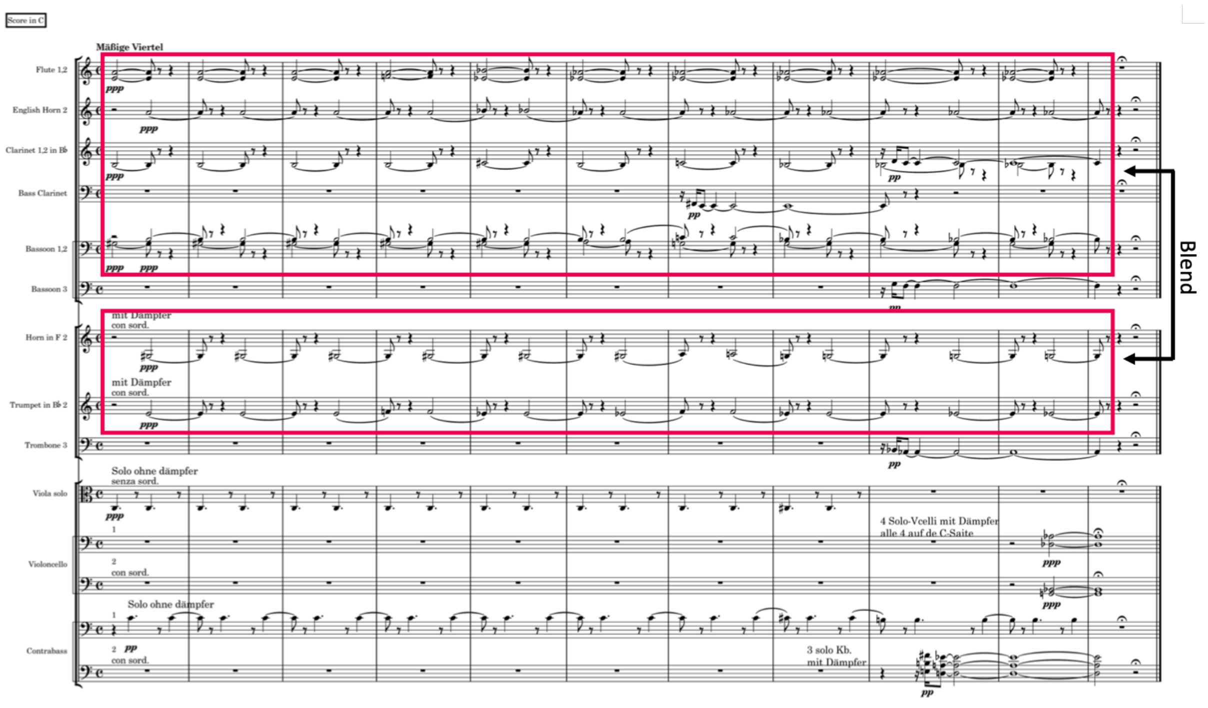


Figure 6.3.: Third movement from *Five Pieces for Orchestra* by Schoenberg, op. 16, measures 1-11. A sustained and transforming timbral emergence blend. The different parts are not all in perfect synchrony.
Figure and audio reproduced from [178].
http://algomus.fr/fm/schoenberg-five-pieces-iii-1.mp3

and thought in music theory classes. We separate perfect consonances (unison, octave, fifth and fourth), imperfect consonances (third and sixth) and dissonances (second, seventh and tritone) [18, 35]. We build the harmonicity score by assigning to every concurrent couple of notes i a score h_i of 1 for perfect consonance, 0.5 for imperfect consonance, and 0 for dissonance. We then average the consonance scores on the number of notes in the studied measure, considering the maximum number of notes between the two instruments in the measure (#notes$_1$ and #notes$_2$):

$$\text{harmonicity} = \frac{\sum_i h_i}{\max(\#\text{notes}_1, \#\text{notes}_2)}$$

In the example displayed in Figure 6.4, the two parts maintain hamonicity during the three measures, alternating between imperfect and perfect consonance.

Pitch Parallelism Two parts are said to be parallel in pitch if the intervals between successive notes are going in the same direction. We implemented the feature that assesses the parallelism between two instruments in the following way. A function examines the note sequences of each part at the same time, comparing successive notes to determine if they are higher, lower, or equal in pitch relative to the previous note.

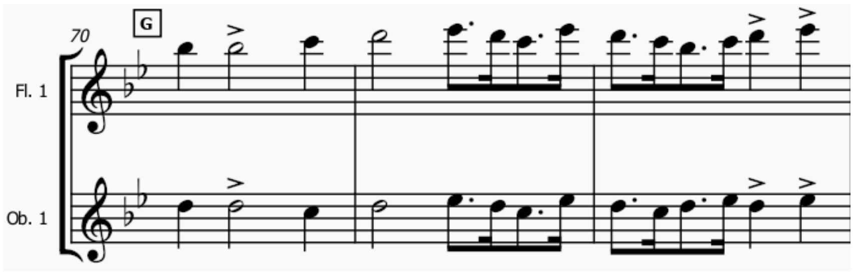


Figure 6.4.: Fourth movement from *Symphonie Fantastique* by Berlioz, measures 70-72. Flute 1 and Oboe 1 maintain hamonicity during the three measures. They start with a D and B♭ which form a minor sixth interval, an imperfect consonance. Then, during the rest of the excerpt, the two parts play the same notes, forming a perfect octave, a perfect consonance. Again they depart from the perfect octave on the second quarter of measure 72 (major and minor sixth), to conclude in perfect octave on the second half of the measure. This produces harmonicity scores of 0.67, 1.00, and 0.83 for the three measures.

Each time the two parts have the same movement, the function increments a counter. The resulting parallelism score is the ratio between this "same movement" counter c and the maximum number of notes between the two instruments in the measure ($\#\text{notes}_1$ and $\#\text{notes}_2$).

$$\text{pitch parallelism} = \frac{c}{\max(\#\text{notes}_1, \#\text{notes}_2)}.$$

Figure 6.5 displays an example of two parallel parts moving down in octave doubling.

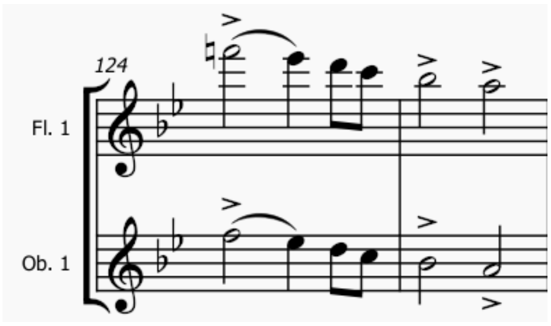


Figure 6.5.: Fourth movement from *Symphonie Fantastique* by Berlioz, measures 124-125. Parallelism in pitch between Flute 1 and Oboe 1. The two parts play a descending scale in octave doubling.

This feature, like synchrony and harmonicity, is a score in the interval $[0, 1]$, corresponding to a proportion of notes that have a parallel motion in pitch.

Cosine Contour Difference We consider an alternative method to evaluate pitch parallelism, or the similarity between the shapes of two instrumental parts. The cosine contour is a compressed representation of melodic contours proposed by Cornelissen et al. [46]. A melodic contour can be thought of as a step function that describes the height in pitch of the notes during time. The cosine contour is computed by taking the discrete cosine transform (dct) of the melodic contour function. The basis functions for the dct are given by $v_k(n) = \alpha_k \cos\left(\frac{\pi(2n+1)k}{2N}\right)$, where $\alpha_0 = \frac{1}{\sqrt{N}}$ and $\alpha_k = \sqrt{\frac{2}{N}}$ are normalizing constants. The coefficients c_k of the discrete cosine transform are obtained by projecting the contour on the basis functions. They provide a condensed abstract description of the shape of an instrumental part without precise information on pitches and rhythm. Figure 6.6 shows how a cosine contour is computed starting from a step function representing the piano roll. The coefficient c_0 describes the average pitch of the part, then low order coefficients describe the general shape of the contour while higher order coefficients describe high frequency variations of the contour.

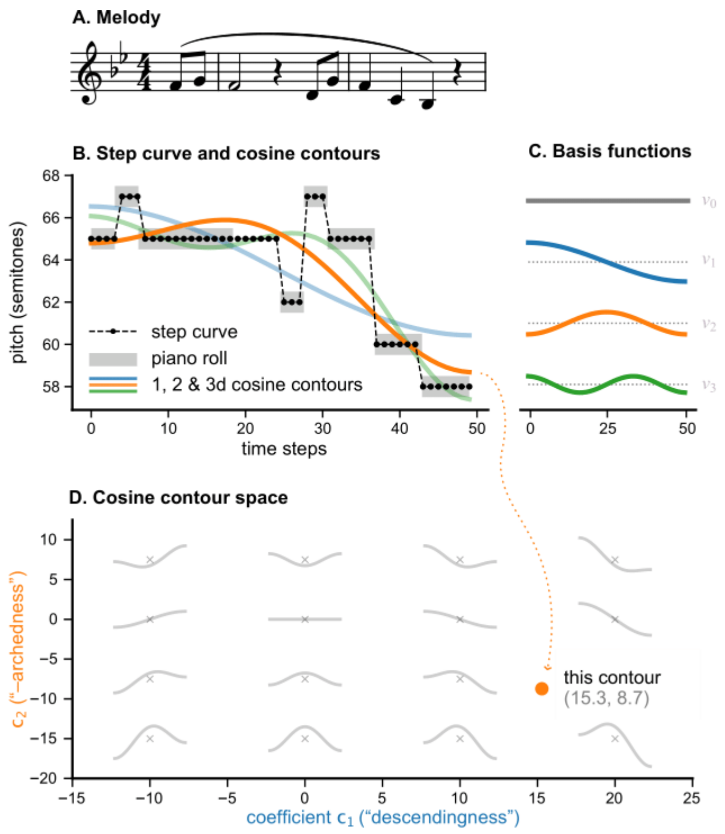


Figure 6.6.: The cosine contour is the cosine transform of the step function representing the melodic contour of a part. Illustration reproduced from [46].

To compute the cosine contour difference between two instruments at measure m, we compute the cosine contour of their two parts considering the notes in measure m and $m+1$. We use more than one measure to give a better overview of the overall movement of the part and for the transform to be less sensitive to local "noisy" movements. We only consider the coefficients c_1 and c_2 of the transform which are called respectively "descendingness" and "archedness" by Cornelissen et al. [46]. In this way every part is placed in a two-dimensional cosine contour space. We exclude c_0, which corresponds to the average pitch, since we already included the median pitch in our features, and we discard also all coefficients c_k with $k > 2$. Two examples of comparison of the cosine contours are displayed in Figure 6.7 (similar shapes) and Figure 6.8 (different shapes).

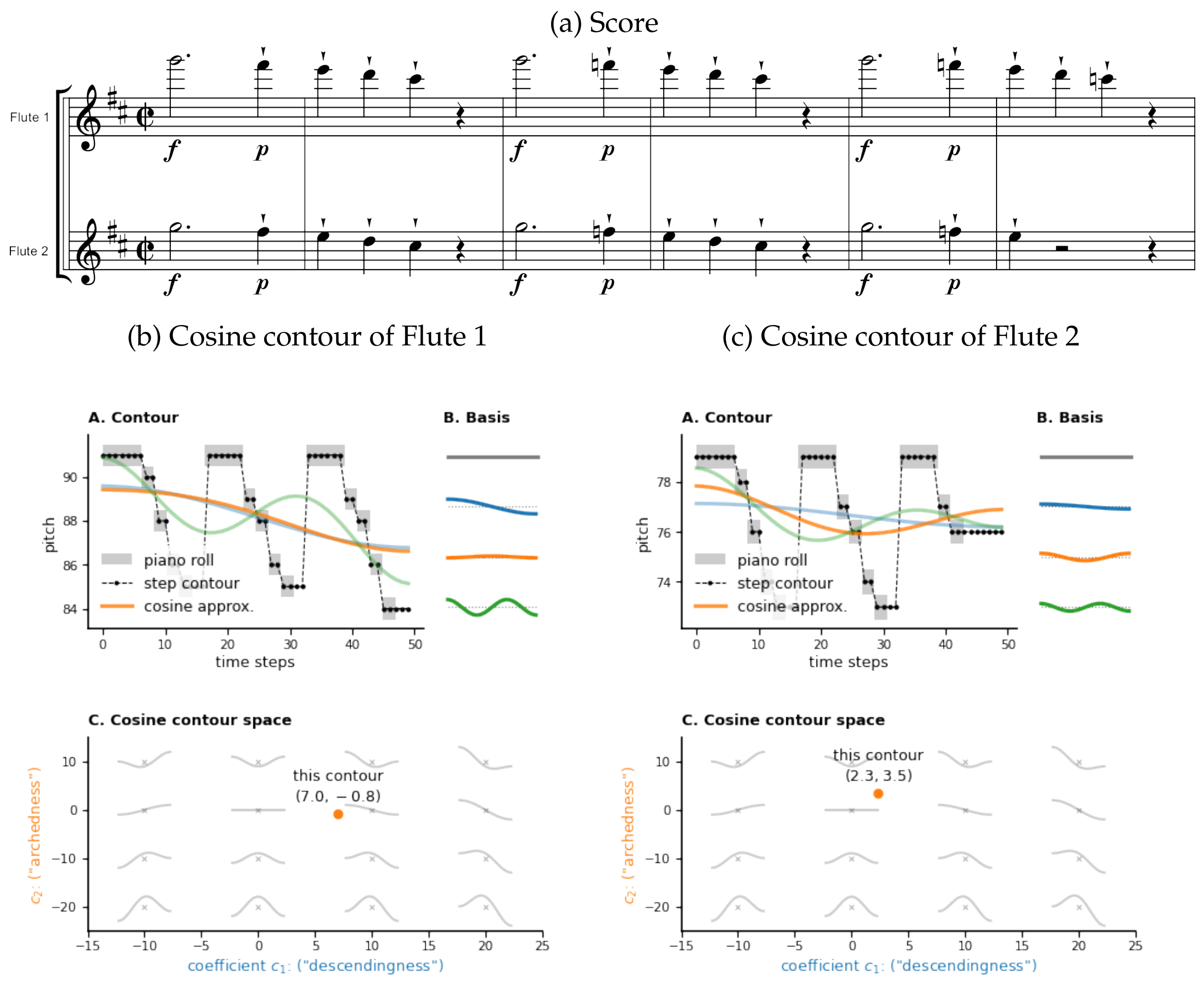


Figure 6.7.: Overture from *Don Giovanni* by Mozart, measures 247-252. Comparison of the cosine contours of Flute 1 and Flute 2. Although not being exactly the same, the two parts have a similar shape, resulting in two close points in the cosine contour space.

We consider the difference between two parts component by component. If c_k^j is the k^{th} component of the cosine contour for instrument j, then

$$\text{cosine contour difference}_1 = \left|c_1^{ins1} - c_1^{ins2}\right|,$$
$$\text{cosine contour difference}_2 = \left|c_2^{ins1} - c_2^{ins1}\right|.$$

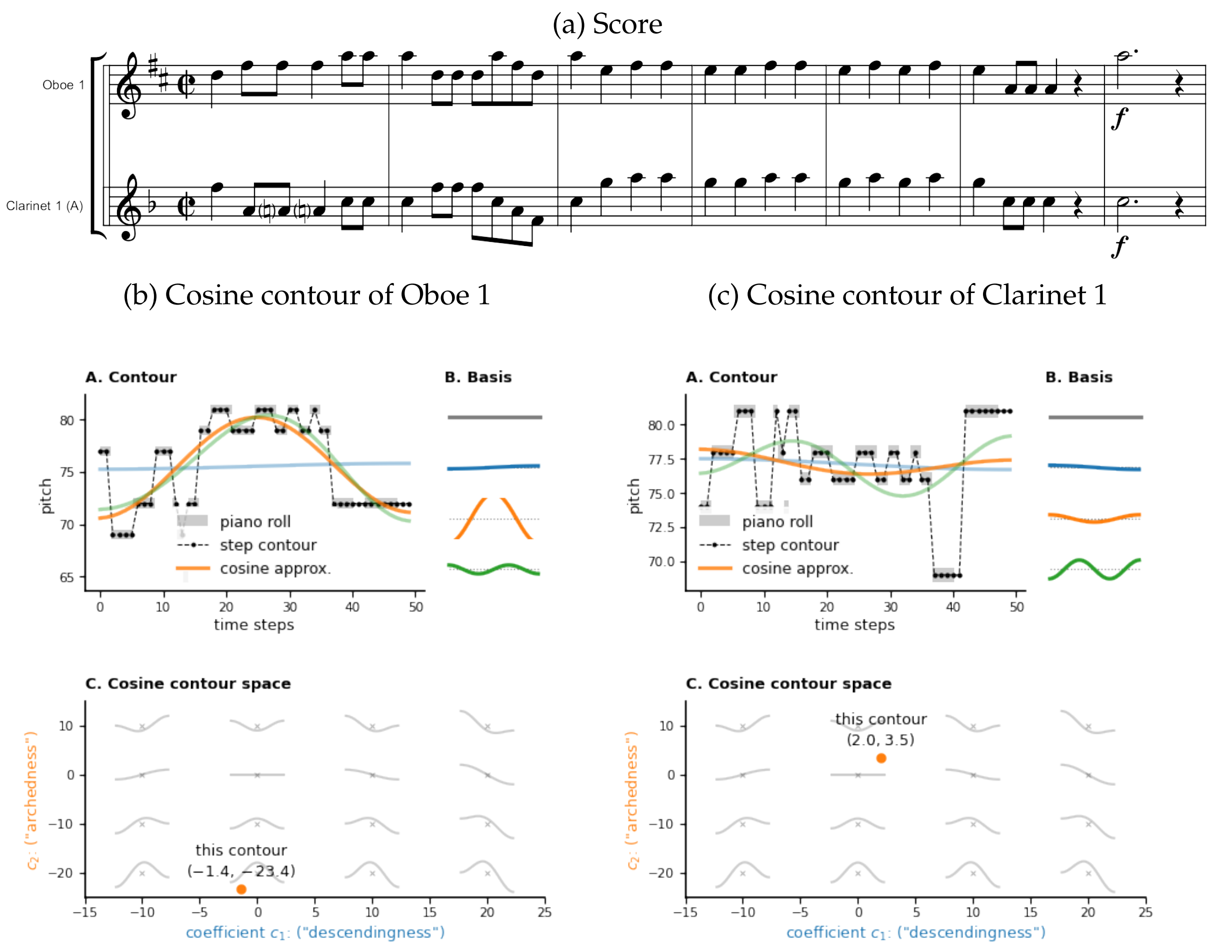


Figure 6.8.: Overture from *Don Giovanni* by Mozart, measures 209-217. Comparison of the cosine contours of Oboe 1 and Clarinet 1. The two parts are perfectly homorhythmic, but they have a different contour shape. The position in the cosine contour space captures the differences in shape between the two parts, more pronounced in the second coordinate "archedness".

These two features can take values in the interval $[0, +\infty]$. We expect that the more two parts are different in shape, the larger the differences in cosine contour coordinates, the less they are likely to blend.

Spectral Descriptors Different timbre characteristics of the instruments can have an effect on their blending qualities. It has been shown that instruments with similar spectral characteristics tend to blend better [234, 152]. We investigate these aspects that have not been much explored in existing computer science models of blends by incorporating a few spectral descriptor features.
The simplest way to take instrumental timbres into account is to feed to the model two categorical variables that simply encode (with a one-hot encoding) the names of the instruments. Through this simple information, a model can learn from data which

instruments tend to blend better together.
We also decided to include the values of some spectral descriptors, that are known for having an effect on instrumental blending. We consider (see Figure 6.9 for a simplified graphical representation)

- the spectral centroid, which is the spectral center of gravity of a sound,
- the spectral spread, which is the standard deviation of the spectrum around its mean value,
- the spectral skewness, which measures the balance and the asymmetry of the spectrum around its mean value with a positive (respectively negative) value indicating more energy in the frequencies below (respectively above) the spectral centroid,
- the spectral kurtosis, which gives a measure of the peakedness or tailedness of the spectrum, this value can change independently from the spectral spread, and
- the spectral flatness, which is a measure of the tonal quality of the sound, obtained by dividing the geometric mean of the spectrum by its arithmetic mean. Maximal spectral flatness (approaching 1) is obtained by white noise, minimal spectral flatness is obtained by pure tones.

The first four descriptors loosely represent the moments of the spectrum. For a precise mathematical formulation of those indicators we refer the reader to the Timbre Toolbox paper [210]. In this work, we want to give a score-based analysis and detection algorithm, so we do not compute these features on recordings of the pieces. Instead, we start from a dataset containing the descriptors presented above for different instruments. This dataset was constructed by Kazasis [138] using the Timbre Toolbox [210] and samples from the Vienna Symphonic Library (VSL)[3]. Each timbre descriptor is characterized by its median value and inter-quartile range throughout the time evolution of the sound. The dataset contains all those descriptors computed on the whole playing range of every instrument, which means that each descriptor has a value for each note pitch and each dynamic level. Figure 6.10 is a plot of the spectral centroid median for the violin which contains the measure computed on all violin samples in the VSL. We disregard the dynamic level and only keep an average of the descriptors per pitch to simplify the computation.
For each measure and each instrument in a piece we compute the median pitch and we retrieve the spectral descriptors for the given instrument at the computed median pitch. We then compute the absolute difference between each spectral descriptor for all the couples of instruments, which leaves us with the absolute differences between the two instruments' spectral centroid median and inter-quartile range (iqr), spectral spread median and iqr, spectral skewness median and iqr, spectral kurtosis median and iqr, spectral flatness median and iqr. The obtained features represent some sort of timbre distances between the two parts on the studied measure.

[3]`https://www.vsl.co.at/`

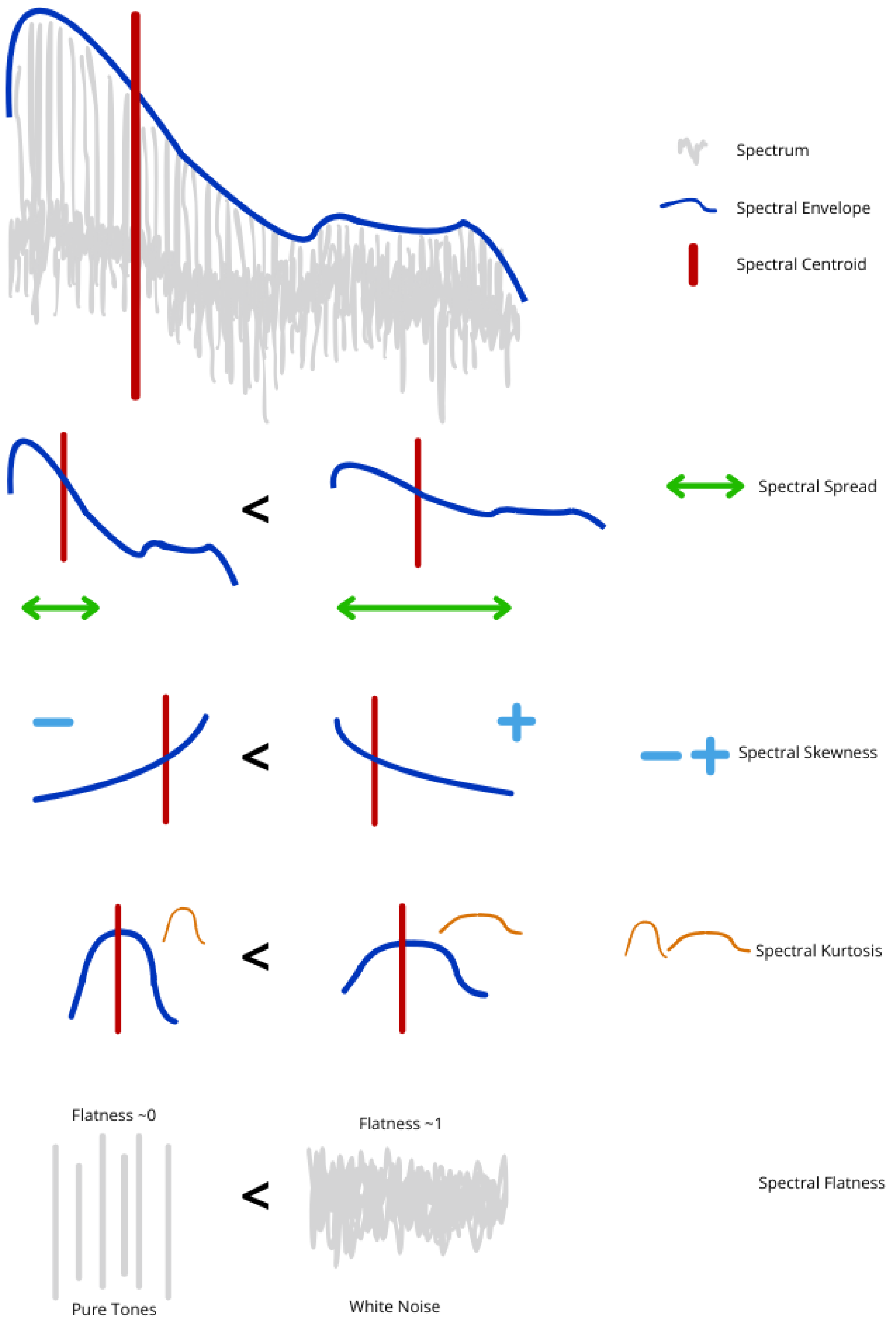


Figure 6.9.: Graphical representation of the spectral timbre descriptors used for this project. See [20] for more practical examples.

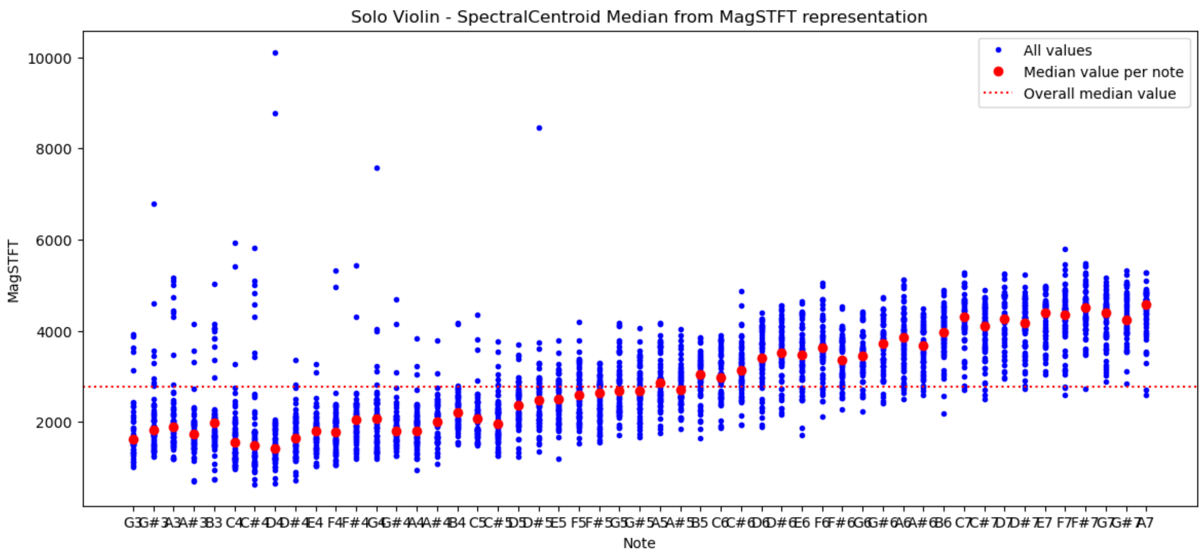


Figure 6.10.: Samples for the spectral centroid median of the violin in the dataset constructed by Kazasis [138]. Every blue point corresponds to a violin sample from the Vienna Symphonic Library. Multiple points correspond to the same pitch, as the different dynamics and playing techniques are collected in this huge library. We highlight the median value per pitch with larger red dots, and we represent the overall median value with an horizontal dotted red line. As expected, the spectral centroid depends on the pitch. We use the median value per pitch (large red dots) in our models.

6.4.3. Training Machine Learning Models

We train and compare several machine learning models on the classification task blend vs non-blend for instrument pairs at a given measure of a piece, using as input the features described in Section 6.4.2, or their manipulations. In our experiments we compare between different models and different combinations of the features. When training the models, we do not consider the rows that correspond to instruments that are silent in the given measure. This obtains the effect of rebalancing the proportion of the target variable. In the inference phase, we always give the prediction of 0 in the case that at least one of the instruments is silent, bypassing the use of the machine learning models. Table 6.1 summarizes all the models that we have trained, indicating their type, the features used, whether standardization of the features to uniform their mean and standard deviation to 0 and 1 has been applied, and types of data augmentation. We used simple statistical models for the classification task: regularized logistic regression, k-nearest neighbor, and decision tree. Those models are lightweight and can be quickly trained and validated on any laptop cpu, allowing to put a cross-validation scheme in place. Three dummy models have been added for comparison: their outputs are always 0, always 1, or a random prediction. All calculation have been performed in python using the `scikit-learn` library. The entire cross-validation scheme for all listed models took about half an hour on a laptop equipped with an Intel Core i7-9750H cpu.

Name	Description	Features used	Scaling
`dummy0`	Always output 0	-	-
`dummy1`	Always output 1	-	-
`dummy_unif`	Output 1 or 0 from uniform distribution	-	-
`log_regr_base`	Logistic Regression (Regularized)	cont. w/o spectral	✓
`log_regr_names`	Logistic Regression (Regularized)	cont. w/o spectral + ins. names	✓
`log_regr_spectral`	Logistic Regression (Regularized)	cont. + spectral	✓
`log_regr_names+spectral`	Logistic Regression (Regularized)	cont. + spectral + ins. names	✓
`knn1`	K Nearest Neighbor ($k = 1$)	cont. w/o spectral + ins. names	✓
`knn5`	K Nearest Neighbor ($k = 5$)	cont. w/o spectral + ins. names	✓
`decision_tree`	Decision Tree	cont. w/o spectral + ins. names	X

Table 6.1.: Table of the models trained for the classification of couples of instruments at a given measure into blend vs non-blend. The first three dummy models are used as baseline. Every model uses all of the continuous features (cont.), including or not the spectral features. Some models have been trained also with a one hot encoding of the instrument names (ins. names). All models used come from the python library `scikit-learn`.

6.4.4. Model Evaluation

In order to evaluate the model we select a subset of the pieces as test set and use the remaining pieces as training-and-validation set. The extracts of pieces included in the test set are not casual, but are selected among the ones for which Antoine et al.'s model performed best and worst. They are Borodine's *In the Steppes of Central Asia* (measures 40-71), the fourth movement of Schubert's *Symphony 9* (meas. 543-564), and the second movement of Vaughan Williams's *Symphony 8* (meas. 71-107). We train and validate the models on the training-and-validation dataset using a leave-one-piece out approach, *i.e.* we set aside one piece for validation while training on all remaining pieces, we iterate this process for every piece in the train-and-validation set, and we average the results obtained in each iteration.
When evaluating the results, we adopt a measure-as-a-unit approximation, which considers every measure of a given blend as an independent entity, as opposed to the blend-as-a-unit approach, which considers the entire blend as one entity. We are well aware that a blend is a phenomenon that can span several measures, but we simplify the problem by evaluating every measure independently from the others.
The choice of good evaluation metrics for the detection of blends is not trivial, with different points of view that can be taken. We present here three methods: pairwise evaluation, through which we evaluate directly the results of the binary classification, groupwise evaluation, through which we evaluate the capability of the model in correctly retrieving the groups of instruments marked as blends in OrchARD, and partition evaluation, through which we evaluate the capability of the model in correctly partitioning the ensemble into blending groups. Here is a list of all the used evaluation metrics:

- accuracy (pairwise),
- precision (pairwise),
- recall (pairwise),
- f1 score (pairwise),

- precision (groupwise),
- recall (groupwise),
- f1 score (groupwise),
- extra instrument score (groupwise),
- extra instrument number (groupwise),
- extra blends number (groupwise),
- purity (partition).

6.4.4.1. Pairwise Evaluation

The evaluation strategy that is most aligned with our modeling approach consists in computing standard classification metrics (accuracy, precision, recall, and f1 score) for the classification problem. We call these pairwise metrics, because they are computed considering the classification problem of pairs of instruments in each measure. This is opposed to the metrics that will be discussed later, in which we use the classification results to construct groups of blending instruments to compare with the target groups in OrchARD. We briefly remind here the definitions of these metrics.

$$\mathrm{acc} = \frac{\mathrm{tp}+\mathrm{tn}}{N}, \qquad \mathrm{prec} = \frac{\mathrm{tp}}{\mathrm{tp}+\mathrm{fp}}, \qquad \mathrm{rec} = \frac{\mathrm{tp}}{\mathrm{tp}+\mathrm{fn}}, \qquad \mathrm{f1} = 2\frac{\mathrm{prec}\cdot\mathrm{rec}}{\mathrm{prec}+\mathrm{rec}},$$

where tp is the number of true positive samples, fp of false positive, fn of false negative, tn of true negative, and $N = \mathrm{tp} + \mathrm{fp} + \mathrm{fn} + \mathrm{tn}$ is the overall number of samples.

6.4.4.2. Groupwise Evaluation

A different point of view is the one utilized by Antoine et al. [11], which evaluates the resulting blends as groups of instruments. For every blend in the target annotations and for every measure of its temporal extension, they evaluate the performance of the retrieval by computing the ratio between the number of correctly identified instruments and the number of instruments in the target blend. They then average for all considered blends. We call this measure groupwise recall, and we extend the procedure by computing also the groupwise precision, and f1 score. There are two critical steps that are required in order to perform groupwise evaluation: the first one is to construct groups of instruments starting from the pairwise classification, and the second one is to match the constructed groups to the target blends.

Grouping Instruments with Depth-First Search In order to construct groups of instruments we consider a graph for every measure of a piece, where the nodes are the instruments, and two nodes are connected by an edge if the couple is predicted to be blending on the given measure. Then we identify the blends with the maximal components of a graph, *i.e.* the maximal connected subgraphs. An example is shown in Figure 6.11. We use a depth-first search algorithm to retrieve the sets of instruments that constitute maximal components [78]. This algorithm uses backtracking to visit every node, and keeps in memory a list of the visited nodes. It starts at any unvisited node and it searches its neighbor nodes to form a group, adding them to the visited nodes. Once a complete subgraph is found, the process repeats with another unvisited node until all nodes in the graph are marked as visited. With this procedure, some instruments that were not predicted as blending might end up in the same blend group, if there exists a path passing through other instruments between them.

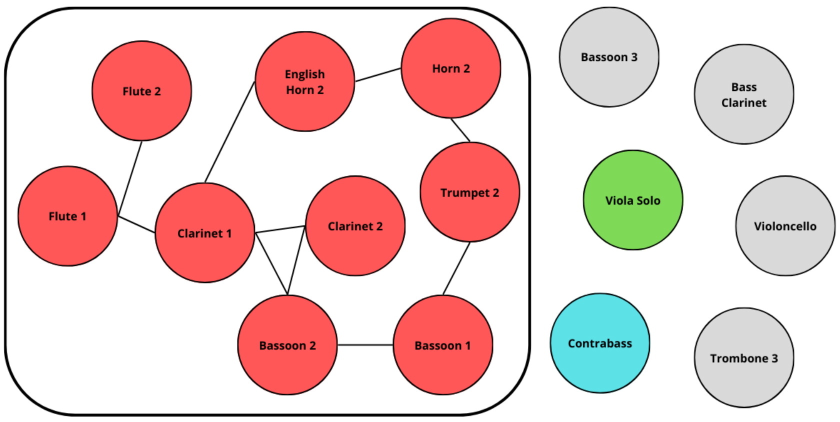


Figure 6.11.: Consider the first measure of Schoenberg's score from the example in Figure 6.3. The depth-first search algorithm retrieves one group in this case, highlighted in red. Not all nodes in the group are mutually connected, but there exist a path between any pair of them. Two instruments are playing but are not blending, and four are silent.

Matching Predicted and Target Groups Once we have constructed our predicted instrument groups, we need to match them to the target blends. This can be framed as a combinatorial optimization problem, more precisely as an assignment problem [146]. For every measure, we consider a list of target blends $T_1, \dots, T_n$, and a list of predicted blends $P_1, \dots, P_\nu$, where $T_i \subset \mathcal{E}, \forall i \in 1, \dots, n$ and $P_j \subset \mathcal{E}, \forall j \in 1, \dots, \nu$. The objective is to find the optimal matching between sets of target blends and sets of predicted blends such that the sum of the number of common elements in each pair is maximized (see Figure 6.12). We can therefore define a cost matrix C for the assignment problem, where the element c_{ij}, with $i \in 1, \dots, n$, and $j \in 1, \dots, \nu$ is given by

$$c_{ij} = -|T_i \cap P_j|.$$

We solve the optimization problem with the python function `linear_sum_assignement` from `scipy.optimization`, which implements a modified Jonker-Volgenant algorithm with no initialization [53].
In the evaluation of their algorithm, Antoine et al. [11] match the predicted groups to the target blends by hand. Whenever their algorithm predicts two separate groups in correspondence of one big target group, they choose to match both predicted groups to the same target group and merge them for the evaluation. One may indeed argue that such a result is close to the target, as the only difference is that the target group has been split in two by the retrieval algorithm. We decide not to merge the groups that have

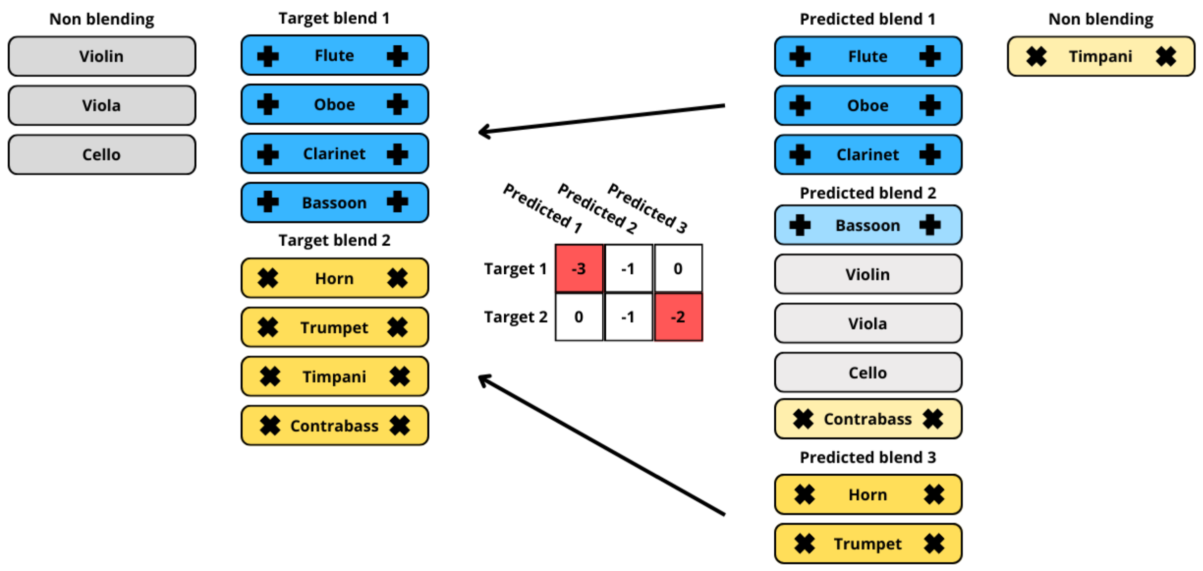


Figure 6.12.: Matching algorithm representation. On the left are target blends, on the the right predicted blends, in the middle the cost matrix for every possible match. The optimal solution is chosen by minimizing the total cost.

been output as separate, and to evaluate the algorithm on its capability to partition instruments into the same blending groups as in the ground truth. If we have the same amount of target and predicted groups, the optimization problem solver gives a one-to-one matching, otherwise there can be an excess of predicted groups or an excess of target groups. We count the number of extra predicted blends and report it with the other groupwise metrics, and we consider excess target groups as not retrieved, contributing to the group precision and recall averages with a value of 0.

Extra Instruments Antoine et al. [11] report a percentage of extra instruments that are added to the blends by their model, but the way it has been computed remains unclear. We report an extra instrument score, which is the proportion of blends (with measure as unit) for which extra instruments are retrieved, and the extra instrument number, which is the average number of extra instruments retrieved on all the blends.

6.4.4.3. Partition Evaluation

A third point of view is to completely avoid the problem of matching predicted groups to target groups, and to look at the results for each measure as a partition of the instruments in the given measure. Metrics for the supervised evaluation of clustering algorithms can be used in this approach. These metrics compare two partitions of the same set of elements, here $\mathcal{E}$, and evaluate the similarity between the two partitions. There exists many different metrics that can be used, but we report here only the purity between the target and predicted partition.

Purity (pur) is a measure of the extent to which predicted clusters contain elements from a single target class. It is defined as

$$\mathrm{pur} = \frac{1}{N} \sum_{j \in \{1,\dots,v\}} \max_{i \in \{1,\dots,n\}} |P_j \cap T_i|,$$

where $\{T_1, \dots, T_n\}$ is the target partition, $\{P_1, \dots, P_v\}$ the partition predicted by the model, and N the number of instruments in $\mathcal{E}$. The purity will tend to be higher if each predicted cluster contain mostly points belonging to the same target subset [168]. Some known problems of this index are its sensitivity to imbalanced data, and to a high number of clusters. Therefore we compare the partitions obtained by considering all blends as distinct clusters and by treating all the remaining out-of-blend instruments as one additional cluster. This means that we would have only one cluster in case there is no blend in a measure, and two clusters in case there is one blend that does not include all of the instruments. We apply the same procedure to the target and to the predicted blends.

6.5. Results and Discussion

We report the preliminary results obtained with pairwise, groupwise, and partition evaluation metrics, using simple statistical models in Tables 6.2, 6.3, and 6.4. The same results are shown through bar plots, for easier readability. Our best performing model is `log_regr_names`, which is a logistic regression classifier, trained on all features except the spectral descriptors, and including the instrument names. It achieves a performance that is comparable to the state of the art model `antoine` [11].

When considering the pairwise performance metrics (Figure 6.13), accuracy is very high, also when considering the dummy models. This happens because the proportion of blending and non-blending pairs of instruments at a given measure is very unbalanced, with 94.0% of samples corresponding to non-blends in the training and validation set. The `dummy0` model that always outputs 0, is therefore correct for 94.0% of the samples. Moreover 84.2% of the samples are excluded from training and automatically classified as non blending for all models, since they contain at least one instrument that is silent on the measure. This fixes to 84.2% a lower bound for the accuracy. A high precision here means that the model is good at keeping the number of non-blending pairs misclassified as blending low; conversely, a high recall means that most of the blending pairs are correctly classified. It is not surprising therefore that `dummy1` gives the highest recall. The f1 score is a balanced measure between the two, and `log_regr_names` and `antoine` obtain similar performances, the first having higher precision, and the second one higher recall. Notice that the `log_regr_names+spectral` obtains the same results as `log_regr_names`, meaning that the inclusion of the spectral features seems irrelevant when the timbre is already characterized by the instrument names.

Our model and `antoine` achieve similar results in groupwise precision, recall and f1 score (Figure 6.14). A high groupwise precision means that the predicted blend contain

Leave One Piece Out (Training and Validation Dataset)				
model name	accuracy	f1	precision	recall
dummy0	**0.940**	0.000	0.000	0.000
dummy1	0.876	0.429	0.296	**0.778**
dummy_unif	0.908	0.338	0.298	0.391
log_regr_base	0.939	0.315	0.482	0.233
log_regr_names	0.938	0.486	0.482	0.489
log_regr_spectral	**0.940**	0.329	**0.496**	0.246
log_regr_names+spectral	0.938	0.486	0.482	0.489
knn1	0.930	0.160	0.286	0.111
knn5	0.931	0.127	0.264	0.084
decision_tree	0.929	0.335	0.387	0.295
antoine	0.934	**0.498**	0.461	**0.542**
Test Dataset				
model name	accuracy	f1	precision	recall
log_regr_names	0.962	**0.301**	**0.189**	**0.740**
antoine	0.962	0.271	0.172	0.649

Table 6.2.: Comparison of the pairwise evaluation metrics. They are computed with a leave-one-piece out strategy on the training and validation dataset (top). Three dummy models, regularized logistic regression, knn, decision tree, antoine. On the bottom the results on the test dataset for our best performing model and `antoine`.

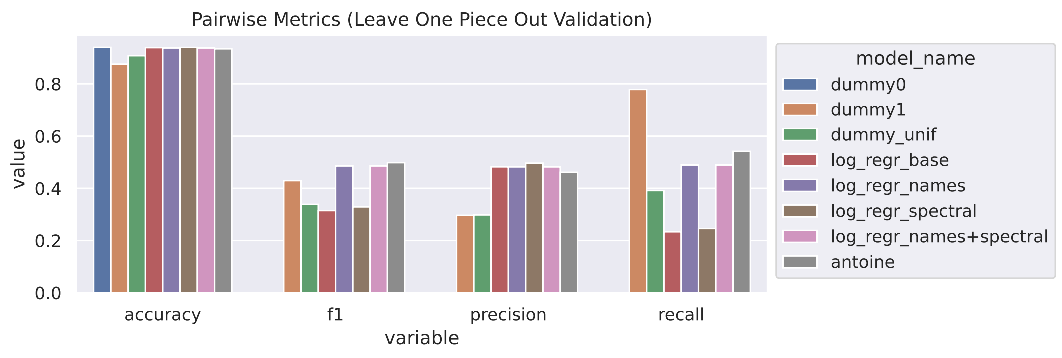


Figure 6.13.: Pairwise evaluation metrics for the logistic regression models.

mostly instruments from the target group; conversely a high groupwise recall means that most of the instruments in the target blend are present in the predicted group, the groupwise f1 score is a balanced measure between the two. The slightly lower precision of `log_regr_names` with respect to `antoine` can be explained by the tendency of the first model to build larger groups, with extra instruments. This also contribute to having a higher recall than `antoine`. One should not be surprised by the fact that the groupwise recall we report for `antoine` model is lower than the performance score reported in the original model paper, as we have used a different approach to match predicted groups to target groups. In our evaluation we do not allow a target group to be associated to more than one retrieved group. It is also to be noticed that `antoine` is the (non dummy) model that has the lowest number of extra retrieved blends.
The partition purity metric gives similar results. A high purity metric is to be interpreted as indicating that the target and predicted partitions resemble, and differ only by a few elements. The results confirm that `log_regr_names` and `antoine` are the two best performing algorithm, and give similar results (Figure 6.15).
We have observed that the use of spectral features does not seem to have an effect on the model performance when added to the model with instrument names. Nevertheless we observe a slightly better performance, on all the metrics, when they are added to the base model (compare `log_regr_spectral` with `log_regr_base`). We would expect a similar improvement to that obtained by adding instrument names. We explain this lower impact as due to the incompleteness of the timbre characteristics dataset. We have access to a lot of detailed information for the analyzed instruments, but we do not have the same analysis for all orchestral instruments appearing in the scores, some of which are missing. Future studies should consider extending the dataset of spectral features.
Table 6.5 compares piece by piece the groupwise recall obtained by our model with the state of the art, both on validation and on test excerpt. The two models are comparable in performance, but there seems to be two groups of pieces. In the first group `antoine` performs significantly better than `log_regr_names`, and on the other one it is the opposite. A more thorough investigation on the characteristics of the pieces in these two groups is required.
Finally, the tables report also the metrics obtained on the test dataset by our best performing model `log_regr_names` and by `antoine`. They have been calculated by retraining the model on the entire train and validation dataset, and by using that model to do prediction on the test dataset. We can observe a similar drop between the two models in all performance metrics with respect to validation. This suggests that the lower performance might be due to the difficulty of these examples rather than to overfitting.
To summarize, we showed that a ML model with knowledge-driven features can match the results of the dedicated algorithm that is state of the art for the problem of identification of orchestral blends. In a perspective study it would be interesting to analyze the coefficients of the best performing statistical models and to look at the results of the prediction on scores, in order to gain a better understanding of their functioning. The results could possibly be interesting for the music perception community, as

Leave One Piece Out (Training and Validation Dataset)						
model name	precision	recall	f1	extra instr. score	extra instr. number	extra blends number
dummy0	0.000	0.000	0.000	**0.000**	**0.000**	**0.000**
dummy1	0.529	**0.796**	0.601	0.708	3.969	**0.000**
dummy_unif	0.516	0.774	0.584	0.706	3.945	0.007
log_regr_base	0.700	0.502	0.550	**0.221**	0.754	0.773
log_regr_names	0.760	**0.733**	**0.709**	0.364	1.452	0.816
log_regr_spectral	0.733	0.514	0.568	0.225	**0.697**	0.873
log_regr_names+spectral	0.760	**0.733**	**0.709**	0.364	1.452	0.816
knn1	0.542	0.470	0.460	0.594	2.379	0.428
knn5	0.574	0.415	0.436	0.532	1.807	0.534
decision_tree	0.576	0.686	0.580	0.616	2.995	**0.186**
antoine	**0.770**	0.682	0.691	0.332	1.039	0.718

Test Dataset						
model name	precision	recall	f1	extra instr. score	extra instr. number	extra blends number
log_regr_names	**0.293**	**0.556**	**0.362**	0.884	3.580	0.621
antoine	0.282	0.435	0.331	**0.696**	**2.072**	**0.318**

Table 6.3.: Comparison of the groupwise evaluation metrics. They are computed with a leave-one-piece out strategy on the training and validation dataset (top). Three dummy models, regularized logistic regression, knn, decision tree, antoine. On the bottom the results on the test dataset for our best performing model and `antoine`.

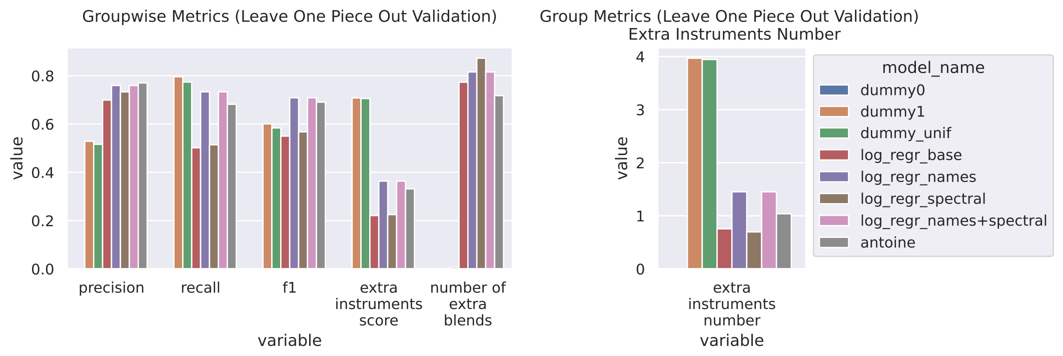


Figure 6.14.: Groupwise evaluation metrics for the logistic regression models. In the graph on the left are groupwise precision, recall, f1 score, the extra insturments score and the average number of extra blends detected. In the graph on the right the average number of extra instruments.

Leave One Piece Out (Training and Validation Dataset)	
model name	purity
dummy0	0.847
dummy1	0.911
dummy_unif	0.910
log_regr_base	0.927
log_regr_names	**0.943**
log_regr_spectral	0.930
log_regr_names+spectral	**0.943**
knn1	0.897
knn5	0.890
decision_tree	0.914
antoine	0.940
Test Dataset	
model name	purity
log_regr_names	0.925
antoine	**0.931**

Table 6.4.: Comparison of the partition evaluation metrics. They are computed with a leave-one-piece out strategy on the training and validation dataset (top). Three dummy models, regularized logistic regression, knn, decision tree, antoine. On the bottom the results on the test dataset for our best performing model and `antoine`.

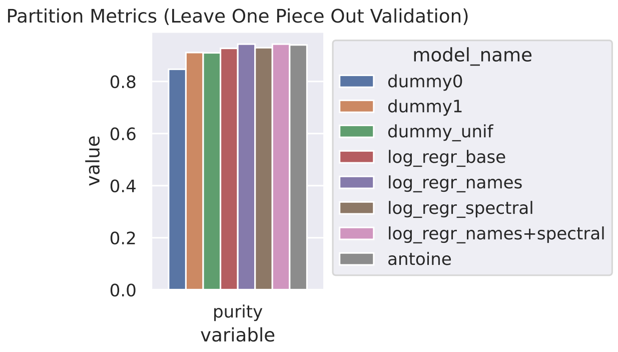


Figure 6.15.: Partition evaluation metrics for the logistic regression models. All logistic regression models give similar purity values to `antoine`.

they would open new ways to investigate the relationship between orchestral scores and blend perception. We would also like to explore more complex machine learning models, to refine the computation of some of the features, and to explore some data augmentation strategies based on the symmetry between instruments in instrument pairs. A comparison with a deep learning model using features autoencoded starting from the note sequence could also be of interest.

Excerpt	groupwise recall antoine t = 60	groupwise recall log_regr_names
Leave One Piece Out Validation Excerpts		
Berlioz SymphFantastique iv (1-77)	**0.795**	0.747
Bizet Carmen Overture (121-147)	**0.810**	0.652
Brahms Symph4 i (1-57)	0.678	**0.711**
Debussy La Mer i (141 mes)	0.423	**0.555**
Debussy La Mer iii-DialogueDuVentEtDeLaMer (31-...	0.556	**0.706**
Haydn Symph100Military ii (1-70)	0.854	**0.933**
Haydn Symph100Military iii (50-65)	**0.851**	0.816
Mendelssohn Symph3-Scottish ii (1-40, 242-273)	0.572	**0.599**
MoussorgskyOrchRavel Tableaux BabaYaga (106-124)	**0.869**	**0.869**
MoussorgskyOrchRavel Tableaux Catacombae (1-22)	0.431	**0.886**
MoussorgskyOrchRavel Tableaux Gnome (57-109)	**0.663**	0.634
MoussorgskyOrchRavel Tableaux Prom-1 (24 mes)	0.883	**0.906**
MoussorgskyOrchRavel Tableaux Prom-2 (12 mes)	0.911	**0.958**
MoussorgskyOrchRavel Tableaux SamuelGoldenbergU...	0.889	**1.000**
MoussorgskyOrchRavel Tableaux VecchioCastello (...	**0.706**	0.529
Schubert Symph8 i (1-62)	0.877	**0.908**
Schubert Symph9 ii (300-310)	**1.000**	**1.000**
Schubert Symph9 iii (187-221, 336-359)	0.800	**0.892**
Sibelius Symph2 ii (150-203)	0.665	**0.866**
Smetana BarteredBride Overture (9-59)	0.896	**0.928**
Smetana Ma Vlast iiMoldau (185-228)	0.332	**0.928**
VaughanWilliams Symph8 iv (12-25, 54-96)	**0.791**	0.642
Verdi Aida ActII-DanzaDiPiccoliSchiaviMori (41-57)	**0.880**	0.800
Verdi Traviata Prelude (17-37)	**0.790**	0.592
Test Excerpts		
Borodine StepsCentralAsia (40-71)	0.065	**0.108**
Schubert Symph9 iv (543-564)	0.500	**1.000**
VaughanWilliams Symph8 ii (71-107)	**0.909**	0.864

Table 6.5.: Excerpt by excerpt comparison of the groupwise recall measure between Antoine et al. [11]'s model and our best performing model. This measure describe the average proportion of instruments from the target blend that are retrieved by the model. On the top are the results of cross validation, and on the bottom the results on the test set.

Part III.

Orchestration and Co-creativity

7. Co-creative orchestration of *Angeles*

This chapter presents a research/creation project for which we have created an AI-human orchestration of two movements of *Angeles*, a piano composition by Gissel Velarde. The origin of this project dates back to the 15th of March 2023, when Gissel Velarde, Bolivian researcher and composer, published a public call on the ISMIR community mailing list seeking for AI orchestrations of some pieces from her composition *Angeles*. Participants were expected to employ any kind of AI method combined with human skills. *Angeles* is a a suite of six short piano solo pieces, recorded in 2011. The first edition of the scores has been published by PRICA Verlag in 2022. We participated by orchestrating the second movement *Inexorable* and the last movement *El Jardin Etereo*, applying the same methodology to two pieces that differ in tempo and style. Our team was composed of four people, including the author of this thesis, his two supervisors, and Mael Oudin, a professional orchestrator and PhD student in music theory at McGill University. The pieces were performed live during a concert by the Orquesta Kronos conducted by Andrés Guzmán-Valdez on the 19th of July 2023 at Nuna Theatre in La Paz, Bolivia. During the concert, the other four pieces, orchestrated by other people with traditional methods, were also performed.

This chapter is partly based on the publication in the conference *evoMUSART 2024* [165]. Together with Mael Oudin, we were nominated as *Outstanding Students of Evo* 2024* for this contribution.

7.1. Goal and Contents

We carried out this project to orchestrate two piano pieces by Gissel Velarde, focusing on *co-creativity*, *i.e.* putting human beings and AI models in the same loop to make music. We used this project as a case study for computational creativity in music and orchestration, where the model has some characteristics of both the *AI as a colleague* and the *AI as a tool* [163, 134]. We not only want the *AI as a tool* to perform certain tedious tasks, but we want the AI to enhance the artists' creativity by suggesting new ideas, that would remain otherwise unexplored. To go towards an orchestration model that can be a *co-creative AI colleague*, we need to depart from black box models that aim at modeling one task in its entirety, and we must target a highly controllable generative model. This project has been used as a practical playground for developing and testing in practice our framework for computer assisted orchestration (Chapter 4). In order to get an outcome of high quality scores, ready to be distributed to the orchestra, in a limited time frame, we focused on the formalization of a possible orchestration

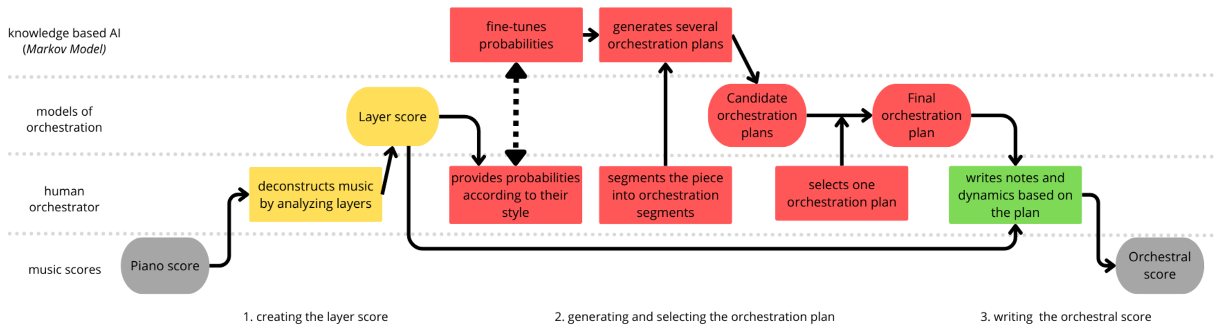

knowledge based AI
(*Markov Model*)
models of orchestration
human orchestrator
music scores
fine-tunes probabilities
generates several orchestration plans
Layer score
Candidate orchestration plans
Final orchestration plan
deconstructs music by analyzing layers
provides probabilities according to their style
segments the piece into orchestration segments
selects one orchestration plan
writes notes and dynamics based on the plan
Piano score
Orchestral score
1. creating the layer score
2. generating and selecting the orchestration plan
3. writing the orchestral score

Figure 7.1.: The workflow for the co-creative orchestration of *Angeles* builds on the *layer score* and on the *orchestration plan*.

process, allowing interactions between a human orchestrator and computational algorithms, and on a simple implementation. We explicitly attempted to model *simple but high-level conceptual data* in a way that it can be handled by AI but that it is also human-understandable, so that the musician can interact with it.
Following Benward and Saker [18], we characterize orchestration in the western classical style as an overlay of different layers, which have the roles of melodies or (harmonic/rhythmic) accompaniment, played by the different instruments of the orchestra and their combinations (see Section 3.1). We then create the *layer score* and *orchestration plan* described in Sections 3.2 and 3.3 of this thesis to realize the orchestration process, implementing a simple Markov model to generate the latter. The AI is personalized for the users through instrumentation presets that are constructed to mimic their orchestration style. AI-human interaction occurs through human segmentation of the score at two stages of the process (layer score and orchestral segments with loudness profile), through the human selection of the final orchestration plan, and through the human writing of the actual orchestral score.
We detail the research aspects of this co-creative project and analyze the roles of the actors involved in the creation of the final piece: the Music Information Retrieval (MIR) researchers, the orchestrators, and the algorithms.

In the framework for co-creative interaction in orchestration presented in Chapter 4, the workflow we adopted in this project is a practical implementation of the computer assisted orchestration process (Task T1, Section 4.2). It is displayed in Figure 7.1:

- The first step is to extract the **layer score** from the piano score by analyzing its musical texture (Tasks T3 and A2). For this step we proceeded manually Section 7.2.

- The second step is to **build an orchestration plan** (Task P3). We assign each layer to an instrument or a group of instruments, taking care of the balance between the different timbres. We have developed a Markov model for this stage, that uses probabilities of finding instruments together, of instrument sequences, and instrumental density, building from instrumentation presets as well as from a segmentation of the score (Section 7.3).

- The last step is to **write the actual instrumental parts** (Task T7), following the orchestration plan and taking care of the peculiarities of range and dynamics of each instrument (Section 7.4). Our orchestrators proceeded manually for this step.

For this experiment, the first and third step were done by human orchestrators, who are also part of the research team for the project: Mael Oudin (professional orchestrator, PO), and Mathieu Giraud (amateur orchestrator, AO). The human-AI co-creative interaction is here mostly concentrated in the second step of the framework introduced in Chapter 4. We neither claim that this three-steps process fully models the art of orchestration, nor that it represents an optimal process, but rather that it is a *plausible workflow* to orchestrate a piece. The process could be followed by human orchestrators alone, but this experiment shows that the framework can be practically implemented in a way that allows the interaction between humans and AI.

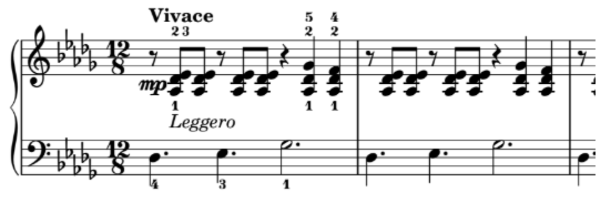
The rest of this chapter details those three steps (Sections 7.2 to 7.4) and the results of this orchestration model, analyzing the roles of the actors – MIR researchers, orchestrators, and computational models. Following that is a discussion of the challenges encountered, our positioning in computational creativity research, and perspectives (Section 7.5).

7.2. Creating the Layer Scores for *Angeles*

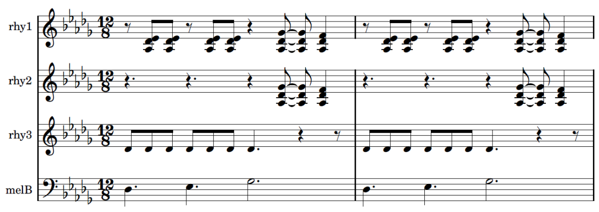
For this project we went through a completely manual creation of the layer score starting from the original piano scores. It is a "creative analysis" process [7], since the analysis is done in support of the creative orchestration process. The orchestrators analyze the pieces to identify layers, and they rewrite the notes into the separate staves of the layer score, one staff per layer. At this point they need to do some abstraction from the typically pianistic textures they might find. After the purely analytic part, the artist is allowed to modify the existing layers, and to add new ones, while being careful in respecting the style and the intentions of the author. This leaves more creative freedom to the orchestrator, allowing for personal interpretation, and for the adaptation of the layers to the envisioned orchestral ensemble.
We illustrate our methodology through the first two measures of *El Jardin Etereo*, the last movement of *Angeles* (Figure 7.2). In the layer score, we have separated into two different staves the melodic λ_{mel} and rhythmic λ_{rhy1} layers that emerge from the piano score. We also decided to stress the two last notes in a separate layer λ_{rhy2} and to add another rhythmic layer λ_{rhy3}. Such decisions in the analytical process have their part of subjectivity and contribute to the co-creativity, allowing to recreate new textures. The professional orchestrator states:

> PO: *My role was to make human musical choices during all stages of the AI-assisted orchestration process. As an orchestrator, identifying the texture is my first job. Analyzing the piano score allows me to deconstruct the music into different roles:*

(a) **Piano score**



(b) **Layer score**



(c) **Segment instrumentation**

$$\mathrm{SI}^{s_1} = \left\{ \begin{array}{l} (\lambda_{\mathrm{mel}}, \{\mathrm{Vc}, \mathrm{Cb}\}), \\ (\lambda_{\mathrm{rhy1+3}}, \{\mathrm{Horn1}, \mathrm{Horn2}, \mathrm{Trp1}\}), \\ (\lambda_{\mathrm{rhy2}}, \{\mathrm{Fl1}, \mathrm{Fl2}, \mathrm{Cl1}, \mathrm{Cl2}\}) \end{array} \right\}$$

(d) **Full orchestral score**

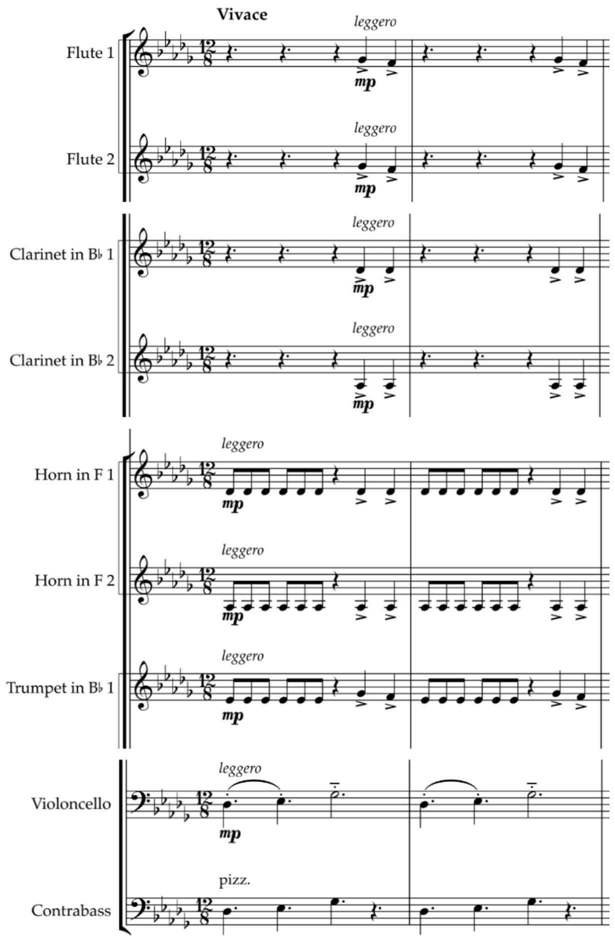


Figure 7.2.: First two measures of movement 6 *"El Jardin Etereo"* from *Angeles* by Gissel Velarde, op. 7. (a) Original Piano score, provided by the composer. The main melody is at the bass, and the right hand plays a rhythmic layer. (b) Layer Score elaborated as an intermediate step of the orchestration process. A melody in the low register λ_{mel}, and a rhythmic layer λ_{rhy1} have been directly identified by splitting the right and left hand of the piano score. Other layers (λ_{rhy2}, λ_{rhy3}) have been added to stress the importance of some notes and to have more possibilities of rhythms, departing from a typical pianistic texture. (c) Segment instrumentation for segment s_1, assigning instruments to each one of these three layers. The human orchestrator decided to have one rhythmic layer $\lambda_{\mathrm{rhy1+3}}$, with added notes in the downbeats. (d) Orchestration of the piece by Mael Oudin. The instrumental parts have been written following the selected orchestration plan. Scores and rendered audio for selected extracts of the piece are available at `http://www.algomus.fr/data`.
`http://algomus.fr/fm/angeles6-1-4.mp3`

the main melody, harmony, rhythm, resonance... I prepare the addition of new parts. The piano is limited by its technique. As I orchestrate, I will add what is "absent but suggested", resonance, missing registers, textures to be recreated. The first stage involved the creation of the "layer score" for this movement.

Creative "*human musical*" choices, such as adding a layer, can already be made at this stage, going beyond a pure analysis of the original score. For example, a music pattern may have at the same time a melodic and an harmonic role. In that case, the layer score could define a mixed melodic/harmonic layer or two distinct layers (see Section 7.5).

7.3. Creating Personalized Orchestration Plans for *Angeles*

We recall briefly our model for an orchestration plan (Section 3.3), which is a function that assigns a group of instruments to any layer in a layer score. When producing the layer score, or starting from the layer score, the orchestrator is segmenting the piece into orchestration segments, which are portions of the piece in which the texture is fundamentally homogeneous, and on which the orchestrator wish for a substantially constant instrumentation. We decide to express the orchestration plan as a set of segment instrumentations (SI), one for each orchestration segment of the score. Each segment instrumentation will match every layer in the segment to its layer instrumentation, *i.e.* a set of the instruments that should play that layer.

7.3.1. Generating Orchestration Plans with Markov Models

Once they have set the orchestration segmentation, the artist could themselves write the layer instrumentation for every layer in every segment of the piece, and create manually an orchestration plan. For this project, instead, we decided to have a simple knowledge-based algorithm to experiment with AI-human interaction. This model for semi-automated layer instrumentation follows two goals. The layers should include instruments that blend together. These are drawn from presets of *possible instrumentations* (defined later). Moreover, the "loudness" of the instrumentation at each segment should be close to the musician's desired outcome, for which they provide a *loudness profile* as input. These concepts are detailed in the following paragraphs.

Loudness profile and acoustic weights. To underline the form, the musician inputs a *loudness profile* as a list of targeted loudness values $(L_1, L_2, \ldots, L_n)$ for the n segments. The effective loudness depends on the dynamics, but also on the number and the qualities of each instrument[1]. A simple model is to consider that each instrument i has an *acoustic weight* w_i. We decided here to have higher coefficients for brasses and instruments of lower range, using the following values, selected by the professional orchestrator in our team based on his experience:

[1]We call "instrument" an instrument group. Groups may include several people (for example, Vl1).

(Fl:1, Ob:1, Cl:1, Fg:1.5, Hrn:1.5, Trp:2, Vln1:1, Vln2:1, Vla:1, Vc:1, Cb:1.5)

The loudness could be directly estimated by combining the weights of the instruments involved (by summing them in a linear or logarithmic scale). However, selecting only the instruments according to such values would not realize a proper orchestration, as it would ignore blending qualities and orchestrator preferences. We proceed instead in defining *possible layer instrumentations*.

Possible instrumentations. For each layer $\lambda_\alpha \in \Lambda$, the musician will thus define a set of *possible layer instrumentations* $p\ell i^\alpha = \{p\ell i_1^\alpha, p\ell i_2^\alpha, \ldots\}$, each one being a weighted list of instruments. For instance, a rhythmic layer λ_{rhy1} could be associated to two distinct instrumentations, either on woodwinds, or brasses:

$$p\ell i^{\text{rhy1}} = \begin{cases} p\ell i_{wood}^{\text{rhy1}} = (\text{Fl} : .3, \text{Ob} : .1, \text{ClBb} : .2, \text{Fg} : .15) & L(p\ell i_{wood}^{\text{rhy1}}) = .825 \\ p\ell i_{brass}^{\text{rhy1}} = (\text{HrnF} : .7, \text{TrpBb} : .2) & L(p\ell i_{brass}^{\text{rhy1}}) = 1.45 \end{cases}$$

Each component (i, f_i) tells that the instrument i should have a probability f_i of being used in this $p\ell i$. The actual instruments that will be used will be a subset of that $p\ell i$. Selecting a $p\ell i$ ensures that these instruments blend together for this particular layer. The sum $\sum f_i$ of the probabilities of a $p\ell i$ is the expected number of instruments in that $p\ell i$. We rather use the *expected loudness* of the $p\ell i$, that is $L(p\ell i) = \sum w_i f_i$, weighting each probability by the acoustic weight of each instrument.

Selecting the $p\ell i$, then the instrumentation for each segment. Given a layer $\lambda_\alpha \in \Lambda$ and a segment $s \in \{1, 2, \ldots, n\}$, the $p\ell i^{\alpha,s}$ is selected in $p\ell i^\alpha$ according to a Markov model (Figures 7.4 and 7.5) that depends on the previous $p\ell i^{\alpha,s-1}$. In order to match the prescribed loudness L_s, the model also tries to minimize $\delta^\alpha = |L_s - L(p\ell i^{\alpha,s})|$ by further multiplying by a penalization coefficient $e^{|\delta^\alpha \tau|}$, with $\tau = 2.0$.
For a given segment s, once all $p\ell i^{\alpha,s}$ are selected for all layers λ_α, instruments are assigned following the individual probabilities f_i. At the end of this step, it may happen that either a layer has no instrument assigned, or that an instrument is assigned to more than one layer. A new iteration of the assignment, based again on the f_i in the $p\ell i$, is used to resolve such cases.

The personalization of the AI "to the style of the orchestrator" is thus obtained through the parametrization of the presets/$p\ell i$ selections. The possibility to steer the AI is also offered by the loudness profile input. Moreover, the implemented method generated segment instrumentations (SI) with three levels of relative loudness for each segment (Figure 7.3b), enabling the orchestrator to further select instrumentations at each segment, but still keeping the coherence of the $p\ell i$.

7.3.2. The orchestration plan of *Angeles*

According to our orchestrator, there are several goals that a good orchestration should pursue: respecting and enhancing the piano composition, a good balance between layers (especially, the melody should not be muzzled by accompaniment), variation

(a) **Loudness profiles**

```
LOUDNESS = [ 0.20, 0.40, 0.20, 0.50, 0.30, 0.40, 0.20, 0.50,
             0.30, 0.50, 0.30, 0.60, 0.40, 0.50, 0.60, 0.70 ]
```

(b) **Generated orchestration plans, with several relative loudness**

```
InstList: <Fl.Ob.ClBb.Fg|HrnF.TrpBb|Vln1.Vln2.Vla.Vc.Cb>
## Gen         104a (0.5)        104b (1.0)        104c (4.0)
[p01]    <..2.|13|...m.>  <2.2.|13|...m.>  <2.2.|13|...mm>   {0.20}
[p02]    <....|..|123mm>  <....|..|123mm>  <...3|m3|123mm>   {0.40}
[p03]    <....|13|..2m.>  <....|13|.22m.>  <....|13|.22m.>   {0.20}
```

(c) **Final orchestration plan**

```
InstList: <Fl.Ob.ClBb.Fg|HrnF.TrpBb|Vln1.Vln2.Vla.Vc.Cb>
[p01] <2.2.|13|...mm>  1:rhy1:brass 2:rhy2:wood 3:rhy3:brass m:mel:mel2 (0)
[p02] <....|..|123mm>  1:rhy1:string 2:rhy2:string 3:rhy3:string m:mel:mel2 (4)
[p03] <....|13|.22m.>  1:rhy1:brass 2:rhy2:string 3:rhy3:brass m:mel:mel1 (0)
```

Figure 7.3.: Creating the orchestration plan of *Angeles*, mvt 6. (a) The score is split by the musician into 16 *instrumentation segments*, each with a target loudness. (b) The model generates, for each segment, three *layer instrumentations* taking into account the expected segment loudness and another relative loudness coefficient (0.5, 1.0, 4.0) (c) In the selected orchestration plan, for the segment `[p01]` (first two measures), there are four layers instrumentations $\ell_{\text{rhy1|brass}}$, $\ell_{\text{rhy2|wood}}$, $\ell_{\text{rhy3|brass}}$, and $\ell_{\text{mel|mel2}}$. The layers are mapped to the instruments appearing in the order declared in `InstList`: For example, the ·"`<2.2.|`" bloc in the woodwinds refers to the layer instrumentation $\ell_{\text{rhy2|wood}}$, with here flutes (Fl) and clarinets (ClBb).

(contrasting moments in the piece should carry different orchestrations), and efficient dynamics (through loudness values and coefficients).
Respecting the piano composition means that the score brings constraints in register and dynamics that need to be reflected in the orchestration plan. For example, movement 6 was a Vivace, with the melody on the bass and an accompaniment more rhythmic than harmonic. The orchestration then had to address the character of the piece and abide by the register of each layer.

> PO: *The bass melody could only be performed by three instruments in the orchestra: the cellos, the contrabasses, and the bassoons. But not every choice would give a satisfying balance to the other layers played by the rest of the orchestra. The bassoons or contrabasses alone, for instance, would not be prominent enough so cellos were necessary here. Any generated orchestration plan that would not nominate cellos for that layer would be in practice almost unusable.*

Balance in an orchestration is also reached through the separation of the orchestra into different groups (namely, the strings, the woodwinds, the brass, and the percussion).

$$\begin{array}{llll llll llll}
p\ell i^{\mathrm{mel},s-1}_{\mathrm{H1}} & \longrightarrow & p\ell i^{\mathrm{mel},s}_{\mathrm{H1}} & :.8 & p\ell i^{\mathrm{mel},s-1}_{\mathrm{H2}} & \longrightarrow & p\ell i^{\mathrm{mel},s}_{\mathrm{H1}} & :.6 & p\ell i^{\mathrm{mel},s-1}_{\mathrm{B}} & \longrightarrow & p\ell i^{\mathrm{mel},s}_{\mathrm{H1}} & :.3 \\
 & \longrightarrow & p\ell i^{\mathrm{mel},s}_{\mathrm{H2}} & :.6 & & \longrightarrow & p\ell i^{\mathrm{mel},s}_{\mathrm{H2}} & :.8 & & \longrightarrow & p\ell i^{\mathrm{mel},s}_{\mathrm{H2}} & :.3 \\
 & \longrightarrow & p\ell i^{\mathrm{mel},s}_{\mathrm{B}} & :.1 & & \longrightarrow & p\ell i^{\mathrm{mel},s}_{\mathrm{B}} & :.1 & & \longrightarrow & p\ell i^{\mathrm{mel},s}_{\mathrm{B}} & :.8
\end{array}$$

Figure 7.4.: Extract of the transition table of the Markov model modeling the evolution of $p\ell i^{\mathrm{mel}} = \{p\ell i^{\mathrm{mel}}_{\mathrm{H1}}, p\ell i^{\mathrm{mel}}_{\mathrm{H2}}, p\ell i^{\mathrm{mel}}_{\mathrm{B}}, \ldots\}$ for movement 6. The transition table was created through iterations between the MIR researchers and the orchestrator. These coefficients are further adjusted by a loudness factor, then normalized.

Harmonic blending is best achieved when all the notes of a chord are performed by instruments in the same group. This was a constraint to our model if we wanted to avoid too much disparity in the instrument combinations proposed in the orchestration plans. Movement 6 had a continuous 3-voice rhythmic layer and we wanted these three voices to be performed by instruments from the same group. This wasn't always the case in the orchestration plans generated so this was also a criteria in the selection of the best outputs.
The ensemble of these parameters creates many constraints on orchestration possibilities, so one of the first human tasks when analyzing the output generated by the model is to remove what seems impossible, for reasons as varied as register limitations, number of voices to be played, and poor blending or contrast. Perspectives include (semi-)automatizing some of these tasks. At the same time, the challenge was to foresee the potential of each generated combination when formalized into a musical score at the next stage of the process. Some combinations, such as opening the rhythmic layer with brass only, were unexpected by the orchestrators but rather "proposed" by the model (Figure 7.3c).

> PO: *I worked on the outer sections of "El Jardin Etereo" using several dozens of orchestration plans generated by the model. It was my responsibility to sort through them and select the most convincing ones according to my taste (while also being open to surprises).*

7.4. Writing and Performing the Orchestral Score

Once the orchestration plan is decided, the orchestrator has a large space of possibilities related to the range, dynamics, and playing techniques of each instrument that can still be creatively explored. The orchestration plan only suggests the instruments to be used in every portion of the piece, but many decisions still need to be taken to get a playable score, in particular to have idiomatic patterns for each instrument of the orchestra.
In the final orchestral score, on the same first two measures (Figure 7.2d), the choice has been made to fill the rest of the first beat of the rhythmic texture to provide a more efficient and easier line to the brass instruments at this fast tempo. The rhythmic layer is then rendered differently from the original pianistic texture, but it preserves the intention. Likewise, the choice of writing pizzicati for the contrabass part, to lighten

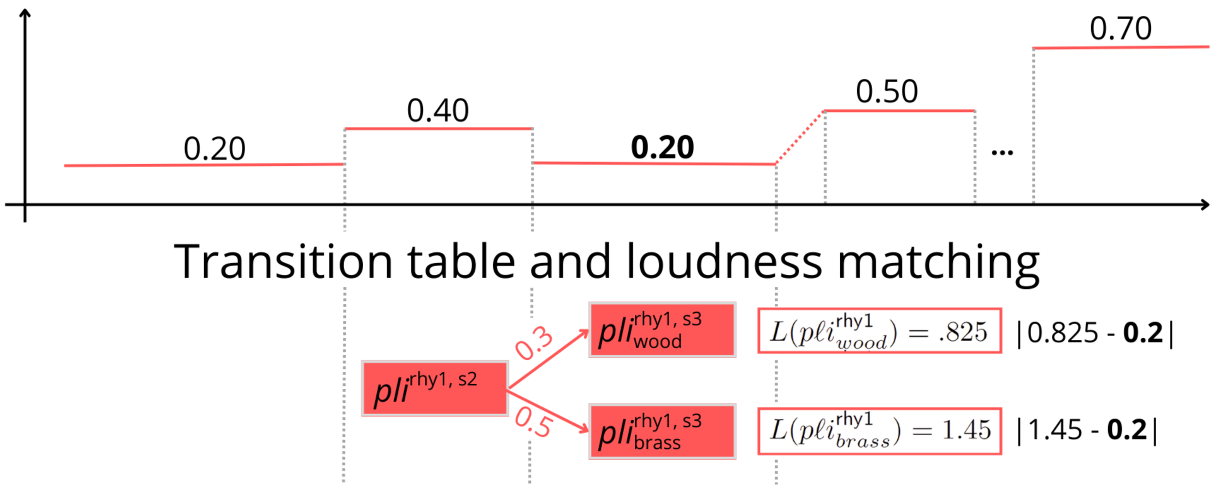


Figure 7.5.: Example of $p\ell i$ selection with the Markov model. For any segment of the piece, and any layer of the segment, the model selects the $p\ell i$ based on the $p\ell i$ for the same layer in the previous segment. The probabilities of selection depend on a transition table with weights that are always the same (here represented by the numbers on the red arrows), and on the distance from the target loudness. The model tries to match the desired loudness (here represented by the graph above). The coefficients from the transition table are multiplied by a penalization coefficient representing the cost of the loudness mismatch. In this case the woodwind $p\ell i$ will have higher probability of being selected, since it matches better the target loudness profile, even though the coefficients from the transition table would favor the brass choice in general.

the orchestral texture and express the *mezzo-piano* dynamic, was taken at that stage of the process.
Our orchestrator shared some reflections on the artistic side of this final step:

> PO: *The distribution of instruments is suggested by the model, but there is still considerable freedom in the choice of notes and registers. It is also up to me to choose and indicate nuances, phrasing, playing modes and expressive indications. Some of the model's choices wouldn't have been what I would have done, like starting directly with the trumpets at the beginning of movement 6. But it's stimulating!*

Movement 2 *Inexorable* and the outer sections of movement 6 *El Jardin Etereo* have been orchestrated with this procedure whereas the middle section of movement 6 has been orchestrated with a "traditional" method. The other movements have been commissioned to orchestrators outside of our team. The whole suite has been performed by the Orquesta Kronos conducted by Andrés Guzmán-Valdez at Nuna Theatre in La Paz, Bolivia, on the 19th of July 2023 (Figure 7.6).
The experience of working with different versions of the model, with increased improvements, showed that the AI-assisted method begins to offer a gain in productivity

Figure 7.6.: The whole suite *Angeles*, including our co-creative orchestrations for movements 2 and 6, has been performed by the Orquesta Kronos conducted by Andrés Guzmán-Valdez at Nuna Theatre in La Paz, Bolivia, on the 19th of July 2023.
`http://algomus.fr/fm/angeles-piano-orchestra-video.mp4`

once the orchestration plans start to be more reliable. It is also a tool for creative thinking:

> AO: *Like any creative work, an orchestrator may face the anxiety of the blank page. Especially as an amateur orchestrator, I enjoyed having such suggestions. Even when they were inappropriate, they stimulated creativity through reinforcement, contrast, or opposition.*

7.5. Discussion and Perspectives

In [61] the use of *low-tech AI* is advocated, to ease the communication between the composer and the researchers, and to obtain tailor-made models with scarce data. We adopted a similar approach here, and we focused on modeling the process of orchestration by identifying possible steps that can be accomplished with human-AI co-creative interaction. The computational model used to generate orchestration plans has been conceived and designed with a continuous back and forth between the artist and the research team. It is meant to be the simplest possible, so that it can be more easily modified to experiment with different inputs and controls. Concepts like loudness profiles and coefficients have been added to the model to respond to the ideas and the necessities of the orchestrator.
The results and the testimonies of our orchestrators lead us to believe that we have succeeded in individuating three well-separated stages of orchestration (Figure 7.1), which could be performed by three different (human or algorithmic) actors. Each step included a self-refining feedback loop. For example, the human task at the final stage of

the process (3. writing the orchestral score) can be described in two phases, the second of which is usual for "traditional orchestration": (3.1) interpret and adapt the orchestration plan, (3.2) write notes for the instruments. In the first stage, the orchestrator will read the output made by the machine and mentally link it to the score to find the best strategy to transform these outputs into music notes (this phase is fundamental to selecting the best outputs from the machine). The second phase consists of writing the notes idiomatically for the instruments, but also the dynamics and the phrasing and expressive instructions.
The model applied here has limitations. The process is not always linear: it was sometimes in phase (3.2) that choices made in phase (3.1) retrospectively appeared to be pitfalls, and the entire process had to be made again (when, for example, a particular arrangement was not compatible with the plan selected for the subsequent section). Other points could also be improved, as for example the formation of mixed layers. More generally, a challenge is to better model *large-scale orchestral thinking*. Orchestral contrast is typically achieved when the same instrument (or combination of instruments) is not used in the same way in two successive contrasting parts. This may lead orchestrators to "reserve" an instrument on purpose for a specific moment in the piece. Somehow, the `pli` presets combined to the acoustic weights and the targeted loudness profiles help such a large-scale homogeneity and steerability, but these models could be refined.

Looking back to the categories proposed by Kantosalo and Jordanous [134], we have experimented with a process rooted in the interaction between the computational models and the artist, in which the role of the model could be described as in between *AI as a colleague* and *AI as a tool*, having some characteristics of both. Dividing the orchestration process into steps has facilitated the introduction of computational models. In this way, the role of the AI is to act on a well defined and specific task, making the model an essential *tool* in the overall process. The algorithm is acting on the product of human actions (the layer score), and is enabling further human processing with its output (writing the final orchestral score). At the same time, the model is able to "suggest" unforeseen ideas to the humans, acting more as a *co-creative colleague*, who can inspire and enhance the inspiration of the human artist. Some questions remain open: *Is the human or the machine in control? What is the perceived role of the AI?* A more thorough investigation on agency [141] and on the perception of the model by the human artists involved in the process is to be explored in future studies.
Any co-creative project confronts us with questions related to the authenticity and the ownership of the such art [181]. When dealing with AI-generated art, ethical, legal, and moral concerns emerge, questioning the status of the product itself as having artistic qualities [180]. In this research and creation project, the development of the algorithm and the artistic creation of the orchestrated score were intertwined processes. For this reason, the scores have been signed with *Orchestrated by Mael Oudin and the Algomus team* (mov. 6) and *Orchestrated by Mathieu Giraud and the Algomus team* (mov. 2), recognizing authorship to all members of the project, musicians and computer scientists, but not to the AI model.

In summary, with this project we first proposed and applied our framework for AI assisted orchestration, in which the orchestrators craft their art in collaboration with AI algorithms. We divided the process into three steps, modeling *layer scores* and *orchestration plans* as intermediate objects. The code for generating orchestration plans is available under an open-source license at `http://algomus.fr/code`. Through this preliminary project, this approach has proven to be effective in formalizing the art of orchestration, enabling the involvement of both machine and human actors, each contributing at different moments. The possibilities for the employment of AI in the process are not limited to the ones selected for the scope of this project. Perspectives include modeling other tasks in the process with Deep Learning AI, both for texture analysis tasks related to the creation of the layer score, and for constrained notes generation, in the creation of the final score. Co-creative interactions would be allowed through model parameters, and through creative modifications of the outputs at several stages: when writing the layer score, the orchestration plan, and the final rendering of the notes. Some of these ideas are examined and experimented in Chapter 8 of this thesis in which we try to address orchestration tasks using Deep Learning models with a Transformer architecture. All these steps can be accomplished partly by the machine and partly by the human being, with a fruitful continuous exchange of information.

8. Orchestration with Transformers

In this last chapter we present a series of experiments in AI-assisted orchestration with Transformer models [265], aimed at performing some orchestration-related generative tasks. Similarly to what we did for the *Angeles* project (Chapter 7), our objective is to conceive and develop algorithms and models that can be used by professional or amateur orchestrators in a co-creative way. Like in the previous chapter, we aim to build models that go beyond the *AI as a tool* and that can act as a *co-creative colleague*. In the context of our framework for co-creative interaction in orchestration (Chapter 4), we separated the orchestration process in three steps. The models that we describe here focus on the second and third steps, and aim at generating orchestral scores conditioned to the input of a layer score (Task T6) and possibly of an orchestration plan (Task T7). We base our model architecture on SymphonyNet [158], and we try to adapt it to our goals. Our main contribution is an extended tokenization strategy, allowing to represent different objects like orchestral scores, with or without texture annotations, layer scores, and orchestration plans. This proposal of an extended tokenization was part of a first submission to the ISMIR conference, but the work was too preliminary to be accepted. Even though the results are not always satisfactory, we report here a complete chronology of the experiments conducted, along with some reflections on each of them, which may be useful to those who intend to pursue research in computer assisted orchestration.

This chapter is organized as follows. We start by presenting our new tokenization framework, common to all experiments, that has the capacity to simultaneously encode orchestral music, layer scores, and orchestration plans (Section 8.1). Then Section 8.2 is divided into four parts, corresponding to each of the experiments that have been conducted. Finally Section 8.3 tries to draw some conclusions and to lay the basis for further work.

8.1. An Extended Tokenization of Music and Texture

To process data with Transformer architectures (see next section) we need to encode the information into a sequence of *tokens*. In this section we present a comprehensive strategy to tokenize both music and texture information. We extend the SymphonyNet tokenization method [158] with new *textural* and structural tokens, identified in the text below with a bell (🔔) and summarized in Table 8.1. These new tokens describe orchestral texture according to Section 3.1 and [149], and can be used in combination with the original SymphonyNet tokens to model orchestration tasks from Table 4.1

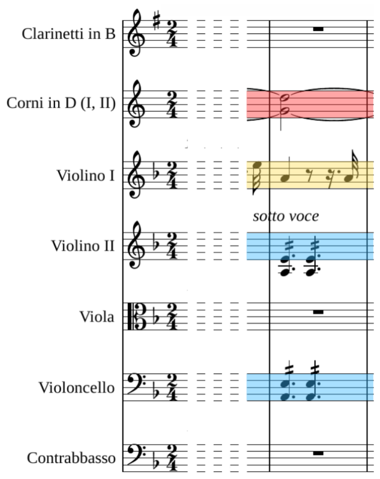
involving multiple objects (for example task T6 Layer Score Orchestration). This tokenization strategy unifies the representations of different objects (orchestral music, texture annotations, layer scores, and orchestration plans) into one approach using the same tokens (Figure 8.1). Similarly to the Compound Word [116], and the Octuple [282] tokenization stategies, the Multi-track Multi-instrument Repeatable (MMR) representation used in SymphonyNet uses embedding pooling to merge in the same "compound token" different characteristics of notes and structure elements. Those different aspects are organized in separated vocabularies, and are converted to embeddings independently in the model. They are then summed together (merging/pooling) as for the position embedding. The following tokens are used (Table 8.1):

Instrumentation			Voc.	Occ.
(🔔)	IIE_i	Instrument i in ensemble	14	176518
Structure				
	BOS	Beginning of score (or training example)	1	8686
	EOS	End of score	1	8686
(🔔)	BOT	Beginning of textural section	–	–
(🔔)	EOT	End of textural section	–	–
	BOM_ℓ	Beginning of measure of length ℓ	51	104232
	POS_j	Position j in the measure	97	1523705
	NT	New Track	1	696635
(🔔)	BLS	Beginning of layer score	1	8686
(🔔)	BOP	Beginning of orchestration plan	1	8686
(🔔)	BFS	Beginning of full orchestral score	1	8686
Layer				
(🔔)	$Role_i$	Role (melody, harmony, rhythm, sparse)	5	490206
(🔔)	Rel_{unison}	Relation (unison, parallel, homorhythm)	4	490118
(🔔)	TIL_t	Track t in layer	40	1568378
Chords and Notes				
	$Chord_c$	Chord c	134	104232
	Pitch	Pitch	127	2326690
	Duration	Duration	32	2326690
	Track	Track	40	2797235
	Instrument	Instrument	14	1461631

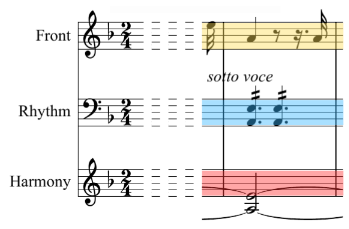
Table 8.1.: Token types, with the number of their appearances in the vocabulary (Voc.) and their occurrences (Occ.) in the finetuning data with the layer-plan-full strategy (Figure 8.4c) used in the last experiment (Section 8.2.4). For example, there are 14 different IIE_i tokens in the vocabulary, one per instrument i, and they appear overall 176518 times in the finetuning data. Notice that the track number is embedded also for other things than notes (for example in the orchestration plan), and that the instrument is not embedded or predicted for every note (for example in the layer score).

- **Instrumentation (🔔):** The optional IIE_i tokens, where i represents an instrument, can be used to weakly constrain the orchestral generation on the available ensemble. According to the task, they can be positioned at the start of the piece or elsewhere.

(a) Full orchestral score with textural labels



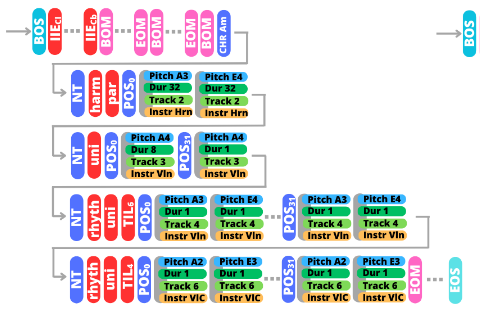
(b) Layer score (with textural labels) (see Section 3.2)



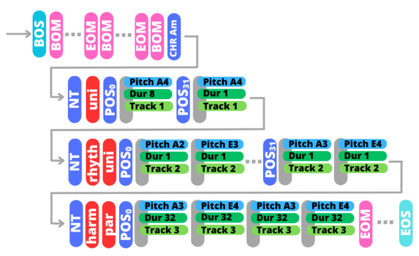
(c) Texture annotations, interpretable as an orchestration plan (see Section 3.3)

```
[3] <0har0r0>
a:-u
h:harm-p
r:rhythm-u
```

(d) Full orchestral score with textural labels tokenization



(e) Layer score (with textural labels) tokenization



(f) Orchestration plan tokenization

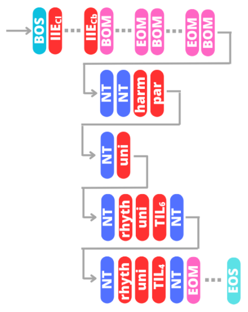


Figure 8.1.: Tokenization of the first movement of Symphony No. 9 by Beethoven with a measure-level description of orchestral texture and layers. Tokens are defined in the main text. The figure focuses on the 3rd measure, where horns are playing an harmonic layer in parallel homorythmy ($\mathsf{Role}_{\mathsf{harmony}}\mathsf{Rel}_{\mathsf{parallel}}$), first violins a front layer with unspecified role, in unison ($\mathsf{Rel}_{\mathsf{unison}}$), and second violins and cellos a rhythmic layer in unison ($\mathsf{Role}_{\mathsf{rhythm}}\mathsf{Rel}_{\mathsf{unison}}$). (a) Reference full orchestral score. (b) Reference layer score. (c) Reference texture annotations. (d) Tokenization of the full orchestral score with textural labels. The instrument is given for each note token, and TIL_j indicates which tracks belong to the same layer – such as the altos and cellos (TIL_6 and TIL_4). (e) Tokenization of the layer score. Instruments are not given, but each layer is represented with its role and relation. Here the rhythmic layer between altos and cellos is given only once. (f) Tokenization of the texture annotations alone, interpretable as an orchestration plan, describing the layer repartition without the notes. Other data could be encoded similarly, as for example a multitrack score without texture tokens. The three encodings reported here are only some examples. We used this precise tokenization in the first of our experiments, but in some others we used different token orders and we included more or less information.

- **Structure:** Begin/end of a piece/score, of a textural section (🔔), of a measure of length ℓ, of a layer/full score segment (🔔), of an orchestration plan segment (🔔), and position in the measure $j \in [0, \ell]$. Lengths and offsets are expressed in 32nd notes, and a measure is a quantized grid [158].

- **Track:** A token indicates the start of a new track (NT), decoupling the track information from the instrument information. In all our experiments, the track changes multiple times within a measure, and the NT token separates notes belonging to different tracks in the given measure.

- **Layer description and composition (🔔):** Specific tokens can describe zero, one, or several layer role(s) and possible in-layer relation following [149]. Their placement depends on the model, for example at the start of each track or at the start of a layer (in an orchestration plan). A list of TIL_t, track t in current layer, can be used to describe the composition of a layer. It can be placed in every track, asserting that track t is in the same layer than the current track (see altos/cellos on Figure 8.1d). Or it can be used to describe the composition of a layer in an orchestration plan.

- **Chord:** At each measure, a token is used to describe harmonic content. Harmony is not always stable in a measure, but this token has proven to be effective [158].

- **Notes:** Notes are represented by a compound token, which immediately follows a position POS_j token (already discussed in structure). The compound token merges a group of four tokens: Pitch, Duration, Track, and Instrument. These four pieces of information are embedded separately and the embeddings are summed in the model. Not all four tokens are compulsory, SymphonyNet [158] does not embed the instrument information in the model input but it does estimate it in the output, and tracks could be dropped if we are not interested in separating repeated instruments.

Compound tokens can be used not only for notes: the NT token, position tokens, and texture tokens (depending on the use) can also be pooled with the track information. In order to simplify embedding pooling, tokens are organized into five vocabularies: event (instrumentation, structure, chord, and pitches), duration, track, instrument, and texture (layer description and composition). The vocabulary sizes are respectively 432, 39, 47, 21, and 56 tokens, with a total of 567 distinct individual tokens (Table 8.1)[1]. These tokens allow to represent most input/output orchestral data presented on Table 4.1 in a cohesive way:

- **Full orchestral score, with texture analysis:** all tokens, including texture. IIE_i tokens are optional, since their information is already carried by the Instrument token at every note event.

- **Multitrack score:** all tokens, except texture tokens with layer description and layer composition. IIE_i tokens are optional.

[1]Summing the sizes of the five vocabularies gives a number that is grater than the total number of individual tokens, since 7 "special" control tokens are replicated in all of the 5 vocabularies

- **Layer score:** all tokens, except IIE_i, and TIL_i. Only one track per layer should be present. There is no embedding of the instrument information, since layer scores have no instruments. In certain experiments we embed a special "`LAYER`" instrument instead.
- **Orchestration plan:** all tokens, except positions and notes.

Chord_c, NT, and texture tokens, are placed within each BOM_ℓ / EOM_ℓ pair, as in SymphonyNet (Figure 8.1), together with the positions and notes inside each track. They could also be placed into a larger BOT / EOT pair to group together measures with the same texture.
Figure 8.1 shows some of the proposed token sequences on the opening of Beethoven 9th symphony. Note that the tokens representing layers in the layer score and in the orchestral score are ordered differently. More generally, certain token sub-sequences have to be "permutation invariant": the tracks can (and, to prevent overfitting, have to) be presented in any order.
We also adopt the 3D position embedding following [158], with the three axes note order (semi-permutation variant), measure order (semi-permutation variant), and track (permutation invariant).

8.2. Experiments

We report here a series of experiments with models extending SymphonyNet [158] with the extended tokenization strategy presented in the previous section. In these experiments we attempt to model some orchestration tasks from Table 4.1 taking into account texture and layers: (U1) Unconstrained Symphony Generation, (C1) Harmony Constrained Symphony Generation, (S1) Unconstrained Reorchestration (reinstrumentation), (A1) Orchestral Texture Analysis, (T6) Layer Score Orchestration, and (T7) Texture-constrained Orchestration from Layer Score.

8.2.1. First Modeling Experiment: Adapting SymphonyNet with Texture

In the first experiment we applied little modifications to SymphonyNet [158], trying to model the tasks of unconstrained generation, harmony-constrained generation, unconstrained reinstrumentation, texture analysis, and layer score orchestration.

8.2.1.1. Model Description

Proof-of-Concept Model Architecture. We extend SymphonyNet [158], an auto-regressive model based on a decoder-only linear Transformer architecture, with a semi-permutation invariant 3D embedding. The model has now five feed-forward heads, to classify the outputs: instrument, track, duration, event, *and texture*. The tokens are collected into five vocabularies, they are embedded independently, and pooled. Instrument information is not provided in the input, forcing the model to learn how to do "instrumentation" (instrument classification) on the side.

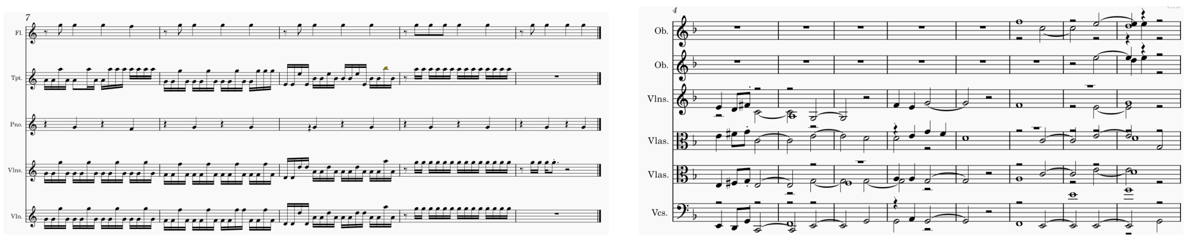


Figure 8.2.: Task (U1), some "best extracts" humanly selected from unconstrained generations by the (left) pre-training and (right) fine-tuning models.
`http://algomus.fr/fm/generated-pretrained.mp3`
`http://algomus.fr/fm/generated-finetuned.mp3`

Tokenization. We implement a tokenization for the full orchestral score (with and without texture annotations), and for the layer score, as described in Section 8.1. We do not apply Byte Pair Encoding to our tokens, as it would create *supertokens* that are too connected to the corpus, making a pretrained model less suitable for transfer learning. The permutation invariance option from the original SymphonyNet was also deactivated in this first experiment, to handle more easily the relationship between track numbers and TIL_t tokens.

Dataset and Training Strategy. The corpus gathers 20,000 multi-track MIDI files from SymphonyNet [158] as well as 7 (expanded to 21 in the following experiments) of the 24 movements of classical and early-romantic symphonies annotated with texture and layers from Chapter 5 and [149]. As the latter is too small to train a large generative model, we adopt a *transfer learning* strategy. We first pretrain our model on the large corpus (99% training, 1% validation), and finetune it for different tasks using the texture and layer corpus. For both trainings, we hold out one piece as our test set. Pretraining took 48 hours on a high-performance server equipped with 2 GPUs Nvidia Turing RTX 2080 Ti[2]. Finetuning for the tasks described in the next section took about 1 hour per task.

8.2.1.2. Tasks Details and Results

We finetune the model to perform the following tasks (compare with Table 4.1).

(U1) Unconstrained Symphony Generation. As in the original SymphonyNet model, the outputs are, in our opinion, far from being close to a "well-orchestrated score". Indeed, the model outputs pitches relevant to some harmony, that is most of the time within a coherent tonal progression, but lacks long-term coherence and consistent phrasing. However, outputs of the new model seem to exhibit more "classical-romantic" textures that may have come from the additional data and texture/layer modeling (Figure 8.2). It should be noted that many other passages do not exhibit the same good quality.
These best examples are weak compared to what can be written by human orchestrators (and contain instrumentation errors), but the new model seems to output more layered

[2]Server provided by the Mésocentre de Calcul Scientifique Intensif de l'Université de Lille, `https://hpc-doc.univ-lille.fr/`

music, with, on that example, winds entering at the second half of the extract. To evaluate whether this improvement could come from the new texture data, we run an internal blind test to evaluate the texture of unconstrained generations from the model finetuned *with the same finetuning corpus*. On one side, we included texture tokens (3/5 acceptable textures), and, one the other, we did not include them (1/5 acceptable textures).

(C1) Harmony Constrained Symphony Generation, (S1) Unconstrained Reorchestration (re-instrumentation). Processed with the same model but with a different sampling procedure, these tasks give similar results. Like for the unconstrained generation, these functionalities are already possible with SymphonyNet [158].

(A1) Orchestral Texture Analysis. We finetune the model for texture analysis by making the model predict a sequence of tokens representing the texture annotations from the content of a measure. We achieve that by inverting the position of layer tokens and note tokens in each track of a measure with respect to Figure 8.1d. The model achieves a cross-entropy loss for texture prediction of 0.967 on the test set. Further investigation work should be done to understand these results, both in statistical and musical terms. A more suitable target loss, and adequate texture recognition metrics should be developed. In the following iterations of the model, however, we completely abandoned the texture analysis task, to concentrate our efforts on generative tasks.

(T6) Layer Score Orchestration. To keep the same basic architecture than the previous tasks, we alternate the tokens for one measure of the layer score and one measure of the full orchestral score (see Figure 8.4a). The model is therefore finetuned to generate the next measure of an orchestration from the layer score for that measure and a given context (the previous measures layer score and orchestral score). The results on this task are deceiving. We were able to obtain an orchestration of a toy example (Figure 8.3) which respects the given layers. However, no other successful example of orchestration could be produced, with the model reacting as in the unconstrained generation task. We initially attributed this behavior to the structure of the training examples, with alternated measures between layer scores and orchestral scores. We interpreted that the model learned how to create a new layer score, and not only how to orchestrate a layer score. We see in the next experiments, that the problem persist also with more careful presentations of the finetuning data. The problem of having the model reproduce a given sequence of notes, like a melody, in the output is very hard to solve with weak constraints, like texture tokens in the sequence [242, 150].

8.2.2. Second Modeling Experiment: New Sequential Strategy for Finetuning, Separating Layer Scores from Orchestral Scores

In the second iteration of the model we focus on generative tasks, and in particular on unconstrained and texture-constrained orchestration (T6 and T7), which are still open and unsolved problems. We keep the same pretraining structure, but we modify the way we present finetuning examples to the model, and the sampling strategy.

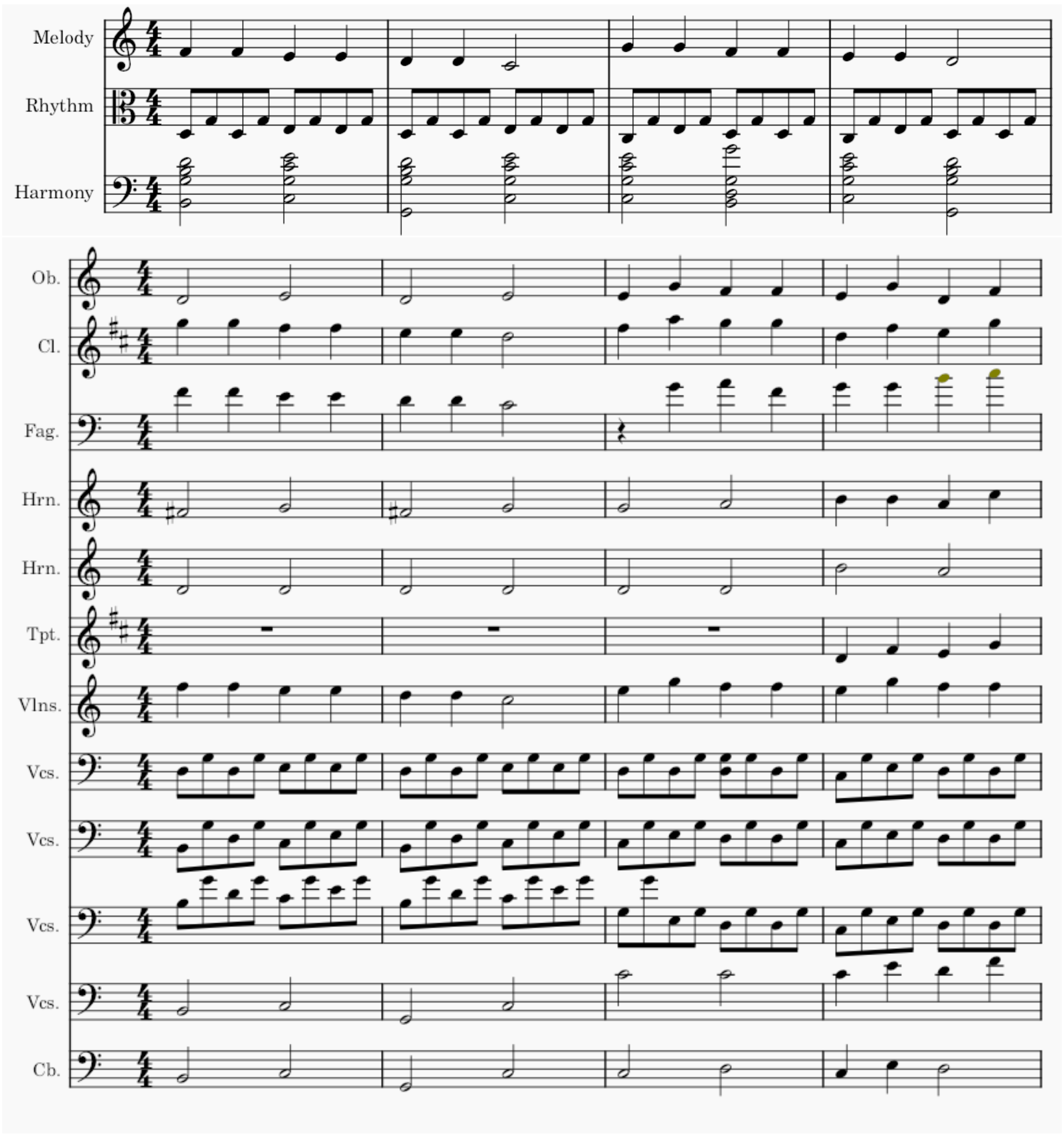
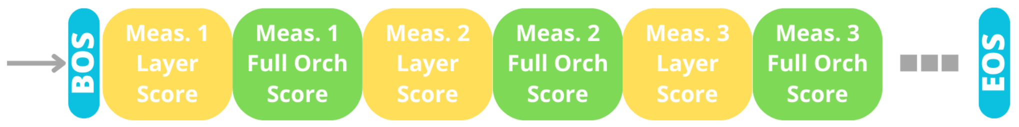
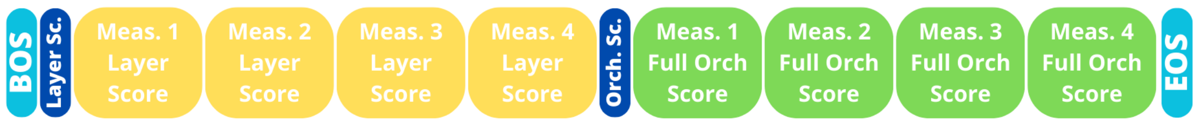


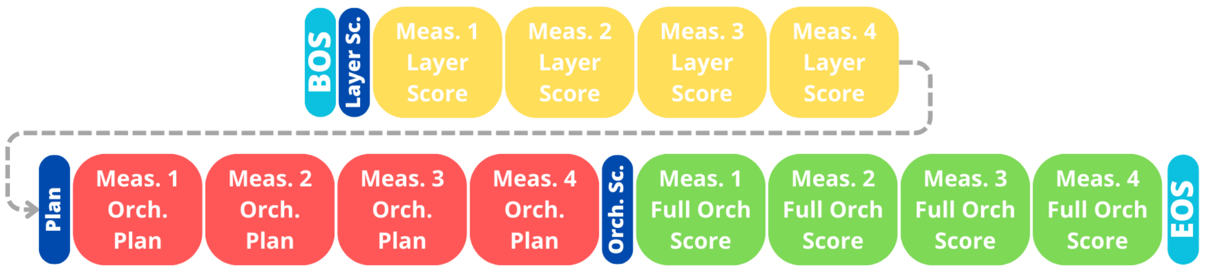
Figure 8.3.: Task (T6), orchestration from a 3-part layer score of the popular theme *Twinkle Twinkle Little Star*. Given the layer score (shown on the top) and an orchestration of the first two measures, the model re-instrumented these two measures and generated an original orchestration of the last two ones (score on the bottom).
http://algomus.fr/fm/twinkle.mp3



(a) Alternated strategy used in the first modeling experiment



(b) Sequential strategy used in the second modeling experiment



(c) Layer-plan-full strategy used in the third and fourth modeling experiments

Figure 8.4.: Comparison of different fintuning strategies for the unconstrained and the texture-constrained layer score orchestration tasks.
(a) Alternated strategy. In the first modeling experiment the finetuning examples present measures from the layer score alternated with their respective measure from the orchestral score.
(b) Sequential strategy. In the second modeling experiment the finetuning examples present four measure from the layer score followed by a separator token and by the same four measures from the orchestral score.
(c) Layer-plan-full strategy. In the third and fourth modeling experiments the finetuning examples present four measure from the layer score, an orchestration plan for the same four measures, and the orchestral score realization of the plan. Separator tokens are inserted between the layer score, the plan, and the full orchestral score.

8.2.2.1. Model Details

Finetuning and Sampling. We build our finetuning examples in the following way (see Figure 8.4b). We segment the pieces from the annotated symphonies corpus into samples of four measure, and we construct examples that present four measures of layer score followed by the same four measures of the orchestral score, including texture tokens. The finetuned model can be used to generate layer score orchestrations in two modes. In unconstrained mode (T6), we give as prompt the measures of layer score, together with one or two measures of orchestral score, and we sample the remaining measures of orchestral score. In texture-constrained mode (T7), we do similarly, but we enforce the presence of the texture tokens from an orchestration plan after every NT token.

Layer Score Data. In this phase we made a step forward in the algorithm that generates layer scores from tokenized orchestral scores with annotations, solving problems related to the correct handling of out-of-layer notes (discussed in Appendix B.1). Thanks to these improvements we were able to increase the number of available scores for finetuning. However the algorithm used here is not yet in its final state (Algorithm 1 introduced in Appendix B). We also reconsider the way we encode instruments in the model, moving from MIDI name to a smaller set of orchestral instruments.

8.2.2.2. Qualitative Evaluation of the Results

In both cases the model does not produce convincing results. It seems very hard to find a balance between overfitting and underfitting while finetuning. In the first case, the generated examples exhibit idiomatic patterns of the classical style of Mozart, Haydn, and Beethoven, similar to those obtained in the previous experiment on unconstrained generation (see Figure 8.2), but they are completely unrelated to the provided layer score. In the second case, the model does not know how to behave after tokens that were not present in the pretraining phase.

8.2.3. Third Modeling Experiment: Permutation Invariance Improvements and Implementation Flexibility

In the third experiment we try to make improvements in different directions while keeping a high level of flexibility, so that different combinations of the new modifications can be tested. The tasks we study are still unconstrained and texture-constrained orchestration (T6 and T7).

8.2.3.1. Model and Data Improvements

Model Modifications. Here are some of the different configurations that we have tried.

- We implement a new strategy for texture constrained orchestration (layer-plan-full, see Figure 8.4c). It consists in presenting the entire orchestration plan to the model in a position prior to that of the orchestral score. When generating an orchestration autoregressively, then the model has access to the entire plan.
- We reintroduce the functionalities exploiting permutation invariance that were present in SymphonyNet [158].
- Segment embedding: we call segments the different parts of a finetuning example. One segment corresponds to the layer score, one to the orchestration plan, and one to the full orchestral score.

- We test with different configurations of the embeddings. For example, we can embed the instrument information, combine it with the track information, or replace the track with the instrument.
- We reserve the lowest N track numbers to layer scores (for example N = 10), and track numbers starting from N + 1 to orchestral scores.
- We test with the smallest possible model.

Data. In this phase we also improve the quality of the layer scores enforcing continuity of layers between adjacent measures, reaching the current state of Algorithm 1 (Appendix B). We implement a data augmentation strategy based on transpositions. We are aware that transpositions can create many orchestration problems. We apply this data augmentation by taking care that the transposed scores do not contain notes out of the original range of the instruments.

8.2.3.2. Qualitative Evaluation of the Results and Discussion

While the generated orchestration is generally respecting the harmonic progressions in the layer score, the melodic lines are never reproduced correctly from the layer score to the orchestral score. The impression that we have when looking at the generated examples is that the model is left with too much freedom, and it is not following the hints given by the texture tokens, that are not real constraints. One way to solve the problem could be to extend the finetuning data with a lot more manually annotated examples, but this is out of reach because it would require more intense human labor. The alternative is to add more constraints to what the model can do. We follow this second strategy in the next experiment in which we force the output of the model to contain certain notes.

8.2.4. Last Modeling Experiment: Enforcing Note-Level Constraints

Analyzing the results from the previous experiments, one observation stands out: the outcomes of the orchestration models resemble unconstrained generation, with minimal connection to the layer score provided as input. Ideally, an orchestration should preserve the essential elements of the music presented in the layer score, with only minor adjustments made to adapt the writing to the specified instruments.
In this final experiment, we introduce note-level constraints to the generated sequence. We maintain the same pretraining and finetuning methods used in the previous experiment (Section 8.2.3) and we modify the sampling strategy by enforcing specific tokens to appear in the output. Similar constraints were applied in prior experiments; for instance, in the second experiment (Section 8.2.2), we forced texture tokens corresponding to a specified orchestration plan to appear in the sequence after every NT token. In this iteration, however, we take this approach further by enforcing specific positions and notes in the output, following a method similar to that used by Le et al. in METEOR to ensure melody fidelity [150]. We focus on the task of texture-constrained orchestration from layer score (T7).

8.2.4.1. Constrained Sampling Schemes

We introduce two different sampling schemes to force notes from the layer score to appear in the orchestral score: weakly constrained and strongly constrained mode. First of all, in both modes, a provided orchestration plan determines the association between every instrument and the layer it should play. When the output sequence is generated autoregressively, we know to which track the generated tokens belong, and which instrument should be playing that track. The token enforcement process works by stopping the sampling as soon as a specific token is generated, and by inserting one or more token(s) into the output sequence just after, and by passing the added tokens through the model. Once this is done, the sampling resumes, using the entire sequence – including the enforced tokens – as the new prompt for further generation. The difference between the two modes is given by which tokens trigger the sampling break and by what is added to the output sequence.
In weakly constrained mode we insert texture tokens after every NT, describing the role and relationship of the layer to which the track belongs to. We follow the numerical order for the tracks (track 2 after track 1, and so on). After each position token, we pause sampling again, we insert any corresponding note tokens from the appropriate layer score track, if applicable, and then resume sampling. The model is free to sample positions that are not in the layer score, it can skip positions that instead are in the layer score, and it can add additional notes to the positions existing also in the layer score. It can also generate more or fewer tracks than those present in the orchestration plan.
In strongly constrained mode we stop the generation after every NT and we add all the tokens for the corresponding track, including texture tokens, positions and notes. This model is then an almost deterministic algorithm that copies the notes from the layer score into the orchestral score following the orchestration plan. The model is left with the freedom of choosing the number of tracks to generate, which can differ from the number of tracks in the orchestration plan. The model can also generate additional positions and notes in the existing tracks (since nothing forces them to appear in chronological order in the generated sequence).

8.2.4.2. Qualitative Evaluation

We test both orchestration modes by asking the model to re-orchestrate the beginning of the first movement of Haydn symphony N. 99, starting from the layer score and the orchestration plan, and by evaluating how close the results are to the original orchestral score. The prompt is structured like in Figure 8.4c: we provide four measures of layer score (Figure 8.5a), four measures of orchestration plan (using as plan the texture analysis of Haydn's reference score), and the first two measures of the reference orchestral score (Figure 8.5b). We ask the model to generate the third and fourth measure of the orchestral score, and we qualitatively evaluate its resemblance to the reference orchestral score by Haydn.
We report two examples of outputs generated in weakly (Figure 8.6) and strongly (Figure 8.7) constrained modes. In the first case we observe still a lot of freedom in what is generated, even though we recognize some patterns. The main melody is forced

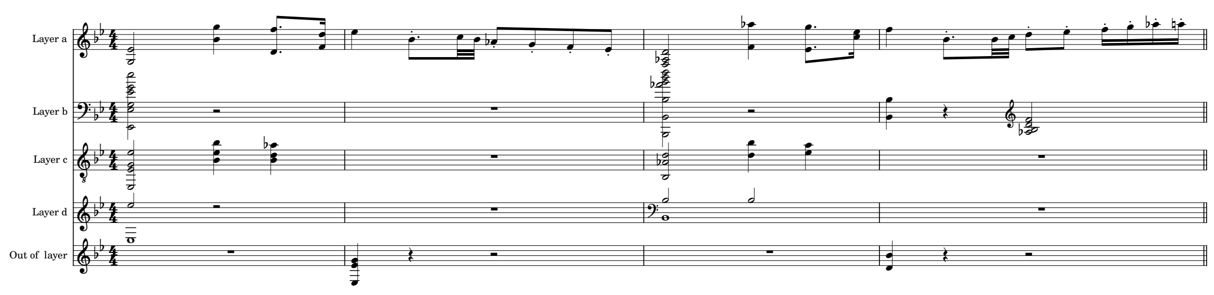
to appear in the violin 1 part, but not all positions are generated, meaning that some notes are skipped. At the same time, we have more notes on the vertical dimension: most of them are directly copied from the layer score, but some (like the high D on the first beat of the third measure, which we consider a mistake) are generated by the Transformer. The instruments chosen by the model, also not always correspond to what is provided as input (there are, for example, three bassoons – FAGOTT). In the second case, the model almost acts like a procedural algorithm, everything is copied and pasted from the layer score to the output. Chords in the layer score are copied in every part, instead of being split between the instruments.
Overall, it appears that the model is not mature enough. However, the approach of using a hybrid Transformer model with deterministic constraints seems very promising for future experiments.

8.3. Conclusions and Perspectives

In this chapter, we proposed an extended tokenization method to represent music, parts, and texture data in orchestral music within the same token sequence. This approach allows to use the same tokens to represent different inputs and outputs for different orchestration tasks (see Table 4.1).
We presented four experiments using Transformers to implement tasks in our framework of co-creative interaction in orchestration. Initial results have been promising, with the extended tokenization approach generating outputs that are closer to the classical-romantic style (in task U1).
However, orchestration models are still in their *early stages of development*. Our choice of using a Transformer model instead of a deterministic algorithm was motivated by the expectation that Transformers could provide enough flexibility to adapt the parts in the layer score to the idiomatic patterns of different instruments. The examples so far have shown that these model have too much creative freedom, making them difficult to control and constrain. In other words, these orchestration models are improvising too much. A promising solution appears to be in the integration of note-level constraints, creating hybrid models that combine Transformers with deterministic procedural methods. However, significant improvements and extended evaluations are still necessary.
Other model architectures should be tested, such as a full encoder-decoder Transformer for sequence-to-sequence translation between layer scores and orchestral scores. The scarcity of data for finetuning remains a challenge, which could be addressed either through automated layer annotation methods (Chapter 6) or by expanding the orchestration corpus with new manual annotations (Chapter 5).
Finally, *evaluation* for such co-creative tasks is challenging, as the results cannot be simply graded for their resemblance to the ground truth. Multiple "correct" orchestrations for the same piece might exist beyond the version by the original composer. Additionally, in co-creativity, the value of a model stands also in what it can bring to the human interacting with it. Quantifying the level of creativity of a model and its usefulness in an artistic creation process is challenging. Some of these things can be measured

indirectly with user studies. In this chapter, we only reported a few experiments. The next step in evaluation could then be the generation of several examples to be rated by a pool of experts.
Looking forward, further research in the field of orchestration and MIR could explore how these methods can be refined to develop more advanced and sophisticated models, address the challenges related to the evaluation of their outputs, and consider their integration into co-creative systems.

(a) Layer score



(b) Reference orchestral score

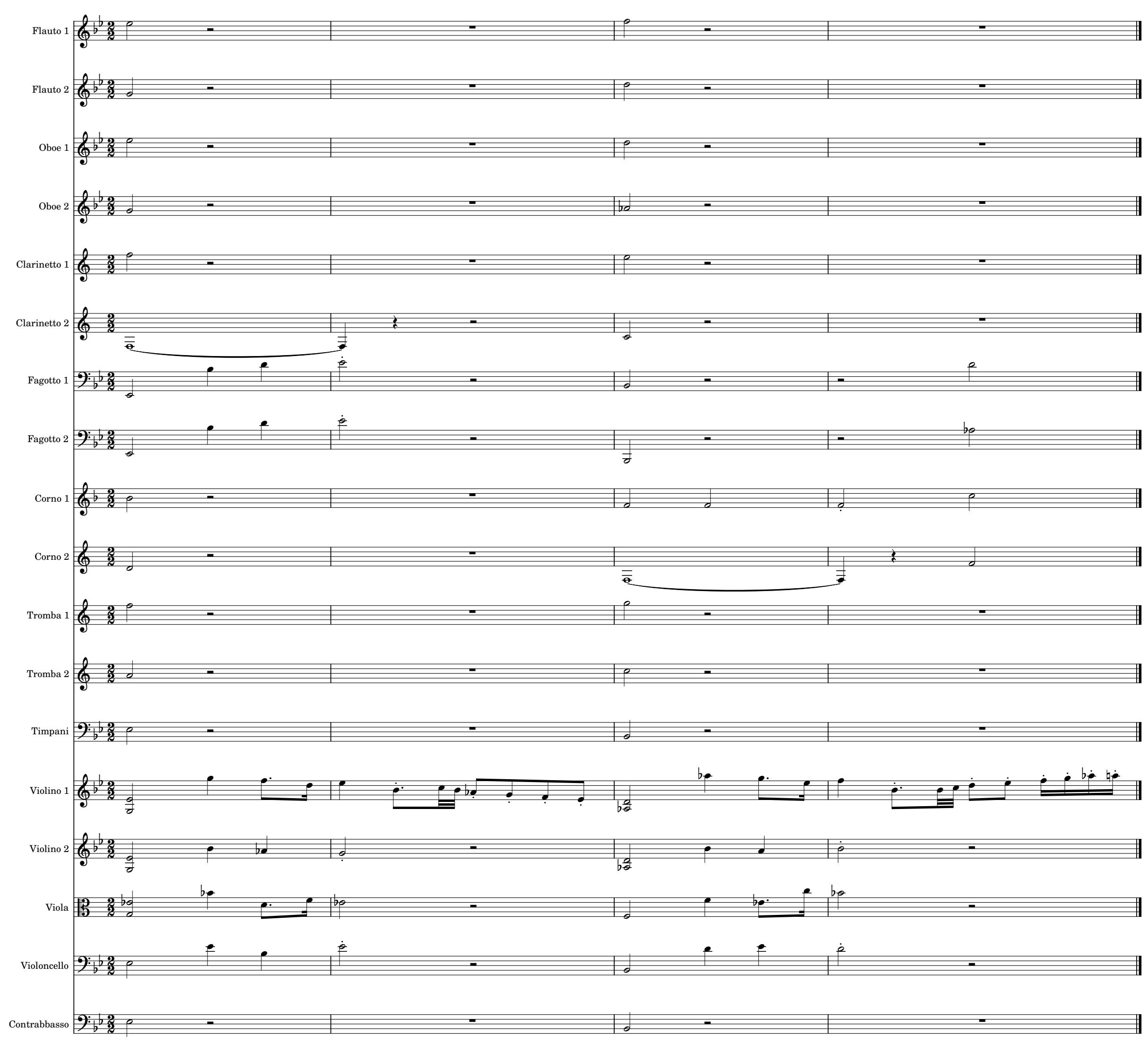


Figure 8.5.: First four measures from Symphony No. 99 by Haydn: layer score and reference orchestral score used for the note-level constrained orchestration experiment. The prompt contains the four measures of the layer score, and the first two measures of the reference orchestral score.
`http://algomus.fr/fm/haydn99-target.mp3`

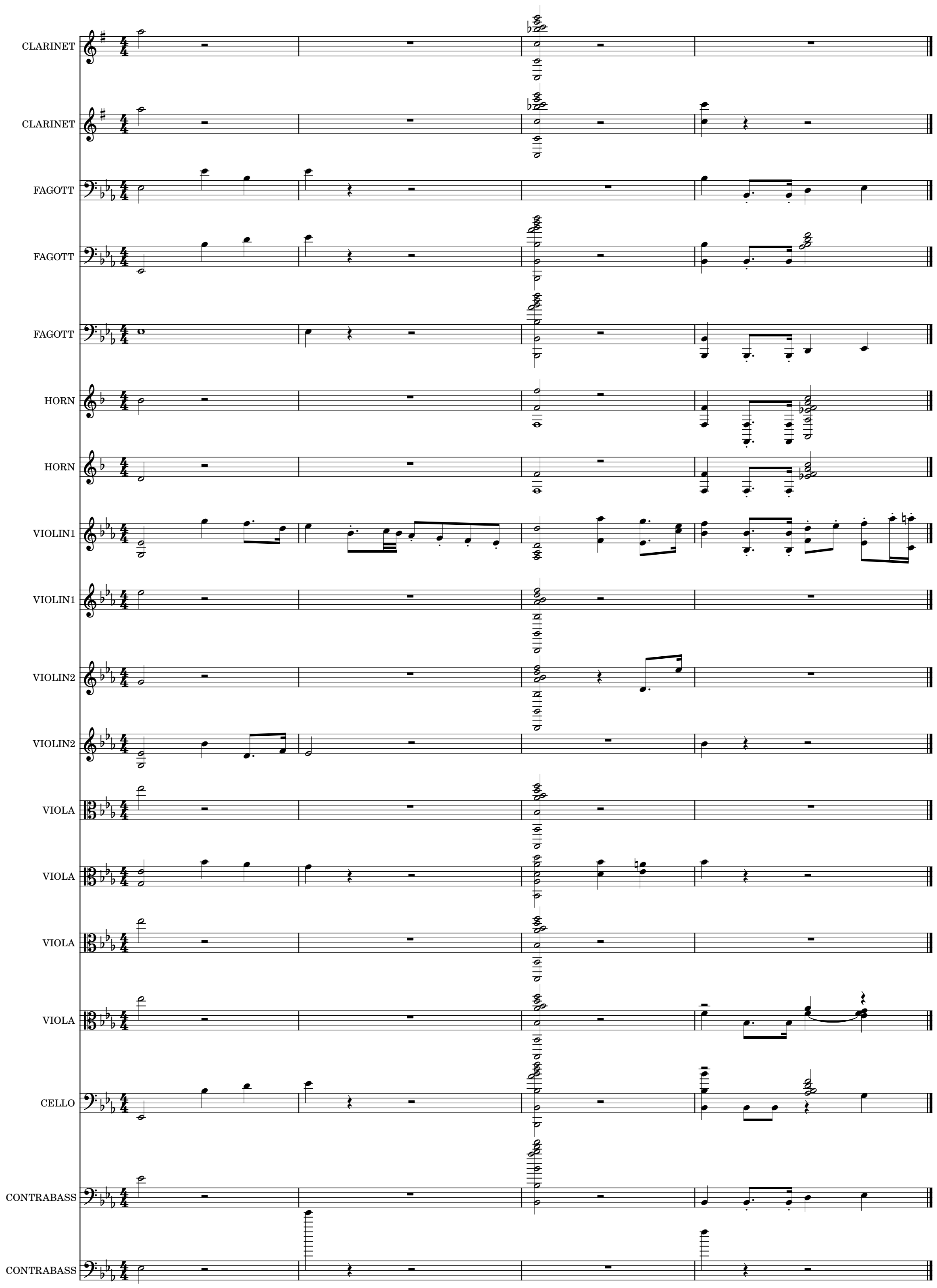


Figure 8.6.: Example of generation with the note-level constrained orchestration experiment, in weakly constrained mode. The first two measures are from the reference orchestration prompt, the third and fourth are the result of the orchestration algorithm.
http://algomus.fr/fm/haydn99-weak-constrained.mp3

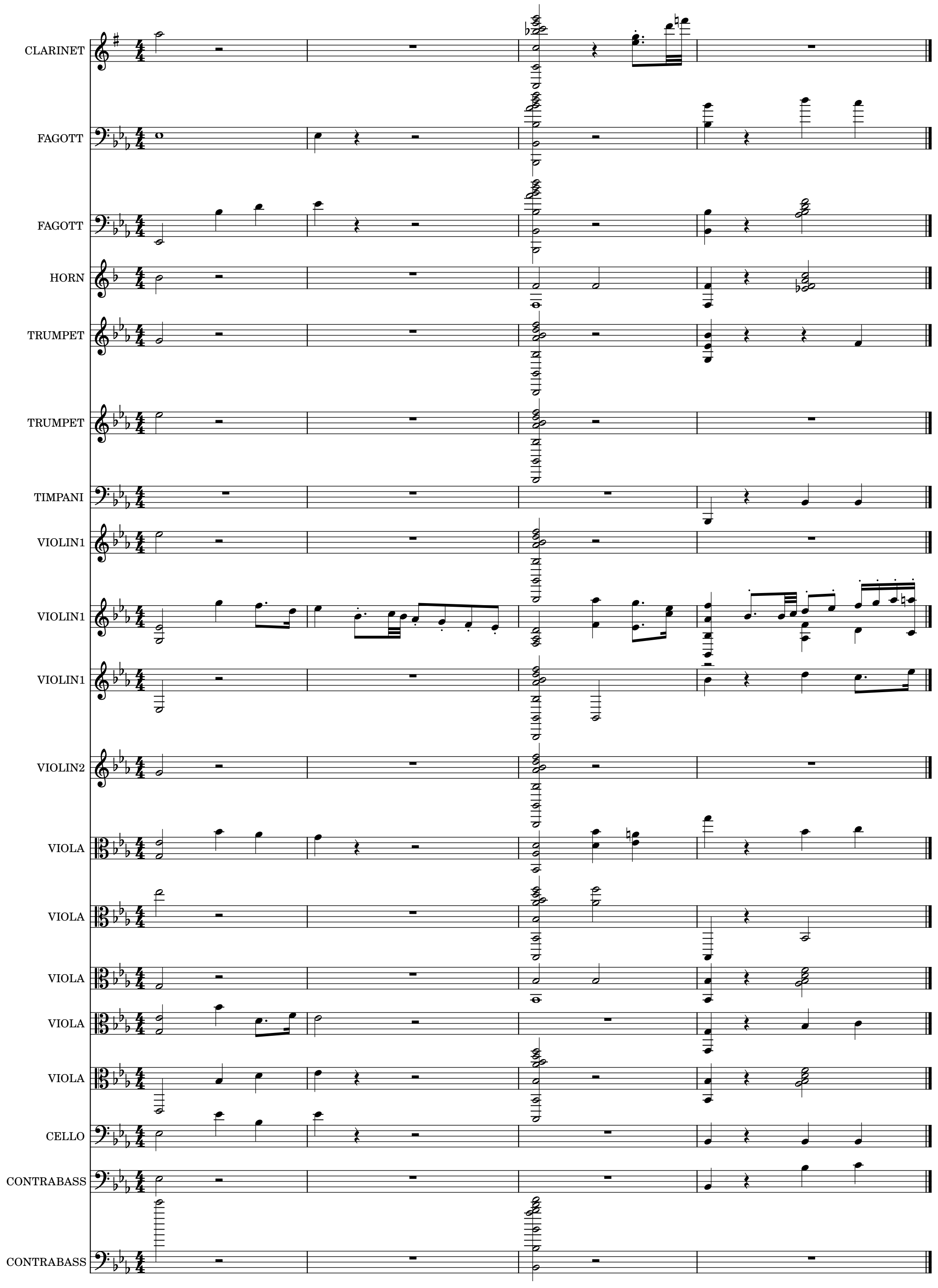


Figure 8.7.: Example of generation with the note-level constrained orchestration experiment, in strongly constrained mode. The first two measures are from the reference orchestration prompt, the third and fourth are the result of the orchestration algorithm.
`http://algomus.fr/fm/haydn99-strong-constrained.mp3`

9. Conclusions

In this thesis we proposed a new way of thinking and modeling orchestration, with a focus on the classical style, and we have shown its value in computer-assisted orchestration and orchestration analysis.

List of Publications

[149] Dinh-Viet-Toan Le, Mathieu Giraud, Florence Levé, and **Francesco Maccarini**, A corpus describing orchestral texture in first movements of classical and early-romantic symphonies. In *Digital Libraries for Musicology (DLfM 2022)*, pages 22–35, 2022.

[165] **Francesco Maccarini**, Mael Oudin, Mathieu Giraud, and Florence Levé, Co-creative orchestration of *Angeles* with layer scores and orchestration plans. In Colin Johnson, Sérgio M. Rebelo, and Iria Santos, editors, *Artificial Intelligence in Music, Sound, Art and Design*, pages 228–245, Cham, 2024. Springer Nature Switzerland.

[14] Charles Ballester, Baptiste Bacot, Louis Bigo, Vanessa Nina Borsan, Louis Couturier, Ken Déguernel, Quentin Dinel, Laurent Feisthauer, Klaus Frieler, Mark Gotham, Richard Groult, Johannes Hentschel, Alexandre d'Hooge, Dinh-Viet-Toan Le, Florence Levé, **Francesco Maccarini**, Ivana Maričić, Gianluca Micchi, Meinard Müller, Alexandros Stamatiadis, Tom Taffin, Patrice Thibaud, Christof Weiß, Emmanuel Leguy, and Mathieu Giraud. Interacting with annotated and synchronized music corpora on the Dezrann web platform. Submitted to Transactions of the International Society for Music Information Retrieval, August 2024

9.1. Summary

After the introduction (Chapter 1), we started by surveying the music theory, music perception, and MIR literature related to orchestral music and orchestration (Chapter 2).
In Chapter 3 we introduced and formalized three abstract models of orchestration. The first one is a taxonomy through which we can describe orchestral texture in classical symphonies as being formed by layers with roles and relations. The second one is the layer score, which is an abstract version of an orchestral composition, reduced to its essential elements in terms of layers. Information about the notes in different layers and about the layer roles is preserved, but no information on the instrumentation is

given. The third model is the orchestration plan, an object through which the instrumentation of the layers in a composition can be prescribed. These three models are derived from well established concepts from current orchestration teaching practices and music theory and perception literature. The main contribution from this chapter is the formalization of these concepts into the three objects, allowing to isolate some orchestration processes and choices.
In Chapter 4 we presented a framework for the co-creative interaction of human and algorithmic agents around orchestration. Two main contributions are presented. First, we organized and reclassified existing MIR tasks related to orchestration by their inputs and outputs, facilitating the identification of new tasks and challenges, in particular those involving a layer score and/or an orchestration plan. Second, we formalized our process for computer-assisted orchestration in three steps, allowing to orchestrate an existing piece for piano solo: (1) constructing a layer score from the original piano score, (2) writing an orchestration plan with the instrumentation for each layer, and (3) combining the information from the layer score and the orchestration plan to write an orchestral score. The orchestration tasks at every step are clearly identified and the process can be implemented in different ways, assigning the tasks either to humans or to algorithms.
In Chapter 5 we presented a corpus, released with open data licenses, containing first movements of symphonies in the western classical style, composed by Mozart, Haydn, and Beethoven. The corpus contains scores synchronized to public domain recordings, and analyzed with the models from Chapter 3. The corpus have been continuously revised and updated with new material through the duration of this thesis. The main contribution to science from this chapter is the corpus itself, that can be (and has already been) used for various purposes by us and by other scientists.
Chapter 6 presented a method based on statistical models and handcrafted features to detect orchestral blends in scores. Even though the results are still very preliminary, we were able to show that a data driven approach can match the performance of the state of the art model, that is a dedicated algorithm. This opens the way to further research opportunities.
In the last part of the thesis, Chapter 7 presented a research project on human-machine co-creativity, in which we have applied the framework presented in Chapter 4 to create orchestrations of two pieces for piano solo from the suite *Angeles* by Gissel Velarde. In the process we have generated orchestration plans with a customized probabilistic Markov model, while human orchestrators have taken care of constructing the layer scores, selecting the orchestration plans, and writing the final orchestral score. This project provided both scientific and artistic contributions. It is the first real use case experiment with our framework for co-creative interaction in orchestration, we have implemented the first model to generate orchestration plans, and we have collected feedback from the human artists involved. We also produced scores of the orchestrations of the two pieces, that have been performed live in a concert.
In Chapter 8, we looked at a series of experiments trying to solve orchestration tasks using deep learning models with the Transformer architecture. The preliminary results of these experiments are encouraging but not conclusive, suggesting that additional

research is necessary to develop and train a fully operational Transformer model for orchestration. The proposed extended tokenization method is the main contribution in this chapter. It allows to represent orchestral scores, texture annotations, layer scores, and orchestration plans.

9.2. Future Work

The research described in this thesis offers many interesting perspectives for future work in different directions that we organize here in six main themes.

High Level Modeling of Orchestration The need for high level models of orchestration has been addressed in Chapters 3 and 4. The structured framework for co-creative orchestration would certainly benefit from new abstract intermediate objects. For the *Angeles* project (Chapter 7), for example, we have defined a target loudness profile to control the generation of orchestration plans. This and other control parameters could be precisely formalized and introduced in the framework, together with the layer score and the orchestration plan. For example, we believe that formally modeling target perceptions is a necessity for future iterations of our framework. This idea relates very closely to some of the relatively unexplored MIR tasks presented in Table 4.1. In the table we see that there exist still very little studies on task of perception-controlled orchestration (C6) and that they are quite disconnected to the "traditional" orchestration from piano task (T1).
We mention also two other under-explored tasks that are relevant to the improvement of the framework. The first one is the modification of a layer score (S4). In the *Angeles* project we have used handmade refinements of the layer score by the human orchestrators, but the MIR task also appears interesting to study. The second one is texture-constrained reorchestration (S2), which is similar to texture-constrained orchestration, but it starts from an existing orchestral score instead of a piano score.

Synergies with the TOGE and Other Orchestration Analysis Taxonomies In Section 3.1 we have given a formal definition of orchestral layers and we have described grouping rules to annotate texture-based layers in scores. As discussed in Section 3.1.5, the same formal definition can be used for orchestral blends from OrchARD, even though the perception-based grouping rules from the TOGE [178] that are used to annotate them are slightly different. In the majority of the thesis, and in particular to annotate the corpus in Chapter 5, we have adopted the texture-based approach from our taxonomy to identify layers, with the exception of Chapter 6, in which we have studied orchestral blends from OrchARD. The differences between these two taxonomies are still to be addressed in future work. A research in that direction would definitely prove beneficial also to the research theme of perception-controlled orchestration. A first study could compare two different analysis of the same piece. Other taxonomies in ACTOR have followed this approach with a comparison of different analyses of Ravel's *Alborada del Gracioso* [277].

Co-creativity in Orchestration One of the main theme of research in this thesis was co-creativity in orchestration. An effort was made to develop tools to allow the collaboration between human artists and computers in the creation of orchestration (Chapter 8) and to run a first experiment of co-creativity in collaboration with Mael Oudin and Gissel Velarde (Chapter 7). The goal of having fully functioning models ready to be used out of the box is still very far, and further research is required. Research perspectives include, but are not limited to, the improvement of the Transformer models. The problem could benefit from a more targeted pretraining strategy and from a larger finetuning dataset. Certain details that have been left behind could be implemented, in particular what concerns the representation of music dynamics, which have not been included so far. A more drastic change would be to move from a one-sequence language model to a sequence-to-sequence translation model (layer score to orchestral score). Some efforts have been made in this direction, but with no results so far. Another radical idea consists in completely modifying the music representation in order to explicitly model the dependency of the orchestral score from the layer score and the orchestration plan with transformation rules.
The need for large scale orchestral thinking and anticipation emerged from the *Angeles* project. Future models for orchestration plans should try to better model the relationships between consecutive segments, for example allowing to "reserve" instruments for a specific climax in the piece.
Finally, once more stable tools exists, a more controlled study on co-creativity could be envisioned, addressing also agency and the perception of the AI model by human artists. Another study could also compare how different artists approach co-creative orchestration with the same tools.

Orchestration Analysis Large scale computational analysis of orchestral texture has also been very little explored in this thesis. An effort in that direction would be beneficial also to other research themes, as it could help produce larger annotated datasets for texture-controlled generation. One research direction is that of finetuning large language models like the Transformers from Chapter 8 for a texture analysis task. Another possible approach consists in adapting the blend detection algorithm from Chapter 6 for the identification of texture layers in the symphonies' corpus. These two models are in a preliminary stage of development with further research required.

Corpus Expansion The expansion of the corpus is the "fil rouge" that connects all the research themes presented so far. The interaction between corpus and research goes in both directions. New research can lead to the expansion of the corpus with new objects, new analysis, and additional pieces, and the expansion of the corpus can provide new data to train Machine Learning models for analysis and generation tasks.
The corpus can be extended horizontally by annotating new pieces, synchronizing public domain recordings, and creating layer scores. To expand the reference analysis on the classical style the corpus could be enlarged by including the other movements of the symphonies already present. Alternatively, similar studies could extend and use measure-level annotations to romantic works such as symphonies by Berlioz or Mahler

or to 20th-century music. Such orchestral music will generally have an increased complexity, with more diverse instruments, playing techniques, and textural effects. The concept of orchestral layer will probably also be challenged, requiring an extension of the theory to modern writing styles. A comparison of our taxonomy with other perception-based taxonomies would be very beneficial also to this kind of study. It will also be more challenging to find encodings of coherent corpora with full orchestral scores.
Moreover, the corpus can be extended vertically, by adding material of different kinds including new abstract models of orchestration, perception annotations, or other complementary analysis following alternative taxonomies.

A Theory of Texture Finally, in a broader perspective, we have made several steps towards the formalization of orchestral texture. Future work should continue advancing in this direction, aiming at a more general theory of texture, capable of describing music in different styles and for a wider range of ensembles (for example piano solo, string quartet, and classical orchestra). While the language that we developed for orchestral texture annotations works quite well for describing western classical style orchestration, it might not be suitable for later compositions, and it cannot be applied as it is to music for different ensembles. The introduction of the layer score as an intermediate (or a common ancestor) between the orchestral score and the piano reduction, is a promising starting point for expanding the theory.

Bibliography

[1] Samer Abdallah, Nicolas Gold, and Alan Marsden. Analysing symbolic music with probabilistic grammars. In Meredith [184], pages 157–189.

[2] José Abreu, Marcelo Caetano, and Rui Penha. Computer-aided musical orchestration using an artificial immune system. In *International Conference on Evolutionary and Biologically Inspired Music, Sound, Art and Design (EvoMUSART 2016)*, page 1–16, 2016.

[3] Sara Adkins, Pedro Sarmento, and Mathieu Barthet. LooperGP: A loopable sequence model for live coding performance using guitarpro tablature. In *Artificial Intelligence in Music, Sound, Art and Design*, Lecture Notes in Computer Science, pages 3–19, 2023.

[4] Samuel Adler. *The Study of Orchestration*. Norton, 1982 (first ed.).

[5] Andrea Agostinelli, Timo I. Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, Matt Sharifi, Neil Zeghidour, and Christian Frank. MusicLM: Generating music from text, 2023.

[6] Kat Agres, Jamie Forth, and Geraint A Wiggins. Evaluation of musical creativity and musical metacreation systems. *Computers in Entertainment (CIE)*, 14(3):1–33, 2016.

[7] Yun-Kang Ahn, Carlos Agon, and Moreno Andreatta. Structures Ia pour deux pianos by Boulez: Towards creative analysis using OpenMusic and Rubato. In Timour Klouche and Thomas Noll, editors, *Mathematics and Computation in Music*, pages 412–418, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[8] Christina Anagnostopoulou and Chantal Buteau. Can computational music analysis be both musical and computational? *journal of Mathematics and Music*, 4(2):75–83, 2010.

[9] Torsten Anders. Compositions Created with Constraint Programming. In *The Oxford Handbook of Algorithmic Music*. Oxford University Press, 02 2018.

[10] Dimitris Andrikopoulos and Rui Dias. Chamber music in the xxi century: Problematic and challenges, 03 2016.

[11] Aurélien Antoine, Philippe Depalle, Philippe Macnab-Séguin, and Stephen McAdams. Modeling human experts' identification of orchestral blends using symbolic information. In Richard Kronland-Martinet, Sølvi Ystad, and Mitsuko Aramaki, editors, *Perception, Representations, Image, Sound, Music*, pages 363–378, Cham, 2021. Springer International Publishing.

[12] Aurelien Antoine and Eduardo Miranda. Predicting timbral and perceptual characteristics of orchestral instrument combinations. *The Journal of the Acoustical Society of America*, 143(3_Supplement):1747–1747, 03 2018.

[13] Gérard Assayag. Creative symbolic interaction. In *Sound and Music Computing Conference (SMC 2014)*, pages pp–1, 2014.

[14] Charles Ballester, Baptiste Bacot, Louis Bigo, Vanessa Nina Borsan, Louis Couturier, Ken Déguernel, Quentin Dinel, Laurent Feisthauer, Klaus Frieler, Mark Gotham, Richard Groult, Johannes Hentschel, Alexandre d'Hooge, Dinh-Viet-Toan Le, Florence Levé, Francesco Maccarini, Ivana Maričić, Gianluca Micchi, Meinard Müller, Alexandros Stamatiadis, Tom Taffin, Patrice Thibaud, Christof Weiß, Emmanuel Leguy, and Mathieu Giraud. Interacting with annotated and synchronized music corpora on the Dezrann web platform. Submitted to Transactions of the International Society for Music Information Retrieval, 2024.

[15] Alan Belkin. *Artistic Orchestration*. A. Belkin, 2001.

[16] Oded Ben-Tal, Matthew Tobias Harris, and Bob LT Sturm. How music AI is useful: Engagements with composers, performers and audiences. *Leonardo*, 54(5):510–516, 2021.

[17] Ian D. Bent and Anthony Pople. Analysis. In *Grove Music Online*. Oxford University Press, 2001.

[18] Bruce Benward and Marilyn Saker. *Music in Theory and Practice (Volume 1)*. McGraw-Hill Professional, 2008 (8th ed.).

[19] Hector Berlioz. *Grand Traité d'instrumentation et d'orchestration Modernes*. Novello, 1844 (first ed.).

[20] Amélie Bernier-Robert. Spectral envelope. Timbre Lingo - Timbre and Orchestration Writings, 2023. `https://timbreandorchestration.org/writings/timbre-lingo/spectral-envelope/`.

[21] Amélie Bernier-Robert and Ben Duinker. Blend. Timbre Lingo - Timbre and Orchestration Writings, 2023. `https://timbreandorchestration.org/writings/timbre-lingo/blend/`.

[22] Vanessa Nina Borsan, Mathieu Giraud, and Richard Groult. The Games We Play: Exploring The Impact of ISMIR on Musicology. In *International Society for Music Information Retrieval Conference (ISMIR 2023)*, pages 474–481, Milano, Italy, November 2023.

[23] Juan J Bosch, Ricard Marxer, and Emilia Gómez. Evaluation and combination of pitch estimation methods for melody extraction in symphonic classical music. *Journal of New Music Research*, 45(2):101–117, 2016.

[24] Albert S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. The MIT Press, 1990.

[25] Jean-Pierre Briot, Gaëtan Hadjeres, and François-David Pachet. *Deep learning techniques for music generation*. Springer, 2019.

[26] Jean-Pierre Briot and François Pachet. Deep learning for music generation: Challenges and directions. *Neural Computing and Applications*, 32(4):981–993, 2020-02-01.

[27] Gino Brunner, Andres Konrad, Yuyi Wang, and Roger Wattenhofer. MIDI-VAE: Modeling Dynamics and Instrumentation of Music with Applications to Style Transfer, 2018-09-20.

[28] Morgan Buisson, Brian Mcfee, Slim Essid, and Helene-Camille Crayencour. Learning Multi-Level Representations for Hierarchical Music Structure Analysis. In *International Society for Music Information Retrieval (ISMIR)*, Bengaluru, India, December 2022.

[29] Marcelo Caetano and Carmine E. Cella. Imitative computer-aided musical orchestration with biologically inspired algorithms. In Eduardo Reck Miranda, editor, *Handbook of Artificial Intelligence for Music: Foundations, Advanced Approaches, and Developments for Creativity*, pages 585–615. Springer International Publishing, Cham, 2021.

[30] Marcelo Caetano, Asterios Zacharakis, Isabel Barbancho, and Lorenzo J. Tardón. Leveraging diversity in computer-aided musical orchestration with an artificial immune system for multi-modal optimization. *Swarm and Evolutionary Computation*, 50:100484, 2019.

[31] Emilios Cambouropoulos. Voice and stream: Perceptual and computational modeling of voice separation. *Music Perception*, 26(1):75–94, 09 2008.

[32] Grégoire Carpentier, Gérard Assayag, and Emmanuel Saint-James. Solving the musical orchestration problem using multiobjective constrained optimization with a genetic local search approach. *Journal of Heuristics*, 16:681–714, 2010.

[33] Grégoire Carpentier, Damien Tardieu, Gérard Assayag, Xavier Rodet, and Emmanuel Saint-James. Imitative and generative orchestrations using pre-analysed sounds databases. In *SMC'06*, pages 115–122, 2006.

[34] Luca Casini, Nicolas Jonason, and Bob L. T. Sturm. Investigating the viability of masked language modeling for symbolic music generation in abc-notation. In Colin Johnson, Sérgio M. Rebelo, and Iria Santos, editors, *Artificial Intelligence in*

Music, Sound, Art and Design, pages 84–96, Cham, 2024. Springer Nature Switzerland.

[35] Norman Cazden. The definition of consonance and dissonance. *International Review of the Aesthetics and Sociology of Music*, 11(2):123–168, 1980.

[36] Carmine-Emanuele Cella. Orchidea: A comprehensive framework for target-based computer-assisted dynamic orchestration. *Journal of New Music Research*, 2022.

[37] Carmine-Emanuele Cella, Daniele Ghisi, Yan Maresz, Alessandro Petrolati, Alexandre Teiller, and Philippe Esling. Dynamic computer-aided orchestration in practice with orchidea. *Computer Music Journal*, 45(4):40–56, 2023.

[38] Elaine Chew and Xiaodan Wu. Separating voices in polyphonic music: A contig mapping approach. In *International Symposium on Computer Music Modeling and Retrieval (CMMR 2005)*, pages 1–20, 2005.

[39] Oscar Olin Chism. Sonata form in the symphonies of Mozart. Master's thesis, North Texas State University, 1964.

[40] Kristy Choi, Curtis Hawthorne, Ian Simon, Monica Dinculescu, and Jesse Engel. Encoding musical style with transformer autoencoders. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1899–1908. PMLR, 13–18 Jul 2020.

[41] Yi-Hui Chou, I.-Chun Chen, Chin-Jui Chang, Joann Ching, and Yi-Hsuan Yang. MidiBERT-Piano: Large-scale Pre-training for Symbolic Music Understanding, 2021.

[42] Ya-Hsuan Chu and Li Su. Orchestral Texture Classification with Convolution. In *Extended Abstracts for the Late-Breaking Demo Session of the 24th Int.Society for Music Information Retrieval Conf.*, 2023.

[43] Nicholas Cook. Towards the compleat musicologist. In *International Conference on Music Information Retrieval (ISMIR 2005)*, 2005.

[44] David Cope. Experiments in musical intelligence (EMI): Non-linear linguistic-based composition. *Interface*, 18(1-2):117–139, 1989.

[45] David Cope. *Experiments in musical intelligence*, volume 12. AR editions Madison, WI, 1996.

[46] Bas Cornelissen, Willem Zuidema, and John Ashley Burgoyne. Cosine contours: a multipurpose representation for melodies. In *International Society for Music Information Retrieval Conference (ISMIR 2021)*, pages 135–142, 2021.

[47] Louis Couturier, Louis Bigo, and Florence Levé. A dataset of texture annotations in Mozart piano sonatas. In *International Society for Music Information Retrieval Conference (ISMIR 2022)*, 2022.

[48] Louis Couturier, Louis Bigo, and Florence Levé. Comparing texture in piano scores. In *International Society for Music Information Retrieval Conference (ISMIR 2023)*, 2023.

[49] Louis Couturier, Louis Bigo, and Florence Levé. Annotating symbolic texture in piano music: A formal syntax. In *Sound and Music Computing Conference (SMC 2022)*, 2022.

[50] Léopold Crestel. *Neural networks for automatic musical projective orchestration*. Theses, Sorbonne Université, December 2018.

[51] Léopold Crestel and Philippe Esling. Live Orchestral Piano, a system for real-time orchestral music generation. In *Sound and Music Computing Conference (SMC 2017)*, page 434, 2017.

[52] Léopold Crestel, Philippe Esling, Lena Heng, and Stephen McAdams. A database linking piano and orchestral MIDI scores with application to automatic projective orchestration. In *International Society for Music Information Retrieval Conference (ISMIR 2017)*, 2017.

[53] David F. Crouse. On implementing 2D rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696, 2016.

[54] Michael Scott Cuthbert and Christopher Ariza. music21: A toolkit for computer-aided musicology and symbolic music data. In *International Society for Music Information Retrieval Conference (ISMIR 2010)*, pages 637–642, 2010.

[55] Shuqi Dai, Huan Zhang, and Roger B. Dannenberg. The interconnections of music structure, harmony, melody, rhythm, and predictivity. *Music & Science*, 7:20592043241234758, 2024.

[56] Corinne Darche. Analysis of timbral augmentation in the Orchestration Analysis & Research Database. Master's thesis, McGill University, 2024.

[57] Martha de Francisco, Malte Kob, Jack Kelly, Diego Quiroz, and Stefanos Ioannou. Odessa: An interdisciplinary symphonic recording for the study of orchestral sound blending. In *Audio Engineering Society Convention 149*. Audio Engineering Society, 2020.

[58] Daniel Moreira de Sousa. Composing with textures: A proposal for formalization of textural spaces. *Journal MusMat*, 3(1), June 2019.

[59] Daniel Moreira de Sousa. *Textural design: A Compositional Theory for the Organization of Musical Texture*. PhD thesis, Universidade Federal do Rio de Janeiro, 2019.

[60] Reinier de Valk and Tillman Weyde. Deep neural networks with voice entry estimation heuristics for voice separation in symbolic music representations. In *International Society for Music Information Retrieval Conference (ISMIR 2018)*, 2018-05-25.

[61] Ken Déguernel, Mathieu Giraud, Richard Groult, and Sebastien Gulluni. Personalizing AI for co-creative music composition from melody to structure. In *Sound and Music Computing (SMC 2022)*, Sound and Music Computing (SMC 2022), pages 314–321, 2022.

[62] Diana Deutsch. Grouping mechanisms in music. *The psychology of music*, 28:299–348, 1999.

[63] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

[64] Emily I Dolan. *The orchestral revolution: Haydn and the technologies of timbre*. Cambridge University Press, 2013.

[65] Chris Donahue, Huanru Henry Mao, Yiting Ethan Li, Garrison Cottrell, and Julian McAuley. LakhNES: Improving Multi-instrumental Music Generation with Cross-domain Pre-training. In *Proceedings of the 20th International Society for Music Information Retrieval Conference*, pages 685–692. ISMIR, November 2019.

[66] Hao-Wen Dong, Ke Chen, Shlomo Dubnov, Julian McAuley, and Taylor Berg-Kirkpatrick. Multitrack music transformer. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023.

[67] Hao-Wen Dong, Chris Donahue, Taylor Berg-Kirkpatrick, and Julian McAuley. Towards automatic instrumentation by learning to separate parts in symbolic multitrack music. In *International Society for Music Information Retrieval Conference (ISMIR 2021)*, 2021.

[68] Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi-Hsuan Yang. MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *AAAI Conference on Artificial Intelligence (AAAI 2018)*, volume 32, 2018.

[69] Hao-Wen Dong and Yi-Hsuan Yang. Convolutional Generative Adversarial Networks with Binary Neurons for Polyphonic Music Generation. In *International Society for Music Information Retrieval Conference (ISMIR 2018)*, pages 190–196. ISMIR, November 2018.

[70] J. Stephen Downie. Music information retrieval. *Annual review of information science and technology*, 37(1):295–340, 2003.

[71] Jonathan Dunsby. Considerations of texture. *Music & Letters*, 70(1):46–57, 1989.

[72] Taffeta M. Elliott, Liberty S. Hamilton, and Frédéric E. Theunissen. Acoustic structure of the five perceptual dimensions of timbre in orchestral instrument tones. *The Journal of the Acoustical Society of America*, 133(1):389–404, 01 2013.

[73] Jeff Ens and Philippe Pasquier. MMM: Exploring conditional multi-track music generation with the transformer, 2020.

[74] Raymond Erickson. Music analysis and the computer. *Journal of Music Theory*, 12(2):240–263, 1968.

[75] Philippe Esling, Grégoire Carpentier, and Carlos Agon. Dynamic musical orchestration using genetic algorithms and a spectro-temporal description of musical instruments. In *European Conference on the Applications of Evolutionary Computation (EvoCOP 2010)*, pages 371–380, 2010.

[76] Philippe Esling, Axel Chemla-Romeu-Santos, and Adrien Bitton. Generative timbre spaces with variational audio synthesis. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, pages 175–181, 2018.

[77] Philippe Esling and Ninon Devis. Creativity in the era of artificial intelligence. In *Journées d'Informatique Musicale*, Strasbourg, France, October 2020. Keynote paper - JIM Conference 2020 - 12 pages.

[78] Shimon Even. *Graph Algorithms*. Cambridge University Press, 2011.

[79] J. D. Fernández and F. Vico. AI methods in algorithmic composition: A comprehensive survey. *J. Artificial Intell. Res.*, 48(1):513–582, October 2013.

[80] José Miguel Fernandez, Thomas Köppel, Nina Verstraete, Grégoire Lorieux, Alexander Vert, and Philippe Spiesser. GeKiPe, a gesture-based interface for audiovisual performance. In *New Interfaces for Musical Expression Conference (NIME 2017)*, pages 450–455, 2017.

[81] Christoph Finkensiep and Martin Alois Rohrmeier. Modeling and inferring protovoice structure in free polyphony. In *International Society for Music Information Retrieval Conference (ISMIR 2021)*, pages 189–196, 2021.

[82] Cecil Forsyth. *Orchestration*. Courier Corporation, 1914 (first ed.), 1935.

[83] Nathan Fradet, Jean-Pierre Briot, Fabien Chhel, Amal El Fallah-Seghrouchni, and Nicolas Gutowski. MidiTok: A Python Package for MIDI File Tokenization. In *Extended Abstracts for the Late-Breaking Demo Session of the 22nd Int. Society for Music Information Retrieval Conf.*, Online, United States, November 2021.

[84] Nathan Fradet, Nicolas Gutowski, Fabien Chhel, and Jean-Pierre Briot. Impact of Time and Note Duration Tokenizations on Deep Learning Symbolic Music Modeling. In *International Society for Music Information Retrieval Conference (ISMIR 2023)*, pages 89–97. ISMIR, December 2023.

[85] Anders Friberg and Sven Ahlbäck. Recognition of the main melody in a polyphonic symbolic score using perceptual knowledge. *Journal of New Music Research*, 38(2):155–169, 2009.

[86] Philip Gage. A new algorithm for data compression. *The C Users Journal*, 12(2), 1994.

[87] Louis Garczynski, Mathieu Giraud, Emmanuel Leguy, and Philippe Rigaux. Modeling and editing cross-modal synchronization on a label web canvas. In *Music Encoding Conference (MEC 2022)*, 2022.

[88] Daniele Ghisi. *Music across music: towards a corpus-based, interactive computer-aided composition*. PhD thesis, Paris 6, 2017.

[89] Mathieu Giraud, Florence Levé, Florent Mercier, Marc Rigaudière, and Donatien Thorez. Towards modeling texture in symbolic data. In *International Society for Music Information Retrieval Conference (ISMIR 2014)*, pages 59–64, 2014.

[90] Michael Good. Musicxml for notation and analysis. *The virtual score: representation, retrieval, restoration*, 12:113–124, 2001.

[91] Meghan Goodchild and Stephen McAdams. Perceptual processes in orchestration. In Emily I. Dolan and Alexander Rehding, editors, *The Oxford Handbook of Timbre*. Oxford University Press, 2018.

[92] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[93] Mark Gotham, Johannes Hentschel, Louis Couturier, Nathan Dykeaylen, Martin Rohrmeier, and Mathieu Giraud. The "measure map": An inter-operable standard for aligning symbolic music. In *Digital Libraries for Musicology (DLfM 2023)*, page 91–99, 2023.

[94] Mark Gotham, Maureen Redbond, Bruno Bower, and Peter Jonas. The "openscore string quartet" corpus. In *Digital Libraries for Musicology (DLfM 2023)*, DLfM '23, page 49–57, New York, NY, USA, 2023. Association for Computing Machinery.

[95] Mark R. H. Gotham, Kunpeng Song, Nicolai Böhlefeld, and Ahmed Elgammal. Beethoven X: Es könnte sein! (It could be!). In *Conference on AI Music Creativity (AIMC 2022)*, 2022.

[96] Patrick Gray and Razvan Bunescu. A neural greedy model for voice separation in symbolic music. In *International Society for Music Information Retrieval Conference (ISMIR 2016)*, pages 782–788, 2016.

[97] Didier Guigue and Charles de Paiva Santana. The structural function of musical texture: Towards a computer-assisted analysis of orchestration. In *Journées d'Informatique Musicale (JIM 2018)*, 2018.

[98] Nicolas Guiomard-Kagan, Mathieu Giraud, Richard Groult, and Florence Levé. Comparing voice and stream segmentation algorithms. In *International Society for Music Information Retrieval Conference (ISMIR 2015)*, pages 493–499, 2015.

[99] Nicolas Guiomard-Kagan, Mathieu Giraud, Richard Groult, and Florence Levé. Improving voice separation by better connecting contigs. In *International Society for Music Information Retrieval Conference (ISMIR 2016)*, pages 192–198, 2016.

[100] Paul Waldersee Gustav Nottebohm, Carl Reinecke, editor. *Mozart Symphonies*. Leipzig: Breitkopf & Härtel, 1880.

[101] Gaëtan Hadjeres, François Pachet, and Frank Nielsen. Deepbach: a steerable model for bach chorales generation. In *International Conference on Machine Learning (ICML 2017)*, pages 1362–1371. PMLR, 2017.

[102] Gaëtan Hadjeres and Léopold Crestel. The Piano Inpainting Applicationmuse, 2021.

[103] Gaëtan Hadjeres, Jason Sakellariou, and François Pachet. Style imitation and chord invention in polyphonic music with exponential families, 2016.

[104] Masatoshi Hamanaka, Keiji Hirata, and Satoshi Tojo. Implementing methods for analysing music based on Lerdahl and Jackendoff's generative theory of tonal music. In Meredith [184], pages 221–249.

[105] Eliot Handelman, Andie Sigler, and David Donna. Automatic orchestration for automatic composition. In *Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE 2012)*, 2012.

[106] Stephen A. Hedges. Dice music in the eighteenth century. *Music & Letters*, 59(2):180–187, 1978.

[107] William James Henderson. *The orchestra and orchestral music*. Charles Scribner's sons, 1899.

[108] James Hepokoski and Warren Darcy. *Elements of Sonata Theory: Norms, Types, and Deformations in the Late-Eighteenth-Century Sonata*. Oxford University Press, 2006.

[109] Carlos Hernandez-Olivan and José R. Beltrán. Music composition with deep learning: A review. In Anupam Biswas, Emile Wennekes, Alicja Wieczorkowska, and Rabul Hussain Laskar, editors, *Advances in Speech and Music Technology: Computational Aspects and Applications*, pages 25–50. Springer International Publishing, Cham, 2023.

[110] Dorien Herremans and Elaine Chew. Morpheus: generating structured music with constrained patterns and tension. *IEEE Transactions on Affective Computing*, 2017.

[111] Dorien Herremans, Ching-Hua Chuan, and Elaine Chew. A functional taxonomy of music generation systems. *ACM Computing Surveys*, 50(5):69:1–69:30, 2017-09-26.

[112] Lejaren A. Hiller and Robert A. Baker. Computer cantata: A study in compositional method. *Perspectives of New Music*, 3(1):62–90, 1964.

[113] Lejaren A. Hiller and Leonard M. Isaacson. Musical composition with a high speed digital computer. In *Audio Engineering Society Convention 9*, 1957.

[114] Lejaren A. Hiller and Leonard M. Isaacson. *Experimental music: composition with an electronic computer*. McGraw-Hill, 1959.

[115] Yuki Hoshi, Ryohei Orihara, Yuichi Sei, Yasuyuki Tahara, and Akihiko Ohsuga. Versatile automatic piano reduction generation system by deep learning. In *International Conference on Advanced Research in Computing (ICARC)*, pages 66–71, 2022.

[116] Wen-Yi Hsiao, Jen-Yu Liu, Yin-Cheng Yeh, and Yi-Hsuan Yang. Compound Word Transformer: Learning to Compose Full-Song Music over Dynamic Directed Hypergraphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(1):178–186, 2021-05-18.

[117] Yo-Wei Hsiao and Li Su. Learning note-to-note affinity for voice segregation and melody line identification of symbolic music data. In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, pages 285–292. ISMIR, October 2021.

[118] Cheng-Zhi Anna Huang, Hendrik Vincent Koops, Ed Newton-Rex, Monica Dinculescu, and Carrie J. Cai. AI song contest: Human-AI co-creation in songwriting. In *International Society for Music Information Retrieval Conference (ISMIR 2020)*, 2020.

[119] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer. arXiv preprint arXiv:1809.04281, 2018.

[120] Jiun-Long Huang, Shih-Chuan Chiu, and Man-Kwan Shan. Towards an automatic music arrangement framework using score reduction. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 8(1):8:1–8:23, 2012.

[121] Yu-Siang Huang and Yi-Hsuan Yang. Pop music transformer: Beat-based modeling and generation of expressive pop piano compositions. In *ACM International Conference on Multimedia (MM 2020)*, pages 1180–1188, 2020-10-12.

[122] David Huron. Characterizing musical textures. In *International Computer Music Conference (ICMC 1989)*, pages 131–134, 1989.

[123] David Huron. Humdrum and kern: Selective feature encoding. In Eleanor Selfridge-Field, editor, *Beyond MIDI: the handbook of musical codes*, pages 375–401. MIT Press, 1997.

[124] David Huron. Tone and voice: A derivation of the rules of voice-leading from perceptual principles. *Music Perception*, 19(1):1–64, 2001.

[125] Natasha Jaques, Shixiang Gu, Richard E Turner, and Douglas Eck. Tuning recurrent neural networks with reinforcement learning. (Workshop track - ICLR 2017), 2017.

[126] Shulei Ji, Jing Luo, and Xinyu Yang. A comprehensive survey on deep music generation: Multi-level representations, algorithms, evaluations, and future directions. arXiv:2011.06801, 2020.

[127] Zheng Jiang and Roger B Dannenberg. Melody identification in standard midi files. In *Sound and Music Computing Conference (SMC 2019)*, pages 65–71, 2019.

[128] Cong Jin, Tao Wang, Xiaobing Li, Chu Jie Jiessie Tie, Yun Tie, Shan Liu, Ming Yan, Yongzhi Li, Junxian Wang, and Shenze Huang. A transformer generative adversarial network for multi-track music generation. *CAAI Transactions on Intelligence Technology*, 7(3):369–380, 2022.

[129] Douglas Porter Johnson, Alan Tyson, and Robert Winter. *The Beethoven sketchbooks: History, reconstruction, inventory*. Univ of California Press, 1985.

[130] Randolph Johnson. The standard, power, and color model of instrument combination in romantic-era symphonic works. *Empirical Musicology Review*, 6(1), 2011.

[131] Anna Jordanous. A standardised procedure for evaluating creative systems: Computational creativity evaluation based on what it is to be creative. *Cognitive Computation*, 4(3):246–279, 2012.

[132] Anna Jordanous. Has computational creativity successfully made it "beyond the fence" in musical theatre? *Connection Science*, 29:350–386, 10 2017.

[133] Ryosuke Kamada, Seiji Tsuchiya, and Hirokazu Watabe. Automated orchestration systems based on deep learning. In *2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE)*, pages 163–167, 2023.

[134] Anna Kantosalo and Anna Jordanous. Role-based perceptions of computer participants in human-computer co-creativity. In *AISB Symposium of Computational Creativity (CC@AISB 2020)*, 2020.

[135] Ioannis Karydis, Alexandros Nanopoulos, Apostolos Papadopoulos, Emilios Cambouropoulos, and Yannis Manolopoulos. Horizontal and vertical integration/segregation in auditory streaming: a voice separation algorithm for symbolic musical data. In *Sound and Music Computing Conference (SMC 2007)*, pages 299–306, 2007.

[136] Emmanouil Karystinaios and Gerhard Widmer. Roman Numeral Analysis With Graph Neural Networks: Onset-Wise Predictions From Note-Wise Features. In *International Society for Music Information Retrieval Conference (ISMIR 2023)*, pages 597–604. ISMIR, December 2023.

[137] Savvas Kazazis, Philippe Depalle, and Stephen McAdams. Sound morphing by audio descriptors and parameter interpolation. In *Proceedings of the 19th International Conference on Digital Audio Effects (DAFx-16). Brno, Czech Republic*, 2016.

[138] Savvas Kazazis, Philippe Depalle, and Stephen McAdams. Ordinal scaling of timbre-related spectral audio descriptors. *The Journal of the Acoustical Society of America*, 149(6):3785–3796, 06 2021.

[139] Roger A. Kendall and Edward C. Carterette. Identification and blend of timbres as a basis for orchestration. *Contemporary Music Review*, 9(1-2):51–67, 1993.

[140] Diederik P. Kingma. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

[141] Janin Koch, Prashanth Thattai Ravikumar, and Filipe Calegario. Agency in Co-Creativity: Towards a Structured Analysis of a Concept. In *ICCC 2021 - 12th International Conference on Computational Creativity*, volume 1 of *Proceedings of the 12th International Conference on Computational Creativity*, pages 449 – 452, Mexico City (online), Mexico, September 2021. Association for Computational Creativity (ACC). In the Proceedings of the Second Workshop on the Future of Co-Creative Systems.

[142] Charles Koechlin. *Traité de l'orchestration*. Max Eschig, 1941 (completed), 1954-1959 (posthumous ed.).

[143] Panayiotis Kokoras. Towards a holophonic musical texture. In *International Computer Music Conference (ICMC 2005)*, 2005.

[144] Michael Krause, Christof Weiß, and Meinard Müller. A Cross-Version Approach to Audio Representation Learning for Orchestral Music. In *International Society for Music Information Retrieval Conference (ISMIR 2023)*, pages 832–839. ISMIR, December 2023.

[145] Kenneth Kreitner, Mary Térey-Smith, Jack Westrup, D. Kern Holoman, G.W. Hopkins, Paul Griffiths, and Jon Alan Conrad. Instrumentation and orchestration. In *Grove Music Online*. Oxford University Press, 2001.

[146] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[147] Audrey Laplante and Jean-Sébastien Sauvé. Attitudes of music scholars towards digital musicology. In *Proceedings of the 10th International Conference on Digital Libraries for Musicology*, DLfM '23, page 35–39, New York, NY, USA, 2023. Association for Computing Machinery.

[148] Dinh-Viet-Toan Le, Louis Bigo, Mikaela Keller, and Dorien Herremans. Natural language processing methods for symbolic music generation and information retrieval: a survey, 2024. (preprint).

[149] Dinh-Viet-Toan Le, Mathieu Giraud, Florence Levé, and Francesco Maccarini. A corpus describing orchestral texture in first movements of classical and early-romantic symphonies. In *Digital Libraries for Musicology (DLfM 2022)*, pages 22–35, 2022.

[150] Dinh-Viet-Toan Le and Yi-Hsuan Yang. Meteor: Melody-aware texture-controllable symbolic orchestral music generation, 2024. (preprint).

[151] Sven-Amin Lembke, Scott Levine, and Stephen McAdams. Blending Between Bassoon and Horn Players: An Analysis of Timbral Adjustments During Musical Performance. *Music Perception*, 35(2):144–164, 12 2017.

[152] Sven-Amin Lembke and Stephen McAdams. The role of spectral-envelope characteristics in perceptual blending of wind-instrument sounds. *Acta Acustica united with Acustica*, 101(5):1039–1051, 2015.

[153] Julian Lenz and Anirudh Mani. Pertok: Expressive encoding and modeling of symbolic musical ideas and variations, 2024.

[154] Fred Lerdahl and Ray Jackendoff. *A Generative Theory of Tonal Music*. MIT Press, 1983.

[155] Janet M. Levy. Texture as a Sign in Classic and Early Romantic Music. *Journal of the American Musicological Society*, 35(3):482–531, 1982.

[156] John P. Lewis. Creation by refinement: a creativity paradigm for gradient descent learning networks. In *IEEE 1988 International Conference on Neural Networks*, pages 229–233 vol.2, 1988.

[157] Elad Liebman and Peter Stone. Artificial musical intelligence: A survey, 2020.

[158] Jiafeng Liu, Yuanliang Dong, Zehua Cheng, Xinran Zhang, Xiaobing Li, Feng Yu, and Maosong Sun. Symphony generation with permutation invariant language model. In *International Society for Music Information Retrieval Conference (ISMIR 2022)*, 2022.

[159] Ryan Louie, Andy Coenen, Cheng Zhi Huang, Michael Terry, and Carrie J Cai. Novice-AI music co-creation via AI-steering tools for deep generative models. In *Conference on Human Factors in Computing Systems (CHI 2020)*, pages 1–13, 2020.

[160] Gareth Loy. Musicians make a standard: The midi phenomenon. *Computer Music Journal*, 9(4):8–26, 1985.

[161] Peiling Lu, Xin Xu, Chenfei Kang, Botao Yu, Chengyi Xing, Xu Tan, and Jiang Bian. Musecoco: Generating symbolic music from text, 2023.

[162] Wei-Tsung Lu and Li Su. Deep learning models for melody perception: An investigation on symbolic music data. In *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1620–1625, 2018.

[163] Todd Lubart. How can computers be partners in the creative process: classification and commentary on the special issue. *International Journal of Human-Computer Studies*, 63(4-5):365–369, 2005.

[164] Fabien Lévy. A short Analysis of Ravel's Alborada del Gracioso with the concepts of Functional Orchestration. Timbre and Orchestration Resource, 2022. https://timbreandorchestration.org/tor/modules/taxonomy/analysis/alborada-del-gracioso/a-short-analysis-of-ravels-alborada-del-gracioso-with-the-concepts-of-functional-orchestration.

[165] Francesco Maccarini, Mael Oudin, Mathieu Giraud, and Florence Levé. Co-creative orchestration of Angeles with layer scores and orchestration plans. In Colin Johnson, Sérgio M. Rebelo, and Iria Santos, editors, *Artificial Intelligence in Music, Sound, Art and Design*, pages 228–245, Cham, 2024. Springer Nature Switzerland.

[166] Dimos Makris, Ioannis Karydis, and Emilios Cambouropoulos. VISA3: Refining the voice intergration/segregation algorithm. In *Sound and music computing conference (SMC 2016)*, 2016.

[167] Martin E. Malandro. Composer's Assistant: An Interactive Transformer for Multi-Track MIDI Infilling. In *International Society for Music Information Retrieval Conference (ISMIR 2023)*, pages 327–334. ISMIR, December 2023.

[168] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008.

[169] Yan Maresz. On computer-assisted orchestration. *Contemporary Music Review*, 32(1):99–109, 2013.

[170] Alan Marsden. *"What was the question?": music analysis and the computer*, pages 137–147. Crawford, Gibson (eds.), Farnham: Ashgate, 2009.

[171] Alan Marsden. Schenkerian analysis by computer. *Journal of New Music Research*, 39(3):269–289, 2010.

[172] Alan Marsden. Music analysis by computer: Ontology and epistemology. In Meredith [184], pages 3–28.

[173] Deborah Mateja and Armin Heinzl. Towards machine learning as an enabler of computational creativity. *IEEE Transactions on Artificial Intelligence*, 2(6):460–475, 2021.

[174] Guerino Mazzola and Moreno Andreatta. Diagrams, gestures and formulae in music. *Journal of Mathematics and Music*, 1(1):23–46, 2007.

[175] Stephen McAdams. Timbre as a structuring force in music. In Kai Siedenburg, Charalampos Saitis, Stephen McAdams, Arthur N. Popper, and Richard R. Fay, editors, *Timbre: Acoustics, Perception, and Cognition*, Springer Handbook of Auditory Research, pages 211–243. Springer, 2019.

[176] Stephen McAdams and Albert Bregman. Hearing musical streams. *Computer Music Journal*, 3(4):26–60, 1979.

[177] Stephen McAdams and Bruno L. Giordano. 113The Perception of Musical Timbre. In *The Oxford Handbook of Music Psychology*. Oxford University Press, 01 2016.

[178] Stephen McAdams, Meghan Goodchild, and Kit Soden. A taxonomy of orchestral grouping effects derived from principles of auditory perception. *Music Theory Online*, 28(3), 2022.

[179] Matthew McCloskey, Gabrielle Curcio, Amulya Badineni, Kevin McGrath, Georgios Papamichail, and Dimitris Papamichail. Automated Arrangements of Multi-Part Music for Sets of Monophonic Instruments. In *Proceedings of the 16th International Symposium on Computer Music Multidisciplinary Research*, page 379–386. Zenodo, November 2023.

[180] Jon McCormack, Camilo Cruz Gambardella, Nina Rajcic, Stephen James Krol, Maria Teresa Llano, and Meng Yang. Is writing prompts really making art? In *Artificial Intelligence in Music, Sound, Art and Design*, Lecture Notes in Computer Science, pages 196–211, 2023.

[181] Jon McCormack, Toby Gifford, and Patrick Hutchings. Autonomy, authenticity, authorship and intention in computer generated art. In *International Conference on Computational Intelligence in Music, Sound, Art and Design (EvoMUSART 2019)*, pages 35–50, 2019.

[182] George Frederick McKay. *Creative Orchestration*. Allyn and Bacon, 1963.

[183] Neil McLachlan, David Marco, Maria Light, and Sarah Wilson. Consonance and pitch. *Journal of Experimental Psychology: General*, 2013.

[184] David Meredith. *Computational Music Analysis*. Springer, 2015.

[185] Arthur I. Miller. *The Artist in the Machine: The World of AI-Powered Creativity*. The MIT Press, 10 2019.

[186] Lejun Min, Junyan Jiang, Gus Xia, and Jingwei Zhao. Polyffusion: A diffusion model for polyphonic score generation with internal and external controls. In *International Society for Music Information Retrieval Conference (ISMIR 2023)*, 2023.

[187] Eduardo Miranda, Aurélien Antoine, Jean-Michaël Celerier, and Myriam Desainte-Catherine. i-Berlioz : Interactive Computer-Aided Orchestration with Temporal Control. In *Proceedings of 5th International Conference on New Music Concepts (ICNMC)*, Treviso, Italy, 2018.

[188] Marius Miron, Julio J. Carabias-Orti, Juan J. Bosch, Emilia Gómez, and Jordi Janer. Score-informed source separation for multichannel orchestral recordings. *Journal of Electrical and Computer Engineering*, 2016.

[189] Marius Miron, Julio José Carabias-Orti, and Jordi Janer. Audio-to-score alignment at the note level for orchestral recordings. In *International Society for Music Information Retrieval Conference (ISMIR 2014)*, pages 125–130, 2014.

[190] Gautam Mittal, Jesse Engel, Curtis Hawthorne, and Ian Simon. Symbolic music generation with diffusion models. In *International Society for Music Information Retrieval Conference (ISMIR 2021)*, pages 468–475. ISMIR, October 2021.

[191] Raymond Monelle. *Linguistics and Semiotics in Music*. Routledge, New York, May 1992.

[192] Robert Moog. Midi: musical instrument digital interface. *Journal of the Audio Engineering Society*, 34(5):394–404, 1986.

[193] Bhavya Mor, Sunita Garhwal, and Ajay Kumar. A systematic literature review on computational musicology. *Archives of Computational Methods in Engineering*, 27:923–937, 2020.

[194] Fabio Morreale, Megha Sharma, and I-Chieh Wei. Data Collection in Music Generation Training Sets: A Critical Analysis. In *International Society for Music Information Retrieval Conference (ISMIR 2023)*, pages 37–46. ISMIR, December 2023.

[195] Michael C. Mozer. Neural Network Music Composition by Prediction: Exploring the Benefits of Psychoacoustic Constraints and Multi-scale Processing. In *Musical Networks: Parallel Distributed Perception and Performance*. The MIT Press, 02 1999.

[196] Aashiq Muhamed, Liang Li, Xingjian Shi, Suri Yaddanapudi, Wayne Chi, Dylan Jackson, Rahul Suresh, Zachary C. Lipton, and Alex J. Smola. Symbolic music generation with transformer-gans. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(1):408–417, May 2021.

[197] Meinard Müller. *Fundamentals of music processing: Audio, analysis, algorithms, applications*, volume 5. Springer, 2015.

[198] Takuto Nabeoka, Eita Nakamura, and Kazuyoshi Yoshii. Automatic orchestration of piano scores for wind bands with user-specified instrumentation. In *International Symposium on Computer Music Modeling and Retrieval (CMMR 2023)*, 2023.

[199] Eita Nakamura and Kazuyoshi Yoshii. Statistical piano reduction controlling performance difficulty. *APSIPA Transactions on Signal and Information Processing*, 7:e13, 2018.

[200] Daiki Naruse, Tomoyuki Takahata, Yusuke Mukuta, and Tatsuya Harada. Pop Music Generation with Controllable Phrase Lengths. In *Proceedings of the 23rd International Society for Music Information Retrieval Conference*, pages 125–131. ISMIR, November 2022.

[201] Jean-Jacques Nattiez. *Fondements d'une Sémiologie de La Musique*. Dufrenne, 1975.

[202] Jason Noble, Kit Soden, and Zachary Wallmark. The semantics of orchestration: A corpus analysis. In *2nd International Conference on Timbre*, 2020.

[203] Quentin R. Nordgren. A measure of textural patterns and strengths. *Journal of Music Theory*, 4(1):19–31, 1960.

[204] Sageev Oore, Ian Simon, Sander Dieleman, Douglas Eck, and Karen Simonyan. This time with feeling: Learning expressive musical performance. *Neural Computing and Applications*, 32(4):955–967, 2020-02-01.

[205] OpenAI. MuseNet. OpenAI, April 2019. `https://openai.com/blog/musenet/`.

[206] François Pachet. A joyful ode to automatic orchestration. *ACM Transactions on Intelligent Systems and Technology*, 8(2):18:1–18:13, 2016-10-03.

[207] Charles de Paiva Santana and Didier Guigue. The role of orchestration in shaping musical form: Theory and practice of a methodological proposal and its computational implementation. *Musicological Annual*, 58(2):121–153, Dec. 2022.

[208] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[209] A. Parmentier, K. Déguernel, and C. Frei. A modular tool for automatic sound-painting query recognition and music generation in Max/MSP. In *Sound and Music Computing Conference (SMC 2021)*, 2021.

[210] Geoffroy Peeters, Bruno L. Giordano, Patrick Susini, Nicolas Misdariis, and Stephen McAdams. The Timbre Toolbox: Extracting audio descriptors from musical signals. *The Journal of the Acoustical Society of America*, 130(5):2902–2916, 11 2011.

[211] Walter Piston. *Orchestration*. Norton, 1955.

[212] Ernst Praetorius, editor. *Haydn Symphonies*. Ernst Eulenburg, 1934.

[213] James Pritchett. The completion of john cage's freeman etudes. *Perspectives of New Music*, 32(2):264–270, 1994.

[214] David Psenicka. Sporch: An algorithm for orchestration based on spectral analyses of recorded sounds. In *ICMC*, 2003.

[215] Xingwei Qu, Yuelin Bai, Yinghao Ma, Ziya Zhou, Ka Man Lo, Jiaheng Liu, Ruibin Yuan, Lejun Min, Xueling Liu, Tianyu Zhang, Xinrun Du, Shuyue Guo, Yiming Liang, Yizhi Li, Shangda Wu, Junting Zhou, Tianyu Zheng, Ziyang Ma, Fengze Han, Wei Xue, Gus Xia, Emmanouil Benetos, Xiang Yue, Chenghua Lin, Xu Tan, Stephen W. Huang, Wenhu Chen, Jie Fu, and Ge Zhang. Mupt: A generative symbolic music pretrained transformer. arXiv 2404.06393, 2024.

[216] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018.

[217] Dimitrios Rafailidis, Alexandros Nanopoulos, Yannis Manolopoulos, and Emilios Cambouropoulos. Detection of stream segments in symbolic musical data. In *International Conference on Music Information Retrieval (ISMIR 2008)*, pages 83–88, 2008.

[218] Dimitris Rafailidis, Emilios Cambouropoulos, and Yannis Manolopoulos. Musical voice integration/segregation: Visa revisited. In *Sound and Music Computing Conference (SMC 2009)*, pages 42–47, 2009.

[219] Gardner Read. *Style and orchestration*. Schirmer Books, 1979.

[220] Yi Ren, Jinzheng He, Xu Tan, Tao Qin, Zhou Zhao, and Tie-Yan Liu. PopMAG: Pop music accompaniment generation. In *ACM International Conference on Multimedia (MM 2022)*, pages 1198–1206. Association for Computing Machinery, 2020-10-12.

[221] Mark M Reybrouck. Musical creativity between symbolic modelling and perceptual constraints: The role of adaptive behaviour and epistemic autonomy. In *Musical Creativity*, pages 58–76. Psychology Press, 2006.

[222] Matthias C. Rillig, Marlene Ågerstrand, Mohan Bi, Kenneth A. Gould, and Uli Sauerland. Risks and benefits of large language models for the environment. *Environmental Science & Technology*, 57(9):3464–3466, 2023. PMID: 36821477.

[223] Nikolay Rimsky-Korsakov. *Principles of Orchestration: With Musical Examples Drawn from His Own Works*, volume 1. Édition russe de musique, 1873 (begun), 1912 (posthumous ed.).

[224] Jean-Claude Risset and David L. Wessel. Exploration of timbre by analysis and synthesis. In Diana Deutsch, editor, *The Psychology of Music (Second Edition)*, Cognition and Perception, pages 113–169. Academic Press, San Diego, second edition edition, 1999.

[225] David Rizo, Plácido R Illescas, and José M Inesta. Interactive melodic analysis. *Computational Music Analysis*, pages 191–219, 2016.

[226] David Rizo, Pedro J. Ponce De León, Antonio Pertusa, and Jose M. Iñesta. Melody track identification in music symbolic files. In *International Florida Artificial Intelligence Research Society Conference (FLAIRS 2006)*, 2006.

[227] Martin Rohrmeier. Towards a generative syntax of tonal harmony. *Journal of Mathematics and Music*, 5(1):35–53, 2011.

[228] Perry Roland. The music encoding initiative (mei). In *First International Conference on Musical Applications Using XML*, volume 1060, pages 55–59, 2002.

[229] François Rose and James E Hetrick. Enhancing orchestration technique via spectrally based linear algebra methods. *Computer Music Journal*, 33(1):32–41, 2009.

[230] Charles Rosen. *The Classical Style: Haydn, Mozart, Beethoven*. Number 653 in The Norton library. WW Norton & Company, 1997.

[231] Lluc Bono Rosselló and Hugues Bersini. Music generation with multiple ant colonies interacting on multilayer graphs. In *Artificial Intelligence in Music, Sound, Art and Design*, Lecture Notes in Computer Science, pages 34–49, 2023.

[232] Lewis Eugene Rowell. *Thinking About Music: An Introduction to the Philosophy of Music*. University of Massachusetts Press, Amherst, 1983.

[233] Friedemann Sallis. *Music sketches*. Cambridge University Press, 2015.

[234] Gregory J Sandell. Roles for spectral centroid and other factors in determining "blended" instrument pairings in orchestration. *Music Perception*, 13(2):209–246, 1995.

[235] Ian Sapiro. *Scoring the Score: The Role of the Orchestrator in the Contemporary Film Industry*. Routledge, 1st edition, 2016.

[236] Craig Stuart Sapp. Online database of scores in the Humdrum file format. In *International Conference on Music Information Retrieval (ISMIR 2005)*, pages 664–665, 2005.

[237] Pedro Sarmento, Adarsh Kumar, Yu-Hua Chen, CJ Carr, Zack Zukowski, and Mathieu Barthet. GTR-CTRL: Instrument and genre conditioning for guitar-focused music generation with transformers. In Colin Johnson, Nereida Rodríguez-Fernández, and Sérgio M. Rebelo, editors, *Artificial Intelligence in Music, Sound, Art and Design*, Lecture Notes in Computer Science, pages 260–275, 2023.

[238] Alfred Schnittke. Timbral relationships and their functional use. *Orchestration: An anthology of writings*, pages 162–175, 2006.

[239] Arnold Schönberg. *Harmonielehre*, volume 91. Universal-edition, 1922.

[240] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics.

[241] Zhengshan Shi, Craig Sapp, Kumaran Arul, Jerry McBride, and Julius O Smith III. SUPRA: Digitizing the stanford university piano roll archive. In *International Society for Music Information Retrieval Conference (ISMIR 2019)*, pages 517–523, 2019.

[242] Yi-Jen Shih, Shih-Lun Wu, Frank Zalkow, Meinard Müller, and Yi-Hsuan Yang. Theme transformer: Symbolic music generation with theme-conditioned transformer. *IEEE Transactions on Multimedia*, 25:3495–3508, 2023.

[243] Ian Simon, Adam Roberts, Colin Raffel, Jesse Engel, Curtis Hawthorne, and Douglas Eck. Learning a latent space of multitrack measures, 2018.

[244] Federico Simonetta, Carlos Cancino-Chacón, Stavros Ntalampiras, and Gerhard Widmer. A convolutional approach to melody line identification in symbolic scores. In *International Society for Music Information Retrieval Conference (ISMIR 2019)*, pages 924–931, 2019.

[245] Federico Simonetta, Filippo Carnovalini, Nicola Orio, and Antonio Rodà. Symbolic music similarity through a graph-based representation. In *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion*, AM '18, New York, NY, USA, 2018. Association for Computing Machinery.

[246] Christopher Small. *Musicking: The meanings of performing and listening*. Wesleyan University Press, 1998.

[247] Don L. Smithers. Mozart's orchestral brass. *Early music*, 20(2):254–266, 1992.

[248] Louis Soum-Fontez, Mathieu Giraud, Nicolas Guiomard-Kagan, and Florence Levé. Symbolic textural features and melody/accompaniment detection in string quartets. In *International Symposium on Computer Music and Multidisciplinary Research (CMMR 2021)*, 2021.

[249] John Spitzer and Neal Zaslaw. *The birth of the orchestra: History of an institution, 1650-1815*. Oxford University Press, 2004.

[250] Jane R. Stevens. Theme, harmony, and texture in classic-romantic descriptions of concerto first-movement form. *Journal of the American Musicological Society*, 27(1):25–60, 1974.

[251] Bob Sturm and Arthur Flexer. A review of validity and its relationship to music information research. In *Int. Symp. Music Information Retrieval*, 2023.

[252] Sturm, Bob and Santos, João Felipe and Korshunova, Iryna. Folk music style modelling by recurrent neural networks with long short term memory units. In *16th International Society for Music Information Retrieval Conference, late-breaking demo session*, page 2, 2015.

[253] Philip Tagg. Melody and accompaniment. In *Encyclopedia of Popular Music of the World (EPMOW)*, 2000.

[254] Hirofumi Takamori, Haruki Sato, Takayuki Nakatsuka, and Shigeo Morishima. Automatic arranging musical score for piano using important musical elements. In *Sound and Music Computing Conference (SMC 2017)*, pages 35–41, 2017.

[255] Pierre Talbot, Carlos Agon, and Philippe Esling. Interactive computer-aided composition with constraints. In *43rd International Computer Music Conference (ICMC 2017)*, Shanghai, China, October 2017.

[256] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In Věra Kůrková, Yannis Manolopoulos, Barbara Hammer, Lazaros Iliadis, and Ilias Maglogiannis, editors, *Artificial Neural Networks and Machine Learning – ICANN 2018*, pages 270–279, Cham, 2018. Springer International Publishing.

[257] Jingjing Tang, Geraint Wiggins, and Gyorgy Fazekas. Reconstructing human expressiveness in piano performances with a transformer network, 2023.

[258] Peter M. Todd. A sequential network design for musical applications. In *Proceedings of the 1988 connectionist models summer school*, pages 76–84, 1988.

[259] Peter M. Todd. A Connectionist Approach To Algorithmic Composition. In *Music and Connectionism*. The MIT Press, 10 1991.

[260] Maude Alice Trimmer. *Texture and Sonata Form in the Late String Chamber Music of Haydn and Mozart*, volume 1. City University of New York, 1981.

[261] Hiroaki Tsushima, Eita Nakamura, Katsutoshi Itoyama, and Kazuyoshi Yoshii. Interactive arrangement of chords and melodies based on a tree-structured generative model. In *International Society for Music Retrieval Conference (ISMIR 2018)*, 2018.

[262] Alexandra Uitdenbogerd and Justin Zobel. Melodic matching techniques for large music databases. In *International Conference on Multimedia (Multimedia 99)*, pages 57–66, 1999.

[263] Max Unger, editor. *Beethoven Symphonies*. Ernst Eulenburg, 1938.

[264] Andrea Valenti, Antonio Carta, and Davide Bacciu. Learning style-aware symbolic music representations by adversarial autoencoders. In *ECAI 2020*, pages 1563–1570. IOS Press, 2020.

[265] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[266] Olga Vechtomova and Gaurav Sahu. LyricJam Sonic: A generative system for real-time composition and musical improvisation. In *Artificial Intelligence in Music, Sound, Art and Design*, Lecture Notes in Computer Science, pages 292–307, 2023.

[267] Michele Della Ventura. Voice Separation in Polyphonic Music: Information Theory Approach. In Lazaros Iliadis, Ilias Maglogiannis, and Vassilis Plagianakos, editors, *Artificial Intelligence Applications and Innovations*, IFIP Advances in Information and Communication Technology, pages 638–646. Springer International Publishing, 2018.

[268] Daniel Villegas Vélez. The matter of timbre: Listening, genealogy, sound. In Emily Dolan and Alexander Rehding, editors, *The Oxford Handbook of Timbre*. Oxford University Press, 11 2021.

[269] Anja Volk, Frans Wiering, and Peter Van Kranenburg. Unfolding the potential of computational musicology. In *International Conference on Informatics and Semiotics in Organisations (ICIS 2011)*, pages 137–144, 2011.

[270] Dimitri von Rütte, Luca Biggio, Yannic Kilcher, and Thomas Hofmann. Figaro: Controllable music generation using learned and expert features. In *The Eleventh International Conference on Learning Representations*, 2022.

[271] Ziyu Wang, Dingsu Wang, Yixiao Zhang, and Gus Xia. Learning interpretable representation for controllable polyphonic music generation. *International Society for Music Information Retrieval Conference (ISMIR 2020)*, 2020.

[272] Ziyu Wang and Gus Xia. MuseBERT: Pre-training Music Representation for Music Understanding and Controllable Generation. In *International Society for Music Information Retrieval Conference (ISMIR 2021)*, pages 722–729. ISMIR, October 2021.

[273] Ziyu Wang, Dejing Xu, Gus Xia, and Ying Shan. Audio-to-symbolic arrangement via cross-modal music representation learning. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 181–185, 2022.

[274] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3:1–40, 2016.

[275] Christof Weiß, Vlora Arifi-Müller, Michael Krause, Frank Zalkow, Stephanie Klauk, Rainer Kleinertz, and Meinard Müller. Wagner ring dataset: A complex opera scenario for music processing and computational musicology. *Transactions of the International Society for Music Information Retrieval*, Oct 2023.

[276] David L. Wessel. Timbre space as a musical control structure. *Computer Music Journal*, 3(2):45–52, 1979.

[277] Taxonomy Workgroup. A timbral and orchestrational pluralistic analysis of Ravel's Alborada del gracioso. Timbre and Orchestration Resource, 2022. `https://timbreandorchestration.org/tor/modules/taxonomy/analysis/alborada-del-gracioso/all-analyses`.

[278] Iannis Xenakis. *Formalized music: thought and mathematics in composition*. Number 6 in Harmonologia series. Pendragon Press, 1992.

[279] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang. MidiNet: A convolutional generative adversarial network for symbolic-domain music generation. In *International Society for Music Information Retrieval Conference (ISMIR 2017)*, pages 324–331, 2017.

[280] Botao Yu, Peiling Lu, Rui Wang, Wei Hu, Xu Tan, Wei Ye, Shikun Zhang, Tao Qin, and Tie-Yan Liu. Museformer: Transformer with fine- and coarse-grained attention for music generation. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 1376–1388. Curran Associates, Inc., 2022.

[281] Tasos Zembylas and Martin Niederauer. *Composing processes and artistic agency: Tacit knowledge in composing*. Routledge, 2017.

[282] Mingliang Zeng, Xu Tan, Rui Wang, Zeqian Ju, Tao Qin, and Tie-Yan Liu. MusicBERT: Symbolic music understanding with large-scale pre-training. In *ACM International Conference on Multimedia (MM 2021)*, 2021.

[283] Jingwei Zhao, Gus Xia, and Ye Wang. Accomontage-3: Full-band accompaniment arrangement via sequential style transfer and multi-track function prior, 2023.

[284] Jingwei Zhao, Gus Xia, and Ye Wang. Q&a: Query-based representation learning for multi-track symbolic music re-arrangement. In Edith Elkind, editor, *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 5878–5886. International Joint Conferences on Artificial Intelligence Organization, 8 2023. AI and Arts.

Appendices

A. Annotation Syntax

This appendix reports the syntax for the annotation of orchestral layers and textures used in the corpus.

Instruments

Instrument name	Syntax
Flute	`Fl`
Oboe	`Ob`
Clarinet	`Cl`
Bassoon	`Fg`
Horn	`Hrn`
Trumpet	`Trp`
Timpani	`Timp`
Violin 1	`Vln1`
Violin 2	`Vln2`
Viola	`Vla`
Cello	`Vc`
Double bass	`Cb`

Table A.1.: Syntax for the annotation of instruments.

The syntax for the instruments is reported in Table A.1. An annotation file begins with the definition of the orchestral ensemble, as a list of instruments divided into their instrumental families in the following format:

```
InstList:  Wood:Fl.Ob.Cl.Fg|Brass:Hrn.Trp|Perc:Timp|Strings:Vln1.Vln2.Vla.Vc.Cb
```

Relations

Homorhythm (`-h`), parallel motion (`-p`), and unison or octave doublings (`-u`).

Roles

The syntax for the roles is reported in Table A.2. Annotating sub-roles is optional. When a sub-role is used, the main role can be omitted for brevity, such as `decmel` instead of `mel+rhythm::decmel`. Optional rhythm precision can be added to a role, such as:

Roles and sub-roles	**Syntax**
Melody	`mel`
Imitation	`mel::imitation`
Rhythmic accompaniment	`rhythm`
Repeated notes	`rhythm::repeat_note`
Oscillation	`rhythm::osc`
"Batterie"	`rhythm::batt`
Arpeggio	`rhythm::arp`
Scale	`rhythm::scale`
Harmonic accompaniment	`harm`
Mixed	
Decorative melody	`mel+rhythm::decmel`
Sparse elements or Sparse chords	`harm+rhythm::sparse`

Table A.2.: Syntax for the annotation of roles and sub-roles.

- `rhythm16::repeat_note`: Rhythmic accompaniment composed of semiquavers
- . . .

Layers

A layer is composed of an identifier `id` that gathers instruments, their relation and an optional role (if the role is omitted, its role is considered as None). Layers are annotated as:

`id:role-relation`
or
`id:-relation`

A `~` can be added `~id:role-relation`, meaning that the layer continues from a previous segment, as described below.

Identifiers

An identifier is a letter used to group instruments belonging to the same layer. We adopt the following (non-compulsory) convention for chosing the letter.

- Melody: `a`, `b`, `c`, `d`, . . .
- Rhythmic accompaniment: `r`, `s`, `u`, `v`, . . .
- Harmonic accompaniment: `h`, `i`, `l`, `m`, . . .
- Tutti (or almost all instruments): `t`

Segment

A segment is composed of mulitple layers. Segments are annotated as:
`[measure_range] <identifiers> layer1 layer2 ...  {specification}`

`measure_range` is the range of the segment, the order of `<identifiers>` is defined by the ensemble `InstList` at the beginning of the file, the optional `{specification}` can describe further properties or effects such as CR (Call-and-response) schemes.

Example A.1. Consider the following line.
`[5] <h000|h0|0|rrrr0> h:harm-p r:repeat_note-h`
Using the same heading `InstList` defined above, at measure 5, two layers are heard:

- A layer (identified by `h:`) composed of flutes and horns playing sustained harmonic in parallel motion.
- A layer (identified by `r:`) composed of violins 1 & 2, violas and cellos playing repeated notes in homorhythm.

Layers that continues through different segments can be annotated as `~layer`. This allows the instrumentation of a given layer to change at a successive measure.

Example A.2. Consider the following lines.
`[5] <h000|h0|0|rrrr0> h:harm-p r:repeat_note-h`
`[6-7] <h000|00|0|aa000> ~h:harm-p a:mel-u`
The harmonic layer first played by flutes and horns in measure 5 continues in measures 6 and 7, with flutes only.

Separated instruments (*divisi*) which belong to different layers can be entered with parenthesis, such as `<(ha)000|h0|0|rrrr0>`, where the flutes have been split in two.

Measure Range

A measure range can be specified by any combination of single measures or ranges, such as `[1-6]`, `[4]`, `[1-6,8]`, or `[1-6,15-17,19]`.

Meta-Layers

The optional `{specification}` can describe further properties or effects such as CR (Call-and-response). The typical structure is the following.

```
{XX frequency (identifiers1)[measure-list](identifiers2)[measure-list]...}
```

- `XX` indicates the kind of effect, for example CR (Call-and-Response) or TE (Timbral Echoes),
- `frequency` is a number that has a different meaning according to the effect,

- `identifiers` contains a list of identifiers (parenthesis are omitted if only one identifier is present) and
- `measure-list` contains the measures to which the identifiers refer to.

Example A.3. Consider the following specification.
`{CR 1 a[595,597]a{596,598}}`
This means that there is a call and response scheme in measures 595-598 between the instruments identified with `a` in measures 595 and 597, and the instruments identified with `a` in measures 596 and 598. The call and the response alternates with a frequency of 1 measure.

Score

The full score is composed of segments.
Comments are entered at any location with `# Comment`.
Formal sections are annotated beginning with `#!label Comment`. In this case, segments between a line `#!label1` and `#!label2` are labeled as `label1`.

B. Constructing Layer Scores

Here are the steps to build a layer score from an existing musical score: we start by analyzing it to identify its layers, we reduce them to their essential elements (main melodic lines, rhythmic figures, and pitches forming the harmony), and then we write the notes in a score with as many staves as layers. These staves do not carry instrument names but are named after the abstract layers to which they correspond.
In practice a layer score can be created in different ways and for different purposes. We can classify layer scores for three aspects. The first one is its origin: we can build a layer score starting from an orchestral score, a piano score, or a score for a different ensemble (for example for solo guitar or for string quartet, among others). In case two or more versions of the same piece exists we would expect only one common layer score for them all. The second aspect is that of the purpose, the final goal for which a layer score is constructed. This can be purely analytical, to facilitate the study and the comprehension of the music, or it can be in the context of a creative project, like an orchestration or a piano reduction. Finally, these operations can be done completely manually, or automated algorithmically. In Table B.1, we classify some of the experiences that have been done in the context of this thesis for origin, goal, and procedure.

		Origin Score		Goal		Procedure	
Excerpt	n meas.	Piano	Orchestral	Analytic/Corpus	Co-creative Project	Manual	Automated
HMB Symphonies corpus (Appendix B.1)	8528		yes	both			yes
Mozart, *Symphony No. 40 (Exposition)* (Appendix B.2)	101		yes	yes		both	
Velarde, *Angeles (mov. 2 and 6)* (Section 7.2)	109	yes			yes	yes	

Table B.1.: Classification of three aspects of constructed layer scores: origin (piano or orchestral score), goal (for analysis and corpus creation purposes or for a (co-)creative project), and construction procedure (manual or automated). The full Haydn, Mozart, and Beethoven (HMB) symphonies corpus has automated layer scores, whereas we manually produced layer scores for the two movements of the *Angeles* project. The layer score for the exposition (measures 1-101) of Mozart Symphony No. 40 has been automated, and manually revised.

B.1. Algorithmic Construction of Layer Scores from Annotated Orchestral Scores

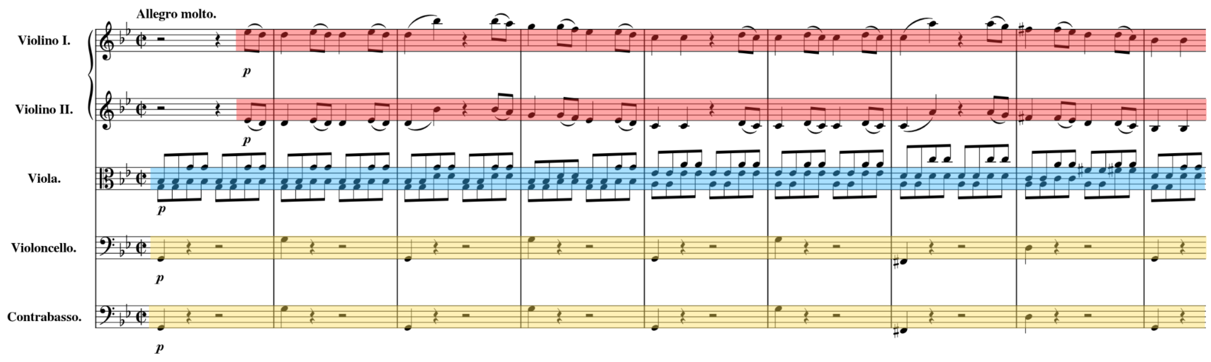
To model certain tasks in orchestration it can be useful to have at one's disposal a dataset of layer scores, paired with orchestral scores. The operation of writing layer scores completely manually, either by imputing notes on a blank score or by modifying the orchestral score, is extremely time consuming, so we decided to generate automatically the layer scores of the first movements of Symphonies by Haydn, Mozart, and Beethoven in the corpus, starting from the orchestral scores and the annotations of texture (Task T4). We have implemented a simple algorithm (Algorithm 1) in python to work with tokenized MIDI files (using the tokenization method presented in Section 8.1) but it could be implemented also to work directly with MIDI files and with musicXML scores.
Some practical considerations are that the number of layers in a piece can easily be over what is possible to fit in a page of a printed score. Moreover, layers have a limited time span. Thus, assigning one staff to each layer would result in a very sparse score, with silence almost everywhere. Consider for example an orchestral layer that contains two instruments for the first 3 measures of a piece. From measure 4 onward, the corresponding staff in the layer score would be completely filled with pauses. For practical reasons, we adopt a compressed representation in the implementation. Consider that each layer has a limited time span, with a beginning and an end. At the beginning of the piece we initialize the layer score with a number of staves that corresponds to the number of layers that have their beginning at measure 1. We place each of these layers in one of the staves, and then we proceed measure by measure. When a layer reaches its end, the staff on which it is written becomes available for other layers whose beginning is later in time. When a new layer starts, we place it on the first staff that is available starting from the top of the page, and if no staff is available, a new one is created to fit the new layer. The positions in the score of layers are recorded in a file with a similar format of that of the annotations, with the difference that each layer is mapped to one and only one staff.
Another practical problem is that of notes that do not belong to any layer. The end of a melodic layer, for example, usually coincides with the end of a phrase, which might be concluding with a note on the downbeat of a measure. According to annotation rules, this measure is not annotated as belonging to the same layer as the rest of the phrase. It could be part of a new layer that is starting there, or part of no layer if the rest of the measure consists of a pause. The same would happen for an anacrusis at the start of a phrase. It is not annotated as belonging to the layer whose annotation starts from the immediately following measure. Since those notes might not belong to any layer, there are several alternative strategies to deal with them in a layer score. One is to simply discard them, another one is to store them in a special staff dedicated to them (the first one, indexed by 0), and the third option would be to extend the layer to include them. The second strategy is the one we have adopted in our implementation. Notice that to discard all those notes, it is sufficient to erase the first staff from the output score. Further work should be done on the evaluation of the layer scores resulting from this

procedure. Quantities to assess the coherence of the resulting score and its adherence to the original orchestral score content should be defined. At this stage we subjectively assess them by visual inspection.

Algorithm 1: Construction of a layer score from an annotated orchestral score

```
layer_score ← empty_score;
available_staves ← [];
list_of_current_layers ← [];
for $m \in M$ do
    Retrieve from the annotations the layers ending at measure $m$;
    for layer ended at measure $m - 1$ (skip if $m == 1$) do
        Add corresponding staff $s$ to available_staves;
        Remove layer from list_of_current_layers;
    end
    Retrieve from the annotations the layers starting at measure $m$;
    for layer starting at measure $m$ do
        if available_staves is not empty then
            Select the lowest staff number $s$ in available_staves;
            Remove $s$ from available_staves;
        else
            Retrieve the highest staff number $h$ in layer_score ;
            Create new staff with staff number $s = h + 1$ in layer_score ;
        end
        Append layer identifier and staff number $s$ to list_of_current_layers;
    end
    for ins $\in \mathcal{E}$ do
        if (meas, ins) belongs to a layer in the annotations then
            current_layer ← layer(meas, ins);
            Search for staff $s$ corresesonding to current_layer in
              list_of_current_layers;
            Put notes from orchestral_score at (meas, ins) in layer_score at
              (meas, $s$);
        end
    end
    for $s \in$ layer_score do
        Reorder notes in time;
        Remove duplicate notes;
    end
end
```

(a) **Orchestral Score.** 11 tracks/instruments (5 in this extract)
Transcribed by Samuel Ricke on `musescore.com`
Textural labels from [149]. *Melody:* Vln1, Vln2; *Oscillation:* Vla (divided); *Sparse:* Vc, Cb



(b) **Layer Score.** 3 layers.

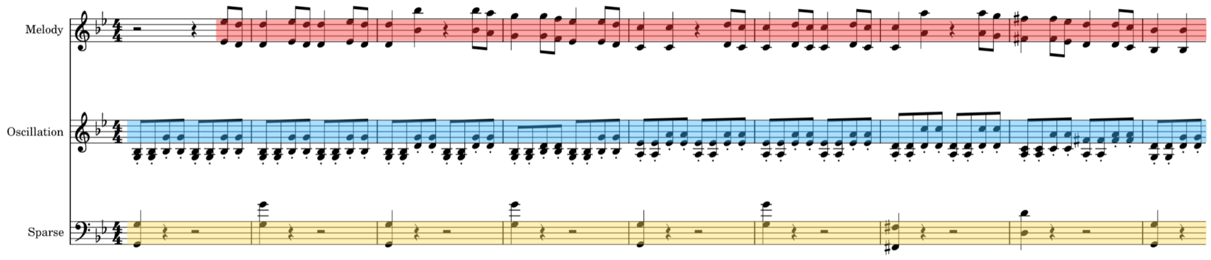
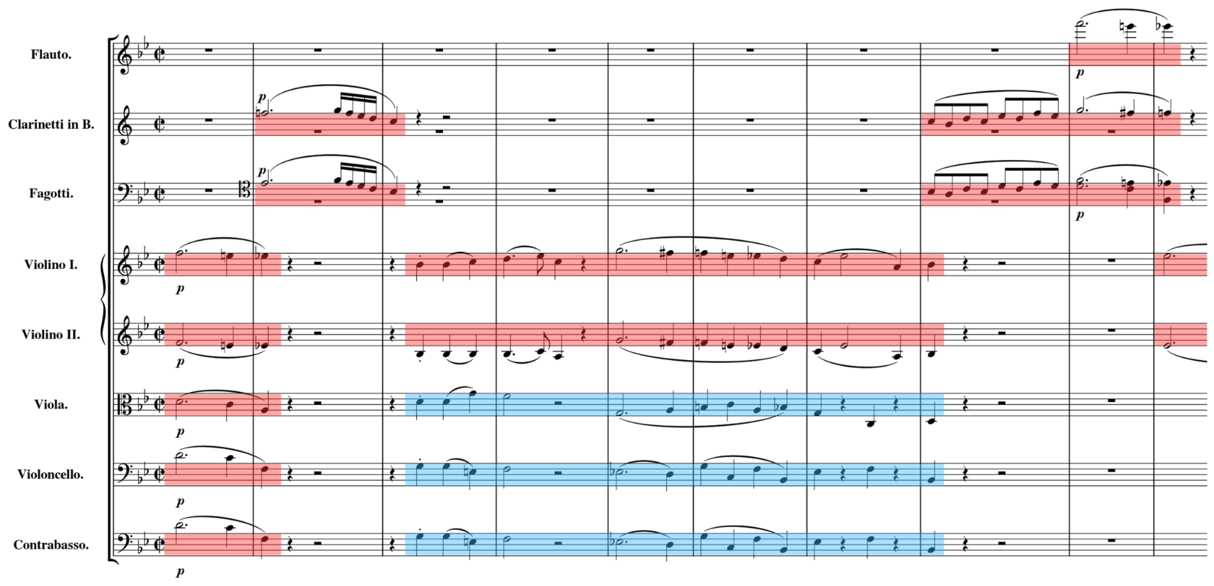


Figure B.1.: Beginning (measures 1-8) of the first movement of Symphony No. 40 by Mozart, k550. Orchestral score and manually curated layer score. The excerpt present a typical homophonic texture with melody and accompaniment. (a) The orchestral score can be decomposed into three layers: one with a *melodic* role (red, dark), one with a rhythmic role with sub-role *oscillation* (blue, light), and a third one with a rhythm and harmony mixed *sparse* role (yellow, very light). (b) The layer score contains one part for each of these three layers, λ_{mel}, $\lambda_{\mathrm{oscillation}}$, and $\lambda_{\mathrm{sparse}}$. Octave doublings have been kept here to highlight the subjectivity of choosing between them.

B.2. Manually Curating Layer Scores

When inspecting the results generated by the algorithm, there are a few weaknesses that emerged, For instance the undefined behavior for *divisi* instruments. A second problem is that of anacruses and phrase endings that we already mentioned in the previous section. In a good layer score, we would expect them to appear on the staff corresponding to the layer the rest of the phrase belongs to. Another fact is the presence of octave doublings, for which one octave should be selected. The criterion for the selection is not clear. For example, we might want to select the upper octave of a melodic layer in which the violin 2 doubles the violin 1 an octave lower, but we might

(a) **Orchestral Score.** 11 tracks/instruments (8 in this extract)
Transcribed by Samuel Ricke on `musescore.com`
Textural labels from [149]. *Melody:* passed around the orchestra; *Decorative melody:* Vla Vc, Cb



(b) **Layer Score.** 2 layers.

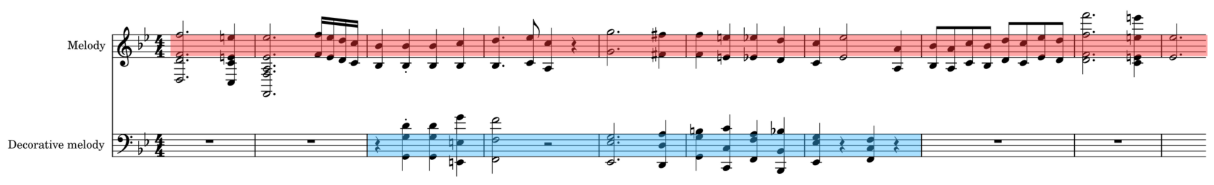


Figure B.2.: Second theme (measures 44-53) from the exposition of the first movement of Symphony No. 40 by Mozart, k550. Orchestral score and manually curated layer score. The texture of the excerpt is almost monophonic, with a melody passed around the orchestra, and occasionally supported by an homorhythmic part. The dialog between the different timbres is one of the most important elements of this excerpt. (a) The orchestral score can be decomposed into two layers: one with a *melodic* role (red, dark), and a second one with a *decorative melody* role (blue, light). (b) The layer score contains one part for each of the layers, λ_{mel}, and $\lambda_{\mathsf{decmel}}$. Octave doublings have been kept here to highlight the subjectivity of choosing between them.

want to select the lower octave when the piccolo flute doubles other parts an octave higher. One last problem that we observe is the fact that there is no coherent order of layer types in the staves of the score. For example, it seams reasonable to have the main melodic line always on the first staff, and we might want that, when a melody finishes,

(a) **Layer Score.** 6 layers. One *Melody* layer, four *Harmony* layers, and one *Repeated Notes* layer.

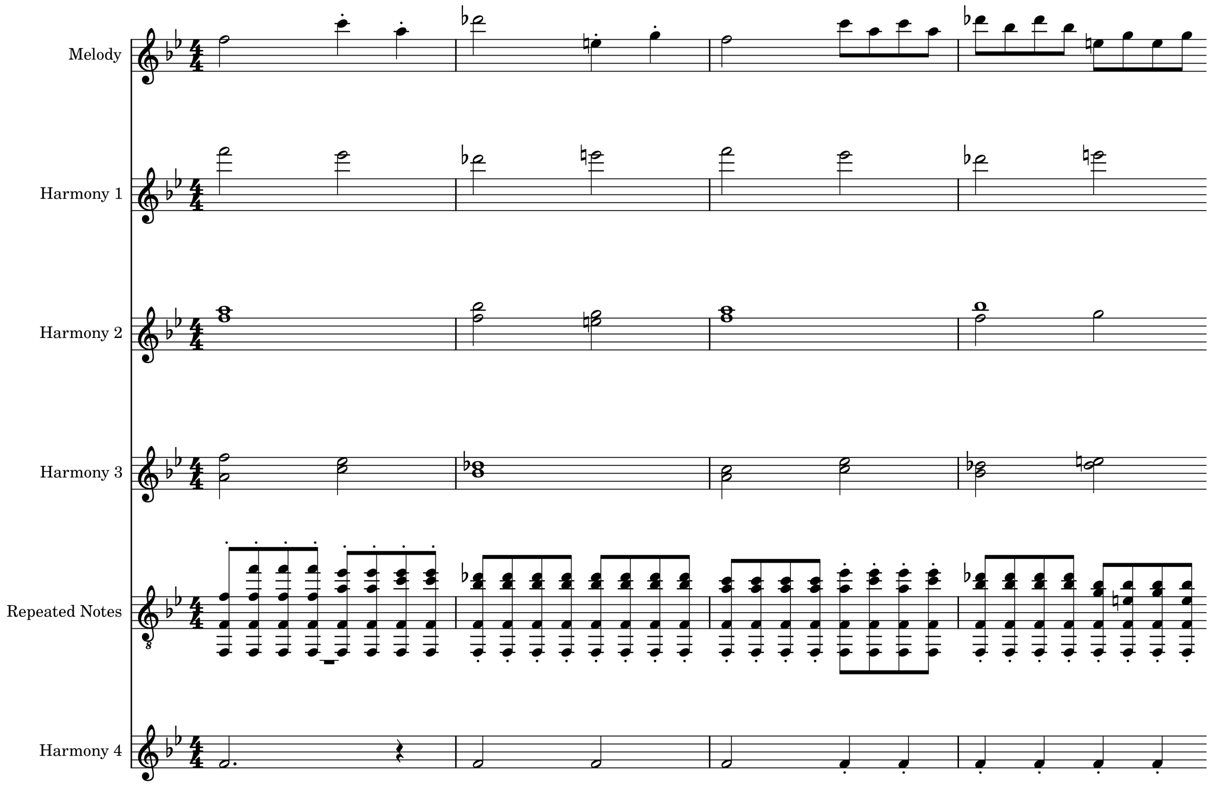


Figure B.3.: End of the first theme (measures 38-41) from the exposition of the first movement of Symphony No. 40 by Mozart, k550. Manually curated layer score. The texture of the excerpt is homophonic, with a main melody and the support of harmonic and rhythmic layers. Several harmonic layers are created starting from the annotations, even if the first three could reasonably be merged into one (non homorhythmic) layer. Octave doublings in the *Repeated Notes* layer have been kept here to highlight the subjectivity of choosing between them.

the new one starts on the same staff as the previous one. There is no control on that in our algorithm.
Many of these problems were indeed avoided when creating the example displayed in Figure 3.16 to introduce layer scores. The *Front* layer with anacruses that is passed around different instruments of the orchestra, would be constructed very poorly by our algorithm, but its relatively easy to analyze by hand. The *Rhythm* layer has octave doublings that were manually removed in the layer score. Moreover we were able to keep dynamic prescriptions that instead get lost when converting scores in MIDI format.
In order to grasp a better understanding of these problems, and to start conceiving some

possible solutions that could be put in place, we perform a manual review and curation of one of the layer scores that where produced automatically by Algorithm 1. This is the exposition of the first movement of Mozart's Symphony No. 40 (see Figure B.1, B.2, and B.3 for some excerpts). When performing curation we have decided to reattach the notes outside of layers to the layers they logically belong to, and we have ordered the layers in the staves placing melodies on the top line. However we have not deleted octave doublings, to highlight the subjectivity of this task in the examples. Another observation emerged about harmonic layers: it can be argued that some of them could be merged together. They have been separated because they are not homorhythmic, but still those sustained notes form one "*tapis sonore*" that might be put together in an abstract representation like the layer score. In Figure B.3, we have kept four harmonic layers separate, following the annotations, but merging the first three would not be surprising. These considerations on correcting layer scores should be better formalized to improve the algorithm to create layer scores and to build criteria for its evaluation.

B.3. Creating Layer Scores from Piano Scores

Layer scores can also be constructed from piano scores (Task T3). We have experimented with creating them by hand in the context of the *Angles* project with the objective to use them for the co-creation of an orchestration with humans and algorithms. The project and the preparation of the layer scores in that context has been discussed in Chapter 7. Automatizing this task of creating a layer score from a piano score is another interesting problem. This is still Task T3, that is related to piano texture analysis (Task A2) and voice identification in polyphonic scores, tasks that have not been explored in this thesis.