Automatic identification of musical schemata

Andreas Katsiavalos

Faculty of Computing Engineering and Media

22 Aug, 2022

A thesis submitted in fulfilment of the University's requirements for the Degree of Doctor of Philosophy

DE MONTFORT UNIVERSITY
LEICESTER

To the best of my knowledge I confirm that the work in this thesis is my original work undertaken for the degree of PhD in the Faculty of CEM, De Montfort University. I confirm that no material of this thesis has been submitted for any other degree or qualification at any other university. I also declare that parts of this thesis have been submitted for publications and conferences.

Abstract

This study was stimulated by the Galant musical schemata theory (GMST), an example–based learning and compositional practice that peaked in popularity around the early 18th century in Europe, suggesting a culturally–defined classification of polyphonic patterns. Under the premises of the GMST and by relating notions from psychology towards a cognitive model for musical schemata identification, an explanatory system based on music-analytical thought–patterns was examined, aiming to describe the mental processes involved in three accumulative operations: a) the schematic analysis of music notation into a stream of salient musical elements and, eventually, GMST–related musical structures, providing the standard form of music notation interpretation for the examined model; b) the example–based learning of musical schemata definitions from annotated examples, and c) the discovery of – similar to the Galant – musical schemata family–types in corpora. The proposed music–analytical model was tested with a novel computational system performing three tasks accordingly: i) search, matching representations of Galant musical schemata prototypes and examining similarity models; ii) classification, classifying segments of schematic analysis according to musical schemata family–type definitions that are extracted and maintained utilising annotated examples and pattern detection methods, and iii) polyphonic pattern extraction, examining methods that form and categorise musical schemata structures. The proposed model was evaluated employing the technological research methodology, and computational experiments quantified the performance of the computational system implementing the aforementioned tasks by utilising Galant musical schemata–annotated datasets and task–oriented performance metrics. Results show a functional cognitive model for complex music–analytical operations with polyphonic patterns, suggesting methodological explanations as to how these may be addressed by the initiate. Based on the foundations established in this project, it may in the future become possible to develop computational tools that have applications in music education and musicological research.

Acknowledgements

I would like express my gratitude to my supervisory team, Bret Battey and John Young, for their continuous support throughout this project. I would also like to thank Tom Collins for his insightful comments on numerous reviews. I should also appreciate Richard Snape for his grounded suggestions. I am very grateful to Robert Gjerdingen for sharing examples from his books in digital formats. Finally, I would like to thank De Montfort University Graduate School Office for their fine handling of my administrative issues, and Kimberlin Library, for its highly convenient services.

Andreas Katsiavalos,
Leicester, UK,
August 2022.

Related publications

Andreas Katsiavalos, Tom Collins, Bret Battey. An initial computational model for musical schemata theory. *Proceedings of the 20th annual conference of the International Society for Music Information Retrieval (ISMIR)*, Delft, The Netherlands, November 4-8, 2019.

Contents

List of Tables

List of Figures

1 Introduction

Galant musical schemata theory (henceforth, `GMST`) has its origins in the Italian compositional tradition of the mid–17th century; it peaked in popularity during the *Galant* period (approximately a hundred years later), with works such as the sonatas by Haydn and Mozart. The theory suggests example–based learning and composition with stereotypical paradigms of short polyphonic patterns, the Galant musical schemata. Each is comprised of two melodic movements (for melody and bass), temporally synchronised in stages with chord–like events. The learning of musical schemata utilises training examples, *partimenti* (Italian, sing. *partimento*), each consisting of a short sequence of musical schemata prototypes/archetypes in minimal instantiation. From these the initiate draws information regarding the musical properties of musical schemata classes, as well as their interrelation and function within phrases. In such a context, any musical work that manifests `GMST`, meaning that it can be analysed into a sequence of – similar to Galant musical schemata – musical structures, is potentially training material. Overlapping with music psychology, this exact process of reducing and organising music information into higher–level composite entities and classifications thereof is termed a 'musical schema'[1], denoting the mental 'control mechanisms' for

[1]Hence the prefix '*Galant*' in `GMST`.

the organisation of music information into categories of 'meaningful' entities, such as melodic movements, and beyond, e.g., categories of musical stream segments. Following the premises of the `GMST` and related notions from the field of Psychology, and in an effort to translate complex music–analytical operations into methodological syllogisms that are applicable by the initiate, this study examined a cognitive model for (Galant) musical schemata identification operations.

Research into automating cognitive tasks, such as classification, has increased considerably over the last few decades. It has been facilitated by a consistent increase in widely available computational power, which has enabled the application of so–called supervised and unsupervised machine learning methodologies. While the aforementioned methodologies are inherently example–based, a property that renders them eligible for the modelling of musical schemata learning and discovery tasks, their core computational approach is statistical by design. The method achieves close approximations to results, yet no explanations about the nature of the systems under examination are offered. For example, in answering the question: 'What are the thought processes involved in learning a classification of abstracted polyphonic patterns, such as the Galant musical schemata?', current machine learning methodologies, including so–called deep (artificial) neural networks and decision–trees, have nothing to suggest.

To overcome the lack of explanation of the current 'black box' approaches in machine learning and to study the analytical and learning processes of musical schemata *per se*, this study examined modelling music–analytical syllogisms: from the top–level sequence of thinking steps, down to the specificities of each individual step. On this central thesis, a high–level cognitive task, such as the discovery of musical schemata prototypes in a corpus, is analysed into a sequence of music–analytical thought–patterns, which are then implemented computationally. Such a top–level methodological approach was inspired by Bertrand Russell's logical atomism, according which 'all truths are ultimately dependent upon a layer of atomic facts, which consist either of a simple particular exhibiting a quality, or multiple simple particulars standing in a relation'[2]. The methodological view based on that premise recommends a process of analysis, a reductionist approach, whereby the analysis of the overall task into simpler and more feasible sub–tasks, such as the stages of a music–analytical syllogism, permits the definition or reconstruction of the more complex task, such as the discovery of `GMST` classifications. The main aim of this study was, therefore, to achieve the formation of a cognitive model for music–analytical operations with Galant musical schemata, gradually performing the following operations:

- The 'schematic' analysis for music notation, to reduce music notation into salient musical structures related to `GMST`, and to facilitate learning and discovery operations;
- The example–based learning of musical schemata prototypes, to create and update definitions for musical schemata family–types through annotated examples, and
- The discovery of – similar to Galant – musical schemata family–types.

The development of the examined cognitive model was facilitated by adopting the following elements from Psychology:

- Perceptual models for micro–analytical tasks when forming musical

[2]From https://plato.stanford.edu/entries/logical–atomism/

structures, utilising such notions as significance, clarity, regularity and bias;

- Hierarchical similarity models for musical schemata representations, considering the composite nature of musical schemata structure, and
- Learning modes from Developmental Psychology for the accretion, tuning, and restructuring of musical schemata prototypes.

The main objective of this study was the formalisation of the proposed cognitive model into a computational system. In such circumstance, the high–level operations of the examined model, as well as the information systems they encompass, could be tested through the following tasks accordingly:

- Search, testing the efficacy of the schematic analysis method through similarity models for musical schemata representations, also examining:
  - A representation for Galant musical schemata prototypes, and
  - A musical schemata identification process matching musical schemata representations in the schematic analysis form.
- Classification, examining the efficiency of the devised methodologies for example–based learning, and the development of:
  - A representation for the classification of Galant–like musical schemata family–types;
  - A method to integrate training examples into musical schemata family–types, and
  - A musical schemata recognition process, identifying 'learned' family–classes in schematic analysis form.
- Polyphonic pattern extraction, testing the cognitive model in its com-

plete form, also examining:

  - Thought–patterns employing previously examined methods for the representation of musical schemata structures and their identification in schematic analysis form,
  - An algorithm that forms and categorises musical schemata structures in score–wise schematic analysis representation, and
  - An algorithm that integrates the outcomes of the above score–wise analysis into the classification of discovered Galant–like family–classes.

The development of a computational system that implements and performs the aforementioned methods and tasks has enabled 'technological' evaluation procedures. Those included the creation of datasets with digital scores containing Galant musical schemata and annotations thereto, and the design of computational experiments by selecting data configurations and performance metrics for the examined tasks, such that results are produced and may be analysed.

This study contributes to existing knowledge in the fields of Computational Musicology and Music Informatics with:

- The explication of high–level music–analytical tasks related to `GMST` with thought–patterns, suggesting human–applicable methodologies for such tasks;
- The automation of music–analytical methodologies, through the computational modelling of thought–patterns in music analysis;
- The computational modelling of concepts from Developmental and Music Psychology, facilitating further experimentation, and

- Computational tools to enhance musicological research, including:
  - A library for low–level operations with symbolic music data;
  - A highly parameterised system for operations with polyphonic patterns, and
  - Datasets of digital scores with Galant musical schemata annotations.

By achieving the main aim and providing a functional music–analytical model, this study serves as a proof–of–concept for the examined music–analytical ‘thought–pattern’ approach. However, this proof comes with inconclusivity in the evidence, given the low performance of the computational implementation of the model. Despite that factor, this investigation sheds light on issues relating to the development of cognitive models for music–analytical operations, providing an informative source for future developments.

1.1 Document outline

The following chapter (Chapter 2, *Background, research context and methodology*) begins with overviews for basic notions from `GMST`, such as the Galant musical schemata prototypes, their classification, and the training–examples (*partimenti*) that are utilised by the theory for learning purposes (*see* Section 2.1.1). The next section describes the notions of ‘schemas’ in Music Psychology and ‘schemata’ in Developmental Psychology, and their relation with the aims of this study (*see* Section 2.1.2). The following section presents the goals of this study through the interdisciplinary field of Computational Musicology, connecting the examined operations with related topics from Music Theory and Analysis, Psychology and Information Theory (*see* Section 2.2). The last section reviews the ‘technological’ approach employed to address epistemological issues, and its methodological elements that enabled the design of computational experiments and the evaluation of the examined model (*see* Section 2.3).

In Chapter 3 (*A cognitive model for Galant musical schemata identification operations*), the proposed cognitive model for the identification of musical schemata is described. First, an overview of the model is given, addressing the basic set of assumptions, terminology, and representation issues (*see* Section 3.2). Then, the music–analytical thought–patterns for each one of the three high–level operations are provided: schematic analysis in Section 3.3, example–based learning in Section 3.4, and musical schemata discovery in Section 3.5. The abbreviations found in this study are listed after the Appendices.

The next three chapters (Chapters 4, 5, and 6) thus report the musical schemata identification tasks of search, classification, and discovery, and have a parallel structure. Each chapter begins with a description of the task examined and, considering the main goal of this study, its aims and objectives. Next, the computational implementations of the methods, from the proposed model, that are influential in the examined task are shown, followed by the workflow of the computational system performing the task examined.Then, the computational experiments performing the examined task are presented, including parameters and results. Each of these three chapters ends with a section analysing the results and discussing findings.

Chapter 4 (*Galant musical schemata search*) reports the task of finding Galant musical schemata representations in scores. The primary aims of this task were to evaluate the efficacy of the proposed method for schematic analysis, and the development of similarity models for musical schemata representations. The main objectives included the computational implementation of the schematic analysis methodology, and the establishment of a musical schemata identification workflow to examine similarity models for musical schemata representations (*see* Section 4.1). Next, the computational implementations for schematic analysis and musical schemata similarity models are described, followed by the the search workflow in the novel computational system (*see* Section 4.2). Then, the following section displays the computational experiments for this task (*see* Section 4.3). The chapter ends with findings and discussion regarding the examined task (*see* Section 4.4).

Chapter 5 (*Classification of Galant musical schemata*) presents the task of classifying segments in schematic analysis form with example–based learned musical schemata prototypes. The primary aim of this task was to evaluate the example–based learning method of the proposed model, while its main objectives were to achieve the computational implementation of a 'learning' algorithm that integrates training examples into a classification of Galant–like family–types (*see* Section 5.1). The next section shows the computational implementations of the aforementioned methods by the novel computational system, including a description of the classification workflow (*see* Section 5.2). Then, the subsequent section reports the computational experiments of this task (*see* Section 5.3), followed by the discussion of the findings (*see* Section 5.4).

Chapter 6 (*Discovery of Galant musical schemata*) presents the task of musical schemata discovery. The main goal of this task was completely to automate the musical schemata identification process; the main objective of doing so was to utilise and extend previous methods for the dynamic formation and categorisation of schematic reduction segments (*see* Section 6.1). The computational experiments of this task are shown in Section 6.2, and the final section of this chapter evaluates the results of the computational experiments and discusses the findings (*see* Section 6.3).

The final chapter (Chapter 7, *Conclusions*) discusses the achievement of goals and objectives, future prospects, and challenging issues of this study. Appendix A (*Examined Galant musical schemata family–classes*) presents the Galant musical schemata prototypes that were examined in this study. Appendix B (*Annotated datasets*) demonstrates overviews of the two annotated datasets developed for this study, the *Galant schemata* dataset (*see* Section B.1), and the *Keyboard sonatas schemata* dataset (*see* Section B.2). Appendix C (*Music as information*) describes how music notation is converted into algorithmically operable representations. Appendix D (*Computational implementations*) outlines the computational elements that were developed and utilised in this study. First, Section D.1 (*A collection of utilities for symbolic music data*) offers an overview of a collection for task–oriented functions with symbolic music data. Next to be presented is the novel musical schemata identification system that implements the proposed model and performs the musical schemata identification tasks (*see* Section D.2). Appendix E (*Results of computational experiments in tabular format*) displays tabular data for the result–graphs that appear in the main body of this document. Lastly, a list with all of the abbreviations found in this document is shown.

2

Background, research context and methodology

This chapter describes basic concepts from the GMST and related topics in Psychology, then to present the aims of this study within the research context of Computational Musicology. The last section outlines the methods of the 'technological' research approach employed to evaluate the proposed musical schemata identification model.

2.1 Background

2.1.1 *Galant* musical schemata theory (GMST)

In his book *Music in the Galant Style*, Gjerdingen (2007) presents *Galant* musical schemata theory (GMST) as a compositional practice that flourished during the Galant period and remained relevant during the Classical period (*circa* 1730–1820). Positioned in the early Classical period, GMST succeeded the rule–based polyphonic Baroque style (e.g., Canons and Fugues by J. S. Bach for keyboard); it suggested an example–based learning and compositional paradigm. This is a system that utilises progressions of stereotypical, and sub–phrase in length, polyphonic patterns: the Galant musi-

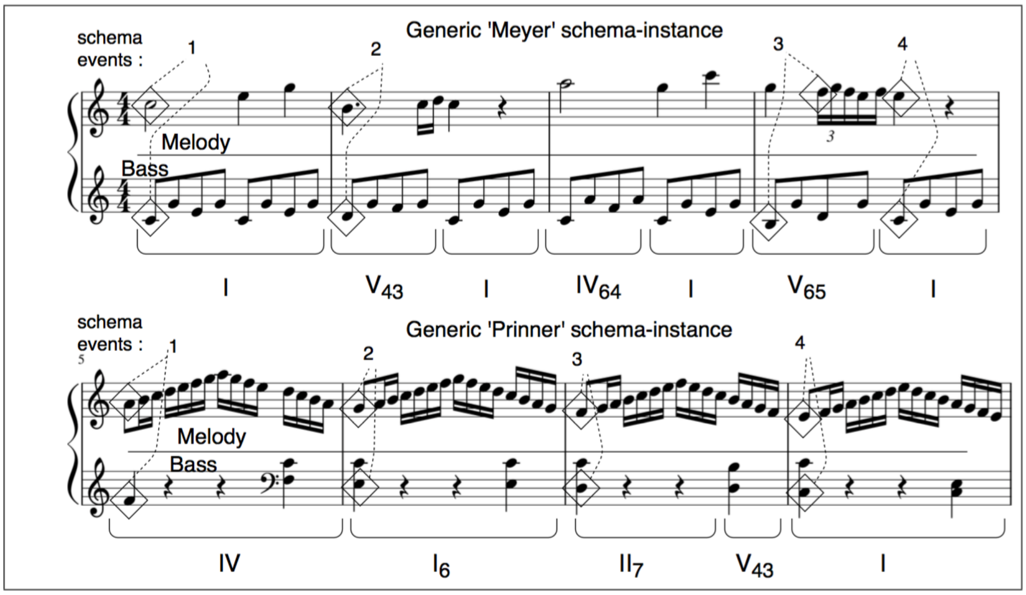


Figure 2.1: A classical phrase with Galant archetypes initiating with the ‘changing–note’ schema (the *Meyer*, Gjerdingen, 2007, *see* Appendix A.5), followed by the *Prinner* schema (*see* Appendix A.7). Note how schema–stages are delimited by harmonic regions, and the interposition of schema–events: in the last beat of measure 7, and the more extended between the second and third schema–events of the *Meyer*, from the second half of the second measure and including the third (Piano Sonata No.16 in C major, K.545 ‘Semplice’, I. Allegro, Mozart, Wolfgang Amadeus, 1788).

cal schemata prototypes (henceforth, Galant archetypes). Galant archetypes represent an archetypal/generic form for family–types of collections with variants. Gjerdingen[1] documented more than 14 types of Galant archetypes (*see* Appendix A); he named them, as he stated, following in the footsteps of Joseph Riepel, occasionally using Italian terms that ‘capture an aspect of their function’. The popularity of `GMST` correlates with the increasing popularity of the relatively new instrument at the time, the pianoforte (now piano); many works written for that instrument, such as sonatas, manifested

[1]Gjerdingen (2007), Appendix A, p.453.

`GMST` (*see* Figure 2.1).

2.1.1.1 The *Galant* archetype

A Galant archetype has a static feature–set consisting of two melodic movements, for melody and bass, and the properties of the implied *schema–event* progression from their temporal synchronisation, a sequence of chord–like structures. Gjerdingen (2007) suggests a representational formalism for Galant archetypes as sequences of schema–events (*see* Figure 2.2). Each schema–event incorporates qualities such as:

- Two melodic movements, for the outer–voices of melody and bass, represented in scale–degree contextualisation;
- The harmonic content of each schema–event, as intervals from bass, and
- The combined metric strength of the notes in a schema–event, derived from the metric context of a given time–signature.

In an attempt to categorise the kinds of information present in a Galant archetype, the following types were considered:

- **Structure–related**, regarding the number of schema–events, and their integrity, concerning the presence of the qualities that constitute a schema–event;
- **Content–related**, concerning primarily the pitch–related information in schema–events, such as:
  - The scale degrees in melody and bass movements (*see* Figure 2.2, $\mathtt{m}_\mathrm{i}$ and $\mathtt{b}_\mathrm{i}$);
  - The intervals from bass for each *schema–event*, (*see* Figure 2.2,

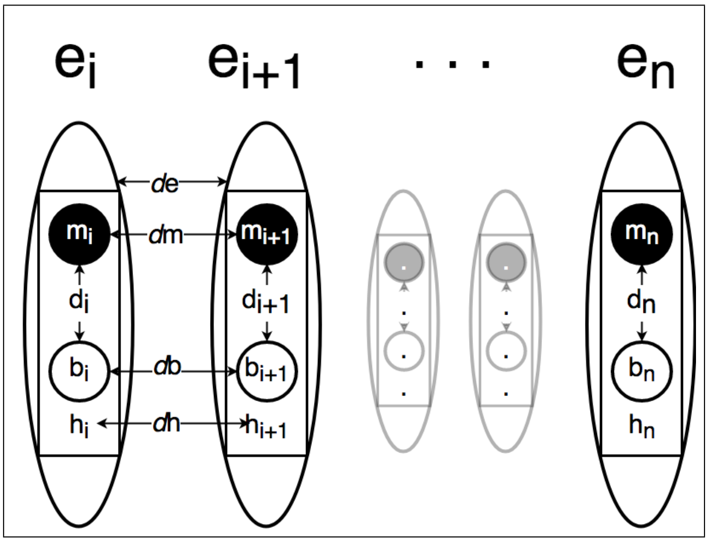


Figure 2.2: Generic Galant musical schemata representation as a progression of schema–events. The duration of a *schema–event*, termed 'inter–voice intra–event' temporal interval (eIVIE) is the distance between two notes of the same *schema–event*, one from the melody and one from the bass movement (d_i). The distance between two adjacent notes of the same *schema–voice*, termed the 'intra–voice inter–event' temporal interval (vIVIE), is also considered as the temporal interval between two adjacent *schema–events* (*d*e). Each melodic movement may be considered as a sequence of directed pitch–intervals (*d*m and *d*b).

h_i);

- The overall metric strength for complete *schema–event*s, (binary, strong/weak), and
- Intra–archetype temporal–relations, including:
  - The intervals between the notes of the same *schema–event* (*see* Figure 2.2 d_i, termed the 'inter–voice intra–event' interval or the eIVIE), and
  - The intervals between adjacent notes of the same *schema–voice* (*see* Figure 2.2 *d*m and *d*b, termed the 'intra–voice inter–event' interval, or the vIVIE).

- **Context–related**, concerning the inter–relations of archetypes and their phrase–related properties, i.e., position, and function.

Utilising the above properties, each Galant archetype may be expressed by profiles. When 'realising' archetypes in music notation, these profiles 'instantiate' their values in the pitch and temporal domains. These properties are at the centre of this study and are detailed when presenting the novel model in Chapter Three, next.

The composite musical structure of (Galant) musical schemata (archetypes), as Temperley (2001) notes, requires knowledge of 'infrastructural features', such as harmonic and key structure, as well as the 'metrically–parallel' grouping of events, adding (Temperley, 2001, 12.3, p.336.) that:

> [A] schema is an entity defined as a cluster of features; not all features may be necessary to the schema (in some cases, no single feature is necessary), but a certain combination of features is generally typical.

Thus, the flexibility of a Galant archetype as a formalism for musical knowledge representation relies in its composite feature–set (i.e., schema–stages, –events, and –voices), and the dynamic constraints upon the variables that express them (e.g., sets of permissible scale–degrees in a schema–event).

2.1.1.2 Learning Galant archetypes

The classification of Galant archetypes is learned by the initiate through considering short and minimally instantiated excerpts of music notation (*partimenti*, *see* Figure 2.3) or, more generally, from annotated works.

When learning Galant archetypes from *partimenti*, all properties of the instantiated Galant archetypes (i.e., structure–, content–, and context–related information) are available through annotations. In this case, creating profiles from the 'instantiated' Galant archetypes is a straightforward process. Even in the case where only the sequence of archetypes is given (i.e., not

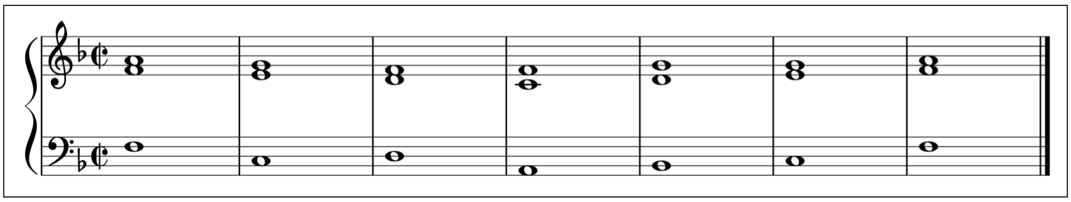


Figure 2.3: A *partimento* from Giacomo Tritto (1733–1824) starting with the *Leaping–Bass Romanesca* schema (Gjerdingen, 2007, p.26, also *see* Appendix A.1) ending with a *Clausula* (*see* Appendix A.9). *Partimenti* are small excerpts of notated music with usually short progressions of basic tonal and rhythmic instantiations of musical schemata prototypes. The goal of *partimenti* is pedagogical, aiming to give primary/essential information about musical schemata prototypes to the initiate.

the exact segments of each archetype instantiation), that would require a segmentation task to define structure–related information (i.e., the number of schema–events for each annotated archetype). Assuming that *partimenti* provide exact definitions for archetypes, structural information ambiguities can be resolved with exposure to only a few diverse examples.

The main issue when learning archetype definitions from *partimenti* concerns the addressing of variability. As a cultural product, Galant archetypes diverge to varying degrees, and multiple definitions may exist for the same Galant archetype. This variability may be expressed as differentiations in structure– and content–related information. Structure–related variability examines the difference in quantity and quality of schema–events. For example, the three archetypes of the *Romanesca* family–type (*see* Appendix A.1) have quantitative structural differences: *Galant* variant has four schema–events; *Leaping–* and *Stepwise–Bass* variants both have six. In addition, *Leaping–* and *Stepwise–Bass* vary in content–related information, and in the bass melodic movement in particular. In a similar manner, the variability in Galant archetype definitions may be expressed as differentiations among structural– and content–related information.

When learning Galant archetypes from annotated works, schema–events usually appear compositionally elaborated, in what Gjerdingen (2007) refers to as *schema–stages* (*see* Figure 2.1). In *partimenti*, schema–stages are usually not apparent, due to the lack of elaborations and the minimal instantiation of the archetypes. In works, schema–stages tend to constitute the norm, encompassing multiple instances of the same and/or variant/ornament schema–events. Thus, learning Galant archetypes from works requires a method of reducing their elements into manœuvrable entities, transforming (possible) schema–stages into schema–events, and facilitating learning with a *partimento* representation of music notation.

The modelling of the learning aspects of the `GMST` includes the development of methods that address the issues described in this section. The first musical schemata identification operation examined in this study performs ‘schematic’ analysis, reducing music notation into *partimenti* form (*see* Section 2.2.2.1). The learning of Galant archetypes from *partimenti* was examined with the example–based learning operation (*see* Section 2.2.2.2).

2.1.1.3 Summary

This part has presented the basic elements of the `GMST`, including the musical structure of the Galant archetype and the information it conveys, and the learning process with *partimenti*. A Galant musical schema is a composite musical structure that may be addressed under a set of pitch– and temporal–related constraints for relationships between elements of explicit structure: a pair of melodic movements, a sequence of schema–events or, more generally, a set of high–level music–analytical structures with a set of conditional relations. Considering the comments of Narmour (1991) on parametric complexes within the Tonal style:

> Schemata exist on all levels, from highly abstract, generic categories, relational families, and prototypes (e.g., the known forms of music) to more concrete configurations (e.g., common tonal schemes, as in Schenker's various *Brechungen* structuring the *Ursatz*), to highly specific instantiations. ... Schemata range from highly instantiated parametric complexes within a style ... to extremely generalized structurings of the elementary materials of a style.

2.1.2 Schemata in Psychology

Considering the properties of the `GMST` and the aims of this study, the present section demonstrates how notions from Psychology, including *schemas* and *schemata* in both Music and in Developmental Psychology are related to this study.

2.1.2.1 Music memory model

In his book *Memory and Music* (Snyder, 2000), Snyder provides a description of a basic three–part model of memory (*see* Figure 2.4) and a three–level description of musical structure, the 'musical mind infrastructure', based on the Atkinson–Shiffrin memory model (Atkinson and Shiffrin, 1968). These are (*see* Figure 2.4, reading upwards):

a) Echoic, feature extraction/perceptual binding (perceptual categorisation);
b) Short–term memory, episodic, and
c) Long–term memory, conceptual categories.

Echoic memory handles auditory sensory data that are less than a second in duration; it is responsible for the extraction of acoustic features and the perceptual binding for the creation of auditory events, meaning that information is no longer continuous and has been categorised. Snyder (2000) states that '[F]eature extraction and perceptual binding together constitute what Gerald Edelman (1989, 1992) has referred to as "perceptual categorization"'. Short–term memory, also called *working* memory, is usually less than a minute in duration and is responsible for control processes and the regulation of information flow (Atkinson and Shiffrin, 1971). When music information passes the level of echoic and short–term memory, it then has the potential for 'permanent' memory storage, with mechanisms described as 'schemas' and 'metaphors' forming conceptual categories in the 'long–term memory'. Considering the above memory architecture, this study regarded the three levels of *sensory*, *short–* (also, *working*), and *long–term* memory to be relevant. Since this study examines methods towards score compre-

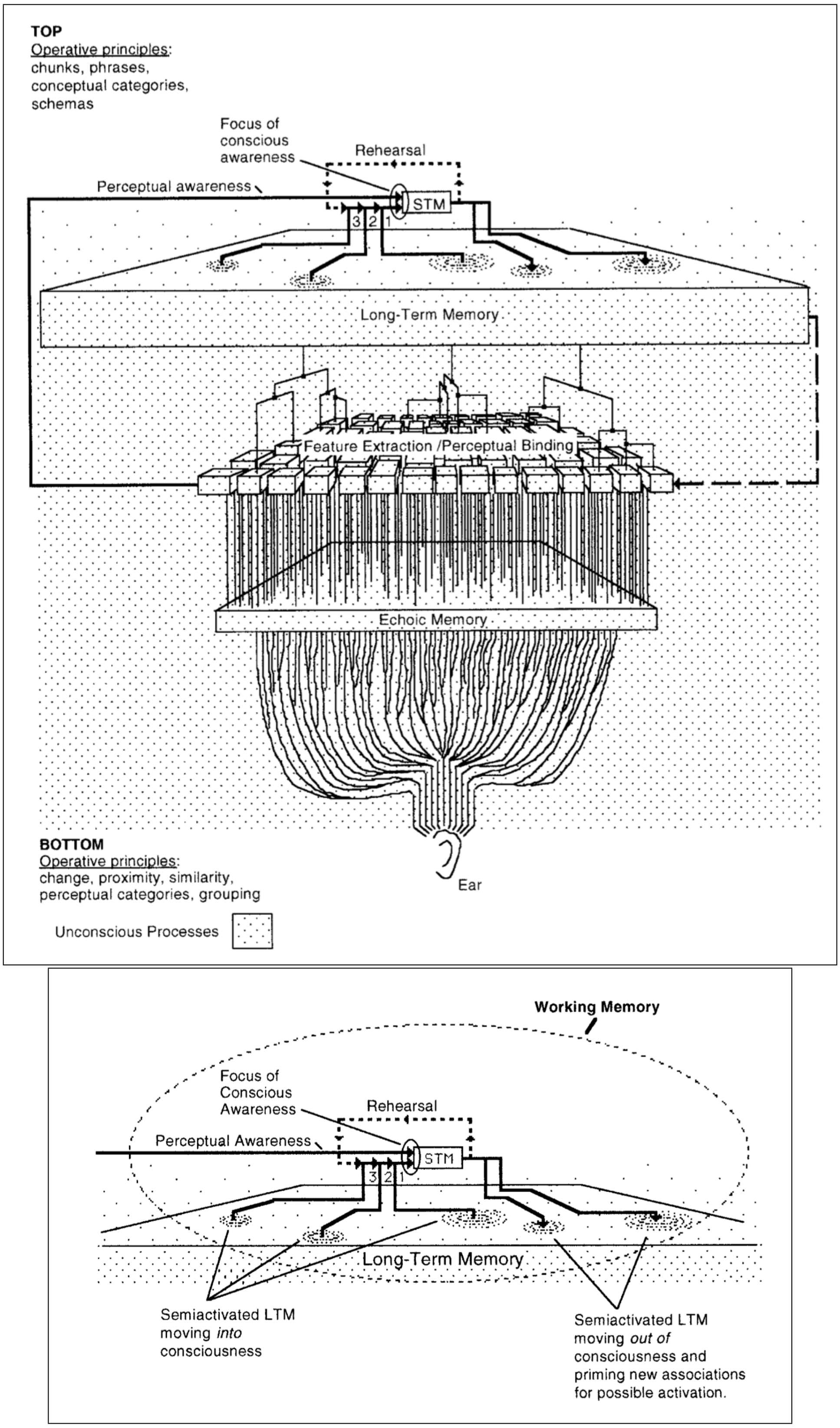


Figure 2.4: Auditory memory diagram (top) (p.6) and working memory (p.49) in detail (bottom) from Snyder (2000).

hension, *sensory* memory and its subsequent psychoacoustics were modelled with perceptive principles (*see* Section 2.2.2.1). The identification of Galant musical schemata occurs in the middle layer of the memory model, the 'working' memory level, and (Galant) musical schemata (archetypes) reside in the 'long–term' memory.

Although the above descriptions seem decisive, as Snyder (2000) suggests, the existence of psychological universals is debatable, since no concrete evidence is provided in answer to the question: 'how far do these operations extend?'. Further to elaborate:

> One of the features of cognitive psychology is that most of its constructs are theoretical – their existence is inferred indirectly through experiment. No one has ever seen an echoic memory or a schema; rather, these theoretical constructs have been created to explain and predict aspects of people's behavior. They are what Ronald Langacker has called "convenient reifications" (Langacker, 1987, p.100). As such, there are different theoretical perspectives on these entities, and even on what the relevant entities are'.

2.1.2.2 Bi–directional music interpretation

Human memory is by definition related to the learning of musical schemata. More specifically, when interpreting music information in the working memory, two procedures occur and interact simultaneously: one based on the perceptual organisation of the music flow, the so–called 'bottom–up' approach, and another based on cognitive operations that consider information stored in memory, the 'top–down' approaches in information processing. The bottom–up approach involves inductive reasoning and utilises elements of a lower abstraction level to reach higher–abstractions (e.g., finding harmonic segments aggregating notes into groups). Research within music psychology (Deutsch, 1975; Povel and Essens, 1985; Bregman, 1994; Krumhansl, 1997; Deutsch, 1999; Huron, 2001) has identified various factors that contribute to the construction of note groups; these facilitate the formalisation of the binding forces that enable the formation of complex musical entities, such as the Galant–related musical structures. In the bottom–up approach, grouping mechanisms initiate from note elements and, applying a set of principles that consider pitch and temporal properties, either support or hinder their merging into groups (*see* Figure 2.5).

'Top–down' processing regards deductive reasoning and considers pre–existing higher–abstractions (or biases on 'elements of interest') when processing elements of lower abstraction levels (e.g., finding harmonic segments considering pre–existing knowledge about their properties, such as three voice harmonies made of intervals of thirds). As Narmour (1991) mentions:

> The top–down system is flexible, variable, and empirically driven. In it, the listeners constructively match and compare representative schemata to current input. ... In contrast, the bottom–up mode constitutes an automatic, unconscious, preprogrammed, 'brute force' system that operates on parametric primitives.

Summarising:

- Music interpretation based on perceptual principles considers:
  - The temporal (extended) 'now';

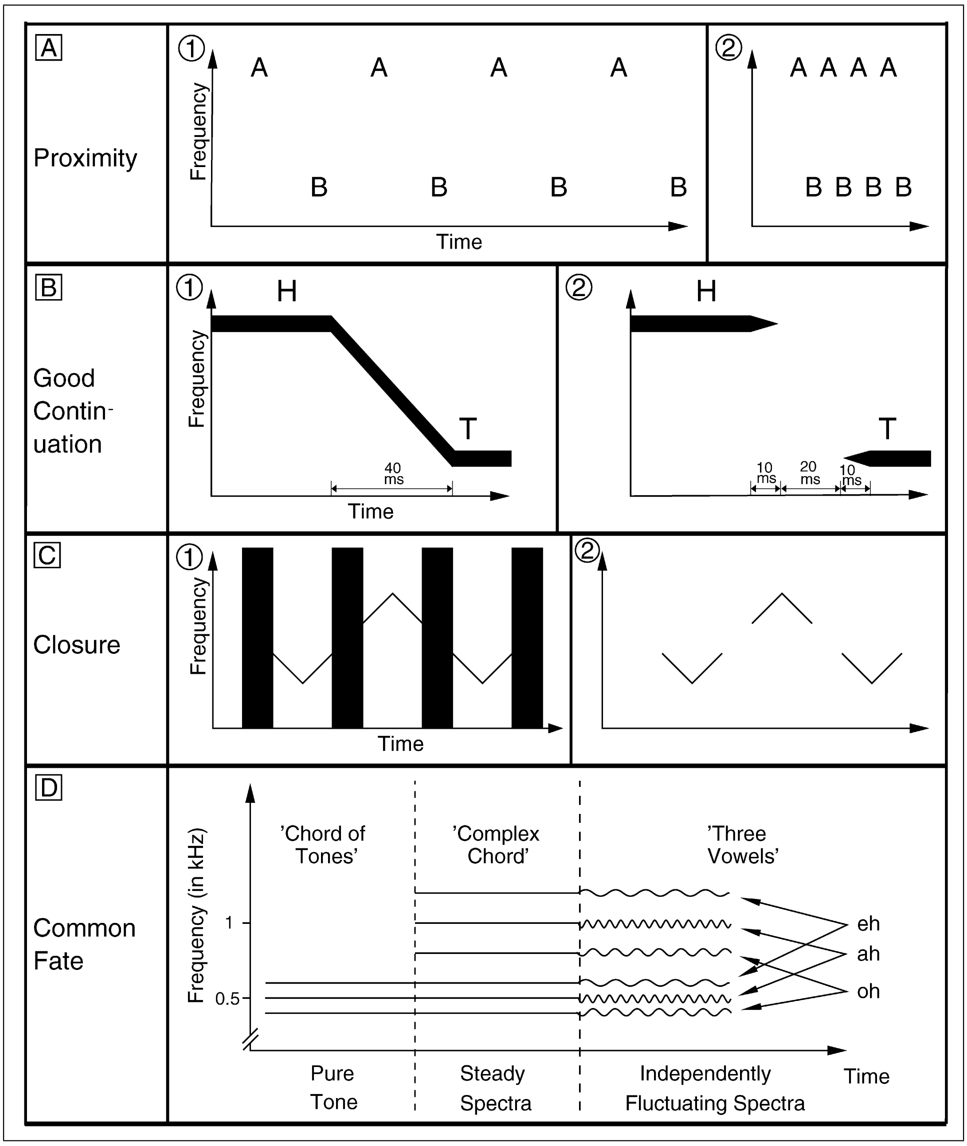


Figure 2.5: Grouping mechanisms according to Bregman (1994) (image from Purwins et al., 2008). A. Proximity, B. Good-continuation, C. Closure, and D. Common Fate.

  - Bottom–up information flow;
  - Data–driven processing;
  - Deductive reasoning, and
  - Gestalt principles for grouping and segmentation of music elements.
- Music interpretation based on memory considers:
  - (personalised) Past experiences and future expectations;
  - Top–down information flow;
  - Memory–driven processing;
  - Inductive reasoning, and
  - Cognitive operations based on schema theory, analogies and metaphors.

In this study, the bi–directional music interpretation methods outlined in this part were utilised to form a music information processing level, where the various types of musical structures are grouped and compared, considering Galant archetypes.

2.1.2.3 *Schemas* in Music Psychology

Bregman (1994) characterises *schemas* as 'control structures' of perception and cognition applied in auditory scene analysis. They are responsible for the organisation of the auditory information into the notion of *auditory streams*; in parallel with visual scene analysis, it identifies the musical 'objects' that are present within a given temporal frame. The notion of auditory streams, as well as the perceptual mechanisms that extract them, have been utilised in the music analysis of digital scores for the identification of 'musical streams'. Initial studies into the matter (Huron, 2001) attempted to systematise the extraction of monophonic voices with the formalisation of voice–leading principles from music theory and Gestalt psychology (Köhler, 1967). Later studies (Cambouropoulos, 2008) focused on 'stream segregation', and the extraction of parallel streams with varying textures, such as monophonic and homophonic. Musical streams are related to Galant musical schemata archetypes, as each musical schema instantiation may be described as a musical stream segment with conditional melodic and harmonic content. Thus, the identifying of the 'control mechanisms' that organise music notation into musical stream segments may facilitate the modelling of the 'schematic' analysis operation.

2.1.2.4 The notion of *musical surface*

The notion of 'musical surface' has been described both implicitly and explicitly in musicological and music–psychology literature, with a general meaning of a systematic 'interpretation' level or layer of musical information in which certain musical elements are identified. Lerdahl and Jackendoff (1985), adapting the term from linguistics, use the term 'musical surface' to denote the elements of the 'shallowest depth' in music representations, that is, the notated music information. In a similar manner, Wiggins (2010) used the term to describe the minimal elements of a music representation, the 'atoms' of an encoding format. For example, in so–called piano–roll representations, commonly extracted from industry–standard Musical Instrument Digital Interface (MIDI) protocol[2] encodings, the 'musical surface', according to the aforementioned views, would refer to the set of note elements that are extracted by tracking MIDI `note-ON` and `note-OFF` events.

[2]https://www.midi.org/specifications

Other studies (Deliège et al., 1996; Cambouropoulos, 2010) suggested that the term 'musical surface' should express the outcomes of a universal perceptual and cognitive processing of music; a high–level interpretation of the score that yields mental structures from music information. Answering the question: 'what are the minimum perceived as wholes?', Cambouropoulos (2010) states:

> [A] listener does not extract a geometric representation from the acoustic input; a listener organises the acoustic continuum into musical streams (Bregman, 1990) and encodes the music into a representation that is closer to strings, i.e. sequences of musical events (such as [a] sequence of notes or chords). ... [I]t is, herein, suggested that the musical surface comprises of (complex) musical events perceived as wholes within coherent musical streams — the musical surface is not merely a sequence of atomic note events.

The identification of the 'musical events' that are perceived as 'wholes within coherent musical streams' is related to the identification of the musical structures of a Galant archetype, the structures of *schema–voices*, *schema–stages*, and *schema–events*. Thus, the identification of the Galant 'musical surface' may be achieved by modelling those 'control mechanisms' in working memory with bottom–up and top–town processing of musical structures related to Galant archetypes.

2.1.2.5 Schemata in Cognitive Psychology

Schema theory has its origins in Developmental Psychology, describing the construction of mental models of the world. The concept of a schema was introduced by Bartlett (1932) as part of his learning theory, with a schema being 'an active organization of past reactions [or] experiences, which must always be supposed to be operating in any well–adapted organic response'. The term 'schema' was introduced by Piaget in the French version of *The Origins of Intelligence in Children* (1952) in 1936 to explain childhood development.

More generally, Snyder (2000) states:

> Schemas are based on what similar situations have in common; because no two situations are ever exactly alike, schemas must be somewhat flexible: their elements, that is, the categories of objects, single events, actions within a scene or event, are variable within certain limits.

As a form of knowledge interpretation mechanism, there are many types of schemata that enable the understanding of objects, persons, social issues, self–awareness and events. Evans (1967) states that:

> A schema is a characteristic of some population of objects. It is a set of rules which would serve as instructions for producing (in essential aspects) a population prototype and object typical of the population. Schematic (or constraint) redundancy (Evans, 1967a) is a measure of the extent to which individual members of the population adhere to the schema rules. A schema family is a population of objects, all of which can be efficiently described by the same schema rules.

Moreover, Rumelhart et al. (1985) suggest that 'Schema theory describes a mechanism for the creation of "internal" knowledge representation and retrieval'. Rumelhart (1978) described schemata as the 'building blocks of cognition' and suggested that schemata operations involve both top–down and bottom–up information processing. In the same work, Rumelhart (1978) singled out six properties of schemata for special mention:

1. Schemata have variables; since no experience is ever exactly repeated, we must be able to discover intuitively both the dimensions of variation and the range of variation that characterise our generalisations of the world;
2. Schemata can embed, one within another: a particular schema may be part of a larger network of relationships;
3. Schemata represent knowledge at all levels of abstraction;
4. Schemata represent knowledge rather than definitions. In Rumelhart's words, 'our schemata are our knowledge. All of our generic knowledge is embedded in schemata';
5. Schemata are active processes; through schemata we can make predictions and form expectations, and
6. Schemata are recognition devices of which the processing is aimed at the evaluation of their appropriacy of fit to the data being processed.

Considering the properties of the `GMST` (*see* Section 2.1.1), the above descriptions of a schema may well apply to Galant musical schemata family–types and their archetypes, suggesting methodological approaches.

Regarding the acquisition of knowledge, Piaget (1952) viewed intellectual growth as a process of adaptation (adjustment) to the world through:

- Assimilation, where new information 'is incorporated into pre–existing schemas';

- Accommodation, where 'existing schemas might be altered or new schemas might be formed as a person learns new information and has new experiences', and
- Equilibration, where an inner sense of balance between new and old schemas is achieved.

Accordingly, Rumelhart and Norman (1976) presented three modes of learning, namely, *accretion*, *tuning* and *restructuring*; these allow for the *consolidation* and *adaptation* of known schemata, and the *accommodation* of new ones, respectively. These learning methods guided the modelling of example–based learning and discovery of musical schemata family–types.

2.1.2.6 Summary

Schemas from Music Psychology are regarded in this study as the perceptual and cognitive 'control mechanisms' that integrate and abstract music information forming the (Galant) 'musical surface'. These mechanisms occur in the 'working' memory layer, involving the so–called bottom–up and top–down processing of music information, thus enabling the identification of musical structures by considering both low–level perceptual grouping principles and high–level cognitive biases relating to known information stored in the long–term memory. The processes of specifying and updating those long–term schemata and, in this case, the Galant musical schemata archetypes, are considered to be the three modes of schemata learning proposed in developmental psychology, namely, *accretion*, *tuning* and *restructuring.* These notions guided the development of the cognitive model for the identification of musical schemata, and their specific contribution becomes apparent in the next chapter (*see* Chapter 3), where the proposed model is analysed.

2.2 Research context

This part describes how the aims and goals of this study may be addressed in the interdisciplinary research field of Computational Musicology. First, the relation of the examined cognitive model with Symbolic AI systems is discussed, also because the latter is utilised to translate the examined operations. Then, each of the three high–level musical schemata identification operations of schematic analysis, example–based learning, and discovery is analysed, presenting their main challenges and how these have been or may be addressed in Music Theory and Analysis, Psychology, and Information Theory.

2.2.1 Approach in modelling cognitive tasks

Given that the `GMST` is the subject of both the thesis and the system, the main goal of this study concerns the development of a cognitive model for identification operations with Galant musical schemata archetypes. Reviewing research approaches into the development of such models, Snyder (2000) stated:

> [T]here is a conflict in cognitive psychology between two paradigms. One is an older 'classical' information–processing paradigm, which grows out of metaphors from serial digital computing and information theory. The second is a newer connectionist paradigm, which comes from models of the nervous system based more on

parallel computing.

Aiming to explicate on the mental processes involved when identifying Galant musical schemata archetypes, the examined (cognitive) model followed the former approach. However, the music–analytical thought–patterns constituted the information–processing paradigm of the model, rather than applying metaphors from algorithmic thinking. The difference between those two approaches when modelling (cognitive) information processing is addressable. Utilising applied information processing methodologies and systems (e.g., Markov models), the research focus tends to shift towards the presentation of the examined (cognitive) task to the selected method, subsequently narrowing experimentation options. In contrast, the modelling of information–processing with domain–specific high–level operations, such as music–analytical thought–patterns, although reliant on a layer of 'computational infrastructure' to perform complex information processing, enables further experimentation within the examined domains, maintaining focus on the high–level operations.

Considering the aforementioned approach, the examined cognitive model falls into the category of so–called Symbolic AI systems, modelling human intelligence by the simulation of human–thought. This approach was seemingly the natural way for the computational formalisation of human intellect for the most of the second half of the 20$^{\text{th}}$ century (Haugeland, 1989), as it was also the only widely available.

Common practices that stemmed from research into Symbolic AI systems suggest the representational separation between the 'world' model, knowledge, and reasoning. The 'world' model is an ontology for the kinds of information structures that may be formed and addressed in the examined information environment by the other parts. Knowledge in those systems is represented by collections of predicates and rules, concerning properties and relations for elements from the ‘world’ model, even for composite structures of such elements. ‘Intelligent’ behaviour of such systems is exhibited when a ‘situation’ from the ‘world’ model is to be considered. At such a juncture, reasoning occurs through the deduction of the examined situation: by applying first–order logic through the world–view of the then knowledge–base to draw inferences about the validation of the situation. The knowledge–base of such system was punctiliously built by experts, hence the term ‘expert systems’.

The characteristics of those ‘expert systems’ found in Symbolic AI systems are inherent properties of the `GMST`, though veiled by their musical nature. In a `GMST` ‘expert system’, therefore, *partimenti* are the ‘world–view’ model, the music ontology of two contextualised melodic movements and their implied sequence of schema–events. The Galant archetypes are the knowledge–base, represented by a set of rules and facts concerning their properties (i.e., structural–, content–, and context–related information); subsequently, reasoning is performed through similarity operations between representations of musical schemata prototypes and score interpretations in schematic analysis representation, the ‘world–view’ of the examined model. By utilising this representational separation from Symbolic AI, the examined cognitive model is simulating human thought in music–analytical operations with Galant musical schemata archetypes (*see* Chapter 3).

2.2.2 Musical schemata identification operations

In this systemic approach to music understanding (Witten and Conklin, 1990) with application of music analysis *via* the cognitive model (Wiggins, 2010), the research context of this study was formed through musical–analytical methodologies for the representation, extraction, transformation, and comparison of musical structures related to the `GMST`. These drew from Music Theory and Analysis, Psychology, and Information Theory, in the inter–disciplinary field of Computational Musicology. Considering a 'Galant expert system' architecture for the examined cognitive model, the following sections present each of the three high–level musical schemata identification operations, i.e., schematic analysis, example–based learning, and archetype discovery, highlighting challenging issues and reviewing approaches from the literature.

2.2.2.1 The operation of schematic analysis

In this study, the goal of the 'schematic' analysis operation (henceforth, `SA`) is to regress music notation into a sequence of schema–events, similar to a *partimento* (*see* Section 2.1.1). The `SA` involves the transformation of music notation through the:

- Reduction, filtering and selecting notes;
- Segmentation, grouping notes into schematic–elements (i.e., melody and bass movements, and schema–event sequences), and
- Similarity, comparing schematic–elements and combinations thereof within the scope of the examined music notation (to identify repetitions/parallelism and aid segmentation) and between stored archetype

representations.

The SA operation simulates the working memory, modelling the 'control mechanisms' that create the Galant 'music surface'. Thus, the above processes are applied simultaneously to create the 'world–view' of the examined model, the information environment of two melodic movements and their implied schema–event sequence. This decomposition process has been examined extensively in Music Analysis and Computational Musicology, also under different intentions and depths. The rest of the current section reviews methods from the literature that correlate with the goal of schematic analysis, summarising issues and best practices.

Two music–analytical theories that relate with the goals of the SA operation are the *Schenkerian* analysis (Forte and Gilbert, 1982), and *The Generative Theory of Tonal Music* (GTTM, by Lerdahl and Jackendoff, 1985). Schenkerian analysis suggests an elaborated analytical methodology for primarily melodic movements. Utilising the notions of 'background' and 'foreground' to express the transformation of a basic (musical) idea to the realisation of embellished music notation, the structure of a piece is considered as constituting rules for melodic movement around the tonic triad. The theory suggests methods for prolongation representation, melodic diminutions, such as arpeggiation and passing notes, and rhythmic reduction through a counterpoint model (the modes of Cantus Firmus[3]). Those methods of melodic reduction have been utilised in the modelling of the SA operation, and particularly when selecting notes for a schema—voice (*see* Section 3.3.3). Schenkerian analysis was considered to be more appropriate for greater mor-

[3]Cantus Firmus is the pre-existing melody forming the basis of a polyphonic composition, where additional voices are using the same, or smaller durations, i.e., the modes of the Cantus Firmus.

phological entities, such as complete parts and works, and was only partially utilised in current discussions, since schematic analysis concerns sub–phrase–level temporal spans.

The GTTM formalises the processes of reduction and segmentation through a rule–lbased system, suggesting four levels of music structure, including: a) grouping, b) metrical, c) time–span reduction, and d) prolongational reduction, based on principles for the transformation between surface (externalisations) and deeper (internal) music structures. The approach of the GTTM is highly relevant with the identification of musical schemata prototypes, and selected features are elaborated when presenting the *schematic* analysis method (*see* Section 3.3).

As stated earlier, the operation of `SA` models the working memory, where those 'control mechanisms' facilitate the simultaneous application of both bottom–up and top–down processing in grouping mechanisms that reduce, segment, and compare music information. Therefore, this study considered the development of processes for the extraction of the Galant 'musical surface' with grouping mechanisms that:

- Initiating from note elements, apply a set of principles that consider pitch and temporal properties (*see* Figure 2.5), and that either support or hinder their merging in groups (Section 3.3.3.1 describes an additional set of notions such as 'significance', 'clarity', and 'persistence', and how these may be formalised for such extraction of schema–voices and –events);
- Considering stored/'known' schematic elements from memory, impose grouping preferences on structure— and content—related properties

(Section 3.2.2.2 describes how 'bias' towards the identification of cognitively selected elements of interest was formalised).

For example, when performing `SA` and elements in memory are to be considered, these may control the selection of 'valid' schematic elements by excluding those that do not match collective properties from the archetypes, such as a structural limit (bias) on the minimum number of schema–events. Bias form and intensity may be achieved with combinations of structural– and content–filtering thresholds providing a method to express top–down processing. As the model performs more autonomous operations, the bias options elaborate.

The notion of similarity is omnipresent in this study. Regarding the operation of `SA` in particular, two kinds of comparisons are performed:

1. When filtering content based on stored archetypes (the top–down bias described above), and
2. When identifying intra–score patterns (schema–voices or schema–event sequences).

In the first case, the properties of an archetype may be combined into a threshold to form a filtering profile to be applied when forming schematic elements. For example, in the task of Galant musical schemata search (*see* Chapter 4), the case of 'exact–matching' is the one extreme in applying maximum bias. When identifying repeated schematic–elements within the scope of the examined music notation (represented as a schema–event sequence), finding repetitions becomes a sub–string problem. Thus, the similarity models for Galant archetypes are comparing sequences of schema–events.

Since archetypes are composite entities, similarity models that compare them should include metrics and thresholds for each quality. Metrics for sequences include the Hamming distance, counting the binary differences of elements that appear on the same position in equal–length sequences. Another popular similarity approach is the edit–distance, where the number of edit–operations (such as alteration, addition and removal) that are required to ‘equalise’ two strings are counted and then allotted thresholds. Such approaches may be utilised for the quantification of the qualitative differences among musical schemata representations.

In Computational Musicology, there are numerous approaches modelling various kinds of reduction, segmentation, and similarity operations. Works directly related to this study and their contributions are shown when presenting the computational implementations of the examined model, in the second sections of each of the three ‘task’ Chapters (4, 5, and 6).

Summarising, the operation of schematic analysis provides the information environment for the cognitive model by modelling the control mechanisms that create the Galant ‘musical–surface’, represented by a sequence of schema–events. The operation of `SA` utilises methods for reduction, segmentation and similarity from Music Analysis and Psychology, facilitating the development of grouping mechanisms that model the working memory. The examined method for schematic analysis is shown in Section 3.3.

2.2.2.2 The operation of (Galant archetype) example–based learning

The operation of the example–based learning of (Galant musical schemata) archetypes (henceforth, `XL`) concerns the creation and update of archetype definitions from *partimenti*, or more generally, from annotated segments in schematic analysis form. In the `XL` operation, each (labelled) class aggregates sequences of schema–events (the outcomes of `SA`). This means that the information environment offered by the `SA` operation, i.e., a sequence of schema–events, culminates into archetype representations. Thus, the `XL` operation performs inductive reasoning, inferring an archetypal form for a population of training exemplars by identifying commonalities and differences among their structure– and content–related properties. This part discusses the issues of knowledge representation and update for Galant archetypes.

The `GMST` is a rare exception in Music Theory offering an inherently example–based learning methodology and archetypal representations (*see* Section 2.1.1) for its categories. Thus, a Galant archetype is considered to be a sequence of schema–events, with permissible variations regarding its structural– and content–related properties. Considering the knowledge–representation paradigm from Symbolic AI approaches, each archetype may then be expressed by a set of rules and predicates concerning its properties (structure– and content–related, *see* Section 2.1.1.1).

In the training case where all of the training material of an archetype is available (a batch–processing method), pattern detection algorithms may be applied to achieve the extraction of archetype representations. In an incremental learning workflow where training examples are given individually and when a training example is to be considered against the model, two cases may be: the training information may, or may not, be known to the archetype. In the former case, a quality of ‘concentration’ arises from frequency of appearance for specific properties, and should be considered.

In the latter case, the difference between known definitions and new information should be expressed by the archetype, or a new variant be created. Both of these cases may be addressed by the three learning methods from schemata (*see* Section 2.1.2). Whenever training information is repeated, the process of 'consolidation' of a particular case of the archetype may be modelled. Accordingly, when training information differs from the archetypes, the processes of tuning and restructuring may also be modelled to express the inclusion of variants from the archetypal form, as in the former case, or the creation of variant archetypes, as in the latter. Summarising, in the selected cognitive architecture, the operation of (Galant archetype) example–based learning concerns the representation of archetypes and the methods that update these representations.

2.2.2.3 The operation of (Galant) archetype discovery

The last musical schemata identification operation examined in this study concerns the discovery of Galant archetypes (henceforth, `SD`, as in Schemata Discovery). The idea of `SD` is similar to the goals of the 'Paradigmatic' analysis (Ruwet, 1972; Nattiez, 1975; Ruwet and Everist, 1987): the identification and categorisation of *paradigms*, collections of salient elements in music by the expert musicologists, on the grounds that these are repeated, literally or with more or less varied elaborations, throughout the piece, and/or that they have contextually distinctive musical features in common. The goals of these two methodologies differ in scope for two qualities: paradigmatic analysis suggests work–wise analysis for any kind of *paradigms*, while the `SA` examines corpus–wise analysis for a specific musical structure, the sequence of schema–events of the Galant archetype.

Within the context of the selected cognitive architecture, the operation of `SD` was examined as an additional level of autonomy, utilising and extending the two previously described operations of `SA` and `XL` (*see* Sections 3.4 and 3.5 for details). Thus, the operation of `SD` focused on elaborating the bias function of the `SA` operation, and the learning methods of `XL`, in a highly dynamic long–term memory.

2.2.3 Summary

The examined approach for the cognitive model and its aims fits the descriptions for the development of a Symbolic AI system, transferring the concepts of knowledge–base and reasoning in the musicological domain, and through `GMST`, into archetypes and similarity (*see* Section 2.2.1). Considering such architecture, each of the three high–level musical schemata identification operations was presented through a system of information modules.

2.3 Technological research methodology

According to the categorisation of music information processing tasks presented in the 2013 roadmap for music information retrieval (Serra et al., 2013), this study follows the 'technological' perspective, which includes the:

> [G]athering and organisation of machine–readable music data, development of data–representations, and methodologies to process and understand that data, taking into account domain knowledge and bringing expertise from relevant scientific and engineering

disciplines.

Therefore, the methodological tools of this study adopted:

- Annotated datasets, collections of digital scores (i.e., *partimenti*, excerpts, and complete parts from piano sonatas) in symbolic music data format, accompanied by Galant musical schemata annotations;
- Computational systems that process the annotated datasets performing the examined musical schemata identification operations (*see* Appendix D), and
- Computational experiments that quantify the qualitative features of the examined methods according to the performance metrics of musical schemata identification tasks.

Through examining the effectiveness of the performance of the novel computational system in executing those identification operations, the proposed model for musical schemata identification can then be evaluated. The rest of Section 2.3 describes how the above methodological elements were addressed in this study.

2.3.1 Galant musical schemata annotated datasets

The annotated datasets of this study are collections of digital scores in MusicXML format (Good, 2001), with the addition of computer–readable annotations in JavaScript Object Notation format (JSON[4]) regarding the position (in measure–range format) and type (family–type and variation) of Galant musical schemata prototypes on those files. For the needs of this study, two annotated datasets were created (*see* Appendix B for further detail):

[4]https://www.json.org/

a. The *Galant schemata* dataset (in Appendix B.1), and

b. The *Keyboard sonatas schemata* dataset (in Appendix B.2).

The *Galant schemata* dataset is an almost complete digitisation of the score examples found in Gjerdingen's book *Music in the Galant Style* (2007). This dataset consists of 304 examples, of which 286 are small excerpts a few measures in length comprising from one up to five schemata each, and 16 complete small parts. In total, the *Galant schemata* dataset consists of 935 annotation entries for 24 musical schemata family–types (14 of which contain more than 10 annotation entries) with the following fields:

```
(example-number, schema-family and variation type, measure-range)
```

as elucidated in Appendix B.1.

The *Keyboard sonatas schemata* dataset tracks three family–types of Galant musical schemata (i.e., the *Meyer*, the *Prinner*, and the *Clausula*) in 50 parts of Classical sonatas from three composers: Joseph Haydn (15 parts), Wolfgang Amadeus Mozart (15 parts), and Ludwig van Beethoven (20 parts). This dataset applies the same format for annotations as does the *Galant schemata* dataset presented earlier, with each annotation entry including an additional field regarding the part–wise uniqueness of each annotated schema.

```
(part, schema-family and variant, intra-part relation, measure-range)
```

For example, a musical schema may appear eight times in a part, of which only two of them are unique and have three exact[5] repeats each. In total, this dataset includes 537 annotations: 168, with 21 uniques, for *Meyer*; 90, with 13 uniques, for *Prinner*, and 270, with 39 uniques, for *Clausula.* Further details on this dataset can be found in Appendix B.2.

[5]An exact repeat of a schema instance contains the same notes, yet these may be transposed in different tonalities.

2.3.2 A library for operations with symbolic music data

This study required the development of a library for low–level and common music–analytical operations with symbolic music data to facilitate the development of a computational system that performs musical schemata identification tasks. Such a library (*see* Appendix D.1) consists of autonomous functions that can be utilised in Python[6] programming language scripts. These functions are organised in the following categories:

- MusicXML file converters, creating two (algorithmically) operational representations: note–lists or, more generally termed *datapoints*, and 'minimal segments', i.e., stable pitch continuous temporal segments (*see* Section D.1.1);
- Score–wise task–oriented music–analytical functions, extracting file–wise tonality (Krumhansl, 1990), harmonic segments (Pardo and Birmingham, 2002), and outer voices for melodic and bass movements (*see* Section D.1.2);
- Feature extraction functions, extracting abstract and implicit indicators regarding mainly statistical properties from scores, such as average horizontal and vertical note density, and average (temporal) note–intervals, and
- Similarity models, for the comparison between the various musical structures, including schematic elements and musical schemata representations.

[6]https://www.python.org/

2.3.3 A computational system for musical schemata identification operations

The musical schemata identification model and its examined methods were tested through a novel computational system developed for this purpose, the Adaptive Expert System (henceforth, `AES`). The `AES` includes computational implementations for the examined high–level operations (i.e., schematic analysis, examples–based learning and discovery), and workflows for the corresponding tasks testing them. The above are overviewed in the second and third sections of each computational experiment chapter of this study (*see* Chapters 4, 5, and 6). An overview of the system is presented in Appendix D.2.

2.3.4 Evaluating the musical schemata identification model

As stated earlier, the evaluation of the novel musical schemata identification model relies on performing computational experiments. Computational experiments are the core methodological tool of the study; the current section demonstrates how these were configured.

2.3.4.1 Designing computational experiments

The computational experiments of this study have the following elements:

1. A task, with goals defined by the examined high–level musical schemata identification operation;
2. A computational implementation for the thought–pattern of the pro-

posed model performing this high–level operation;

3. A set of configurations for the parameters of the computational implementation for the thought–pattern;
4. Performance metrics for the task performed, and
5. Run–time details, including information regarding utilised datasets, identification targets (if any), and results, according to selected performance metrics.

The first three elements of the above list form the hypothetical part of a computational experiment: the tasks and goals of a high–level operation represent a set of assumptions in forming the music–analytical problem; the computational implementations of the thought–patterns that perform these tasks represent assumptions about the possible methods to implement these tasks and achieve their goals, and the set of configurations for the parameters of a computational implementation represent assumptions about the tuning of the underlying information processing system that represents a thought–pattern. Performance metrics display the efficiency of a model performing a task. Run–time details represent the scope of the data that are examined by the hypothetical parts of an experiment. These may be viewed as the 'occasion' of information that is selected from all of the possible information that could be processed. Lastly, the results of a computational experiment are the 'factual' information of its hypothetical part, reflecting on a data 'occasion', and under the selected point of view as defined by the performance metrics. Thus, the musical schemata identification operations of the proposed model were examined while performing different tasks, also with different goals. The success of a proposed thought–pattern is related with how well its computational implementation (through the `AES` system) performs in a computational experiment, and under a selected performance metric.

2.3.4.2 Testing hypotheses with computational experiments

The computational experiments of this study tested three types of hypotheses:

- The top–level hypothesis of a cognitive model for musical schemata identification operations;
- Methodological hypotheses on the proposed approaches for the three high–level operations;
- Conceptual hypotheses, examining the impact of the modeling of various notions from Psychology that participate in the music–analytical thought–patterns.

All of the above hypotheses are reflected in the performance of the proposed model for musical schemata identification when performing computational experiments: a hypothesis may have beneficial, detrimental or invariable consequences on achieving the goal of a task, depending on whether it (the hypothesis) has a positive, a negative or a neutral effect on the performance metric utilised by the examined computational experiment. The evaluation of the top–level hypothesis was performed by the collective performance of all of the identification operations performed by the `AES`. Methodological, as well as conceptual, hypotheses were evaluated by analysing the results of each musical schemata identification task.

2.4 Summary

This chapter has presented core elements from the `GMST` (*see* Section 2.1.1) and related elements from Psychology (*see* Section 2.1.2) in order to describe the musical schemata identification operations undertaken by the examined cognitive model in the context of Computational Musicology (*see* Section 2.2). This study elected to follow a 'technological' research methodology (*see* Section 2.3) for the evaluation of the proposed cognitive model. Applying this approach, this study envisaged the development of two annotated datasets (*see* Appendix B), a library of autonomous functions for operations with symbolic music data (*see* Section D.1), and the development of computational system (`AES`, *see* Section D.2) that tested the model with three musical schemata identification workflows (for search, classification, and discovery).

3

A cognitive model for Galant musical schemata identification operations

This chapter describes the cognitive model designed to study the automatic identification of musical schemata archetypes. The model integrates the concepts of Galant archetypes and *partimenti* from the `GMST` (*see* Section 2.1.1) and schemata in Psychology (*see* Section 2.1.2), to form a memory architecture that facilitates the examined musical schemata identification tasks. After presenting the architecture of the model and basic terminology in this hypothetical mental system of music information transformations, these are utilised to describe the music–analytical thought–patterns for the three high–level musical schemata identification operations of schematic analysis, example–based learning, and discovery. The computational implementation of the model and its operations are described in the following chapters (*see* Chapters 4, 5, and 6), and an overview of the computational system that hosted the examined model can be found in Appendix D.2.

3.1 Overview

The goal of this study was to explicate on the mental processes involved when performing (Galant) musical schemata identification operations. To facilitate this goal, the abstract notions of memory spaces were simulated by information systems that perform specific operations, also under the premises of the `GMST`. The cognitive model presented here, therefore, is a system of music information transformations that facilitates the reference to musical structures, and the examination of the operations that create and process them.

The examined cognitive model is simulated by the interaction between two systems, one managing the classification of Galant archetypes in the long–term memory, and the other creating the Galant information environment in the working memory, through the operation of schematic analysis (`SA`). Considering the Symbolic AI approaches, the classification system of the Galant archetypes is the knowledge–base, and the process of schematic analysis offers the 'world' view of music notation. Reasoning occurs in both systems yet with different goals. In the classification system, the reasoning concerns the extraction of a class–similarity function. In the operation of schematic analysis, the reasoning addresses the extraction of the Galant feature–set with grouping mechanism. Within this simple architecture, music information processing paths and inter–locking routines were formed, culminating in music–analytical thought–patterns performing the examined musical schemata identification operations.

3.2 Model architecture

The cognitive model examined and its musical schemata identification operations focused on the perceptive and cognitive processes in and between the working memory (*WM*) and the long–term memory (*LTM*), when interacting with music–notation and annotations thereof (*MD*). Aiming to describe these processes in a methodological manner, and further utilise them to explicate the operations of schematic analysis (`SA`), example–based learning (`XL`), and discovery (`DS`), this highly abstract music information interaction system was simulated by the interaction between two information systems:

- A classification system for Galant musical schemata family–types (henceforth, `GMSC`), simulating the *LTM* schemata, and
- The process of schematic analysis (`SA`), simulating the 'control' mechanisms of the *WM* when extracting the Galant 'musical surface'.

Each Galant musical schemata family–type (henceforth, `GMSC.SF`) in the `GSMC` is represented by an exemplar–base, a collection of temporally contextualised *partimenti*, from which an exemplar (or more) has the role of the archetype/prototype (or variant), and all other exemplars are expressed as approximations through an archetype–class–similarity function. In the highly dynamic *WM*, information from both the *MD* and the *LTM* interact in varying degrees, with the goal to create a *partimento* view of music notation and facilitate the learning and discovery musical schemata identification operation. The following segments of this section present the long–term memory schemata represented by the `GMSC` system; also shown is a description of participatory processes in the *WM*, overviewing the operation of `SA`, which is described in detail in the following section (*see* Section 3.3).

3.2.1 Representing long–term memory Galant musical schemata

The classification system for Galant archetypes (`GMSC`) examined a combinatorial representation scheme, where each Galant archetype (`GMSC.SF.AR`) accommodates the 'memory–space' that is created by the polarisation and quantisation of the following qualities:

- Structural integration;
- Value contextualisation, and
- Frequency of appearance.

The structural integration of a Galant archetype representation expresses its 'completeness', measured by the presence of schematic–elements from the archetypal form. For example, when only a single schema–event of a Galant archetype is to be considered, then this is a representation of that class with low–level structural integration. In another example, when the melodic movement of a Galant archetype is to be considered, this is a representation with greater than the previous example structural integration.

Value contextualisation concerns the levels of explicitness in the content–related values of a Galant archetype. These start from explicitly defined pitch and temporal values, up to tonality and rhythm contextualisations. For example, when reading a *partimento*, a melodic interval is represented as a horizontal note–interval between notes with explicitly defined values, e.g., an F–sharp crotchet note on the fourth beat of a four–beat measure, followed by a G quaver on the first beat of a four–beat measure. At another end, considering the contexts of rhythm and tonality, the previously mentioned melodic movement is expressed as a melodic interval of +1 scale degree from the seventh scale degree in G major, with an inter–onset–interval–to–measure ratio of 1/4, and a durational ratio of 1/2. This parameter enables access to any kind of value–explicitness that may be required by the system.

Frequency of appearance considers the amount of times a specific musical structure, also in a specific contextualisation, appears in the memory space. This quality is especially useful when extracting archetypal forms, as it equates directly to the importance of such a structure.

The power set of the above parameters substituted by their possible value–set creates the 'memory–space' of a Galant musical schemata archetype:

```
( integration x contextualisation x frequency )
```

Each Galant family–type is represented by such (Galant) 'memory–spaces', which are accessed and updated according to the operation performed.

3.2.1.1 Galant musical schemata representation

The 'memory–space' of each Galant musical schema is managed by the `GMSC` with the following elements:

- A super set of a Galant family–class (`GMSC.SF`), encompassing at least one archetype (`GMSC.SF.AR`);
- An exemplar–base (`GMSC.SF.AR.EB`), collecting segments in schematic analysis form, a kind of temporally abstracted *partimenti*;
- An archetypical form (`GMSC.SF.AR`), the most prominent exemplar from the exemplar–base, and
- A class–similarity function (`GMSC.SF.AR.CS`), validating all of the exemplars in the `GMSC.SF.AR.EB` through conditional approximations on the archetypal form.

It is this representation that is sought, learned and discovered in this study, selecting elements of interest that form the 'world' model in working memory, and facilitate the three learning methods of accretion, tuning, and restructuring. The elements of the `GMSC.SF.AR` representation system are now elaborated, as these are utilised to express the musical schemata identification operations.

3.2.1.2 The exemplar–base

The exemplar–base of a Galant archetype (`GMSC.SF.AR.EB`) enables the creation and update of the (Galant) 'memory–space' by maintaining a collection of *partimenti*. Each exemplar is a *partimento* with contextualised temporal– and pitch–related properties. For example, the *partimento* of the 'Leaping–bass Romanesca' schema shown in Figure 2.3 has the following values:

```
[
 (((1:1),1),((1:1),3), (I,53)),
 (((1:7),5):((1:7),2), (V,53)),
 (((2:1),6):((2:1),1), (VI,53)),
 (((2:7),3):((2:7),7), (III,53)),
 (((3:1),4):((3:1),6), (IV,53)),
 (((3:7),1):((3:7),5), (I,53)
 ]
```

Recall that a Galant musical schemata instance is represented as a list of schema–events:

```
[ (schema--event),... ]
```

where each schema–event holds information regarding

```
(bass--note),(melody--note),(chord)
```

with each note represented with

```
((measure, beat), scale--degree)
```

and harmonic information represented as

```
(harmonic--degree, inversion)
```

The exemplar of the partimento above separates pitch and temporal properties and abstracts the latter as follows:

```
[ [  (1,3,(I,53)),
     (5,2,(I,53)),
     (6,1,(I,53)),
     (3,7,(I,53)),
     (4,6,(I,53)),
     (1,5,(I,53))],
  [  ( (1:1), (1:1), (1,1)),
     ( (1:1), (1:1), (1,1)),
     ( (1:1), (1:1), (1,1)),
     ( (1:1), (1:1), (1,1)),
     ( (1:1), (1:1), (1,1))   ]
]
```

The first group of values of an exemplar representation concerns the pitch–related properties in scale–degrees and harmonic–degree and intervals from bass as a sequence of schema–events. The secondary group of values represents the temporal–related properties expressed as interval ratios: the first value is the ratio between durations of two successive schema–event duration, and the second and third, the inter–onset–interval of the notes of those schema–events. The main idea of the exemplar representation is to separate pitch– and temporal–related information, and facilitate various kinds of abstractions.

The exemplar–base is a simple and efficient way to maintain detailed information regarding the instantiations and the variations of a Galant archetype. This data structure is utilised by the classification system for:

- The creation and update of the representational 'memory–space' of a Galant archetype class,
- The selection of the most frequent/important exemplar as the archetype representation for the Galant archetype class, and
- The calibration of the class–similarity function.

3.2.1.3 The Galant archetype

The archetypal representation of a Galant archetype class (`GMSC.SF.AR`) is the most prominent exemplar in the exemplar–base, where prominence is expressed by the frequency of its appearance in the exemplar–base (`GMSC.SF.AR.EB`). In the case where multiple–archetypes are to be considered, these are encompassed into a Galant archetype family–type class (`GMSC.SF`), representing those archetypes separately, each with its own exemplar–base and class–similarity function.

3.2.1.4 The Galant archetype–class–similarity function

The similarity between exemplars may take various forms; these depend on the three factors of which properties of those composite musical structures are considered, the metrics when comparing them, and the context of operation in which similarity is occurring. The class–similarity function of a Galant archetype representation inputs the exemplar–base of a Galant archetype, and creates a set of conditional variations for each property on a selected exemplar that acts as the central archetypal form. This part presents how the Galant archetype class–similarity function performs comparisons between exemplars, and how these are utilised to express their inter–relation and their relation to an archetype.

Recalling the properties of a Galant archetype (*see* Section 2.1.1.1), these are:

- Structure–related, concerning the number of events;
- Content–related, concerning the values on the properties for the qualities found in a schema–event (melody and bass movement, intervals from bass, and metric strength), as well as its temporal relations, and
- Context–related information, concerning the phrase properties of an archetype.

The class–similarity function focused on structure– and content–related differences, electing to leave temporal– and context–related differences for future research.

The structure–related similarity between two exemplars is measured by the quantitative difference of their maximal structural integrations. For example, the Romanesca variants may have four or six events (*see* Appendix A.1), with a difference in structural integration of two schema–events. Considering the Clausula family–type and its many variants, the reason why structural integration was preferred over counting schema–events becomes apparent. In this case, when comparing an exemplar of a 'simple' *Clausula* (*see* Figure A.9) with its structural integration of a single melodic bass–movement from the fifth scale degree to the first, to an exemplar that has maximal structural integration (i.e., both melodic movements and harmony), then structure–related similarity is considered equal, but of low level. This is because despite the fact that both exemplars have two schema–events, only a single melodic movement may be considered. The difference between structure–related information may be expressed with the edit–distance metrics, and the operations of addition/removal of schema–events. Structure–related differences were considered to be an important differentiation factor and were utilised as such by the class–similarity function when considering variants.

Content–related similarity concerns the differences in primarily pitch–related properties. When comparing the pitch–related values of melodic scale–degrees and intervals from bass (the values of a schema–event), their difference is measured according to their value–contextualisation. However, different contexts may also be utilised as well. Thus, when same–order notes of the same type of melodic movements are compared, two metrics may be utilised:

- Absolute scale degree difference, and
- Harmonic difference, a flag indicating that the two scale–degrees are of the same harmony, which is that of the schema–event where they appear.

Similarly, when comparing the harmonic information of two schema–events, these may differ in class and inversion.

When comparing the pitch–related properties of exemplars, a difference vector is created: each entry of the vector represents the difference of the individual properties of the two schema–events of the same order. Each schema–event progression of an exemplar has the following fields:

```
[ [bass scale degree, intervals from bass, melody scale--degree], ... ]
```

When comparing two such structures, the following (exemplar) difference vector (`EDV`) is created:

```
[ [  dmel, dharm, dmel], ... ]
```

Here, `dmel` and `dharm` may represent the difference of scale–degrees and harmony as presented above. The following example shows the pitch–related properties of two exemplars: `A` is a *Meyer* Galant archetype (*see* Appendix A.5), and `B` is a variant (in Haydn, Symphony in D, Hob. I:73, 'La chasse', mvt. 1, Allegro, m. 27, ca. 1781, Example 9.6, p.114, Gjerdingen, R. 2007). The row labelled `df` represents the distance between the pitch–related properties of these exemplars.

```
A   : [1,53,1], [2,63,7], [7,65,4], [1,53,3]
B   : [3,63,1], [2,63,7], [5,43,4], [1,53,3]
df  :[+2,+1i,.], [...],  [-2,+1i,.], [...]
```

The scale–degree similarity function (`dmel`) uses an absolute scale–degree distance metric (`+2` and `-2`). The harmonic similarity function (`dharm`) indicates that both pairs of schema–events (`1` and `3`) have the same harmonic class, with the second schema–event of each pair being one inversion greater (`+1i`).

Utilising exemplar difference vectors such as the one presented above, the class–similarity function converts them into conditional statements and thresholds that validate a match. For example, the difference vector between the two *Meyer* exemplars above (row `df`) is translated into the conditional statement of:

```
(event 1, bass: +2, harmony: +1 inversion)
        AND
(event 3, bass: -2, harmony: +1 inversion)
```

Therefore, utilising the class–similarity function, variations of an archetype may be expressed as a conditional statement for structure– and content–related differences.

The class–similarity function is activated whenever a new exemplar is introduced to the exemplar–base of a Galant archetype class. Initially, when there is no archetype for that class, the properties of the exemplar become the properties of the family's archetype form. In this case the class–similarity function returns clean structure– and content–related difference vectors. When an archetype for the incoming exemplar does exist, then, one of the following cases may be:

- The creation of a variant archetype, if structure–related differences are found;
- The change of the archetype, if the frequency–of–appearance for that particular exemplar becomes the greater among others;
- The change of the similarity–function, if only content–related differences are found, and
- The increase of the 'significance' of that particular exemplar, by increasing its frequency–of–appearance in the exemplar–base.

All of the above cases are considered through the class–similarity function and its thresholds. For example, if an exemplar passes the structure–threshold of the class–similarity, then an additional archetype is considered, with its own exemplar–base and class–similarity function. Similarly, considering content–related thresholds in combination with frequency–of–appearance, more elaborate mechanisms for archetype extraction may be applied. If, for example, a Galant archetype class has in its exemplar–base only two variants with equal frequency–of–appearance, then these may also separate into two variant archetypes.

Summarising, the class–similarity function (`GMSC.SF.AR.CS`) offers a human–friendly method to identify the differences among the exemplars of an exemplar–base of a Galant archetype representation. This function will play a central role in explicating the thought–patterns of the example–based learning operation, and is further utilised by the discovery operation as a mechanism through which to identify potential Galant musical schemata classes. Details on how the class-similarity function is utilised are shown in the following sections, when the high–level operations are described.

3.2.2 Simulating the Galant ‘control mechanisms’ in the working memory

The working memory module (*WM*) is a dynamic plane where information from the long–term memory (*LTM*) interacts with music notation (*MD*) through those ‘control mechanisms’ that interpret the Galant ‘musical surface’: two schema–voices and their underlying schema–event progression. The *WM* is simulated by the operation of schematic analysis (`SA`); all information processing details, as well as the music–analytical thought–pattern of the `SA`, are shown Section 3.3. First follows a description of the `SA` notion of an ‘active plane’ (`SA.AP`) where certain kinds of information become simultaneously available for the control mechanisms to process. Then, an overview of those control mechanisms that extract the Galant surface is presented, describing how a system of hierarchical inferences utilises the information of the 'active plane' for the interaction between the top–down biases and bottom–up constructs of musical structure.

3.2.2.1 The 'active plane' of the schematic analysis operation

The 'active plane' is the main information processing routine of the `SA` operation (`SA.AP`) creating model–wise information–states (i.e., the available information in the model during the `SA` operation in a given time); it does so by aggregating the information–state of the following three sources:

- A 'time–window' note–sampling mechanism, offering the set of notes that are found within the bounds of a temporal range that is gradually overlaid throughout music notation (the time–window info–state, `SA.AP.TW`); this information is subjected to the control mechanisms, extracting note–relations and identifying local contexts and schematic–structures;
- A score–wise segmentation map, initiating with a main tonality and time–signature, and aggregating harmonic and voice segmentation information from the outcomes of the time–window registers (the score–wise info–state, `SA.AP.SW`), as well as the extracted Galant musical surface, and
- Any kind of information stored in the classification of the Galant musical schemata archetypes (`GMSC`), depending on the operation performed (the memory–wise info–state, `SA.AP.GMSC`).

The `SA.AP` routine creates a new model–wise info–state whenever the time–window progresses a step, or a new score is considered. That is because after the operation cycle of the control mechanisms on the note–set of the `SA.AP.TW`, the score–wise info–state is extended to include the outcome of the control mechanisms. Depending on the high–level operation performed, the `GMSC` might also be altered. For example, in the example–based learning operation, if there exists an annotation within the bounds of the time–window, then a learning routine is called into service, altering the status of the `GMSC`. For that reason, at each step of the time–window, a new 'snapshot' of the model's information sources is taken, allowing the consideration of any possible updates on any of its constituent info–states. The notion of the active plane (`SA.AP`) enables the simultaneous consideration of information derived from multiple sources of dynamic information.

The time–window method offers a sequential means of reading the music notation; it was therefore selected to provide an alternative to full score/corpus (batch) information processing. To prevent possible issues that may relate to the size and the pacing of the time–window, an (arbitrarily) fixed length of fifteen measures was considered, with a pacing of five measures. In this study, the selected size and pacing for the time–window were considered as 'fail–safe' when identifying any possible Galant archetype instantiation in a music analytical operation. Score–wise information is initiated with, at the minimum, the tonality and time–signature found on the score. It aggregates the outcomes of the control mechanisms processing the information of the sliding time–window, i.e., the extracted Galant surface, a sequence of schema–events, and all related contextual information, such as tonality and harmonic segments. The Galant archetype classification (`GMSC`) may offer various kinds of information to the 'active' plane, generally simulating what is referred in this study as the top–down 'bias' on the control mechanisms. In the selected model architecture, the use of 'memory' buffers is an inherent property: the time–window provides a temporal locality in music notation, the score–wise information offers a 'session' buffer for complete parts, and the `GMSC` suggesting a 'schematic' representation for complete corpora.

The next part describes how the Galant control mechanisms are simulated by a system of hierarchical inferences, each utilising top–down biases and bottom–up groupings to extract even greater structural–integrations to the Galant surface.

3.2.2.2 Simulating the Galant control mechanisms

The Galant control mechanisms process model–wise information–states from the active plane. Their goal is of transforming the music notation found in the particular time–window buffer into a sequence of schema–events. This is achieved by applying a hierarchical inference system that decides upon an 'optimal' form of the extracted schematic analysis. Such an inference system simultaneously considers a set of inference sub–systems for the three co–existing musical structures of a Galant surface, the 'elemental' levels of structural integration:

- Schema–event (the minimal structural integration);
- Schema–voice, and
- Schema–event progression (the full structural integration).

In such a way, the extraction of each musical structure above has its own inference system that considers two grouping processes:

- The bottom–up construction of the schematic–elements and progressions of schema–events considering the pitch and temporal relations of the notes in the time–window and grouping principles from Psychology, and

- A top–down 'bias' on bottom–up 'constructs', applying structure– and content–related constraints drawn from the score–wise information–state and the various structural–integration levels of the Galant archetypes in the classification.

These grouping mechanisms are the core 'reasoning' method of the `SA` operation as these create the Galant information environment. The basic theory as to how these control mechanisms are simulated is that the bottom–up methods create musical structures by merging sub–structures that form from perceptually modelled temporal and pitch note–relations; the top–down methods threshold those 'constructed' musical structures based on existing information, or any other axiom imposed by the user. The next section describes how each level of schematic–integration is created and how these grouping mechanisms are applied.

3.2.3 Summary

The examined cognitive model is simulated by the interaction between a classification system (`GMSC`), expressing the long–term memory schemata, and the operation of schematic analysis (`SA`), representing the working memory. The Galant musical schemata archetype classification (`GMSC`) represents each Galant archetype with an exemplar–base, a set of abstracted *partimenti*, from which, one (exemplar) is selected by a class–similarity function to be the archetypal form for that class. All operations within the `GMSC` relate to the class-similarity function which, by comparing structure– and content–related information, formalises the differences among exemplars, facilitating higher–level operations with Galant archetype representations.

The operation of schematic analysis (`SA`) simulates the interaction between the `GMSC` and the `MD` with the notion of an 'active' plane (`SA.AP`), where temporally–sensitive (dynamic) information from the three information–states of: a time–window traversing the score; a score–wise buffer, and the state of the `GMSC`, becomes available to a hierarchical inference system of control mechanisms that extract the Galant information environment. The hierarchical inference system validates the 'well–formedness' of schema–events, schema–voices, and schema–event progressions by simulating the control mechanisms that simultaneously perform bottom–up, the perceptual 'construction' of these structures, and top–down biases, thresholding those 'constructs' according to existing information or imposed axioms. Utilising the above architecture, each of the following sections in this Chapter describes one of the three musical–schemata identification operations.

3.3 Schematic analysis of music notation

In this study, the operation of schematic analysis (`SA`) simulates the working memory (*WM*) when converting music notation into a sequence of schema–events, the Galant information environment. The examined music–analytical thought–pattern performing the `SA` operation suggests the gradual increase of 'awareness' of schematic integration into music notation, until an 'optimal' sequence, according to an inference system, of schema–events is identified. After presenting the music–analytical thought–pattern of the `SA` operation, the next segments describe the stages of this operation in detail, highlighting those inference systems that extract schema–events, schema–voices, and eventually schema–event progressions.

3.3.1 The music–analytical thought–pattern for schematic analysis

In the architecture of the examined model, the operation of schematic analysis is performed whenever the information of the time–window buffer in the active plane (`SA.AP.TW`) changes. The thought–pattern for schematic analysis suggests the following steps (C for Case, S for Stage, A for Action):

C1: When considering the notes in a time–window (`SA.AP.TW`).
S1: Perform Tonal analysis.
A1: Identify and apply tonality and rhythm contexts to note–elements.
A11: Scale–degrees in a score–wise tonality.
A12: Metric–strengths, according to score–wise time–signature.
A2: Identify and apply basic note–relations.
A21: Two outer voices.
A22: Harmonic segments.
S2: Perform Schematic analysis (with application of 'significance', 'clarity', and 'regularity' models):
A3: Identify single schema–events.
A4: Identify single schema–voices.
A5: Identify schema–voice–pairs.
A6: Identify schema–event sequence.

The first stage of the `SA` operation's thought–pattern (i.e., Actions **S1.A1** and **S1.A2**) performs what is commonly referred to as Tonal analysis. This stage regards the identification of those 'infrastructural features' (*see* Section 2.1.1). Those are the basic music–theoretic ontology of the score concerning temporal and pitch contexts and entities (*see* Section 3.3.2), such as tonality and scale, rhythm (as measure organisation implied by time–signatures), outer melodic movements and harmonic segments.

In the second stage, the core processes of the `SA` operation are performed, gradually identifying musical structures from low– to high–level schematic integration. In this phase, a series of schematic elements is identified with the application of those 'control mechanisms' that construct and filter musical structures based on perceptive principles; they include positional– and transitional–significance, clarity and regularity. In each action of this stage, the control mechanism of each schematic integration generates a pool of samples/constructs, rated with 'perceptive' power, according to the aforementioned principles. In the final action of the `SA` operation (**SA.A6**), an inference mechanism (`SA.SS.INF`) decides upon which combination of those rated schematic elements is the most prominent for the schematic analysis form.

The rest of this section presents the identification stages of the `SA` thought–pattern in detail.

3.3.2 Score analysis

The Galant musical schemata archetypes represent structured and abstracted information of the following musical contexts and entities:

- Tonality;
- Measure organisation;
- Outer melodic movements, and
- Harmonic segments.

Regarding musical contexts, the identification of tonality (or tonalities) enables the 'translation' of absolute–pitch into scale degrees, and the number of intervals from bass into harmonic degrees. Knowledge about measure organisation, through time–signature(s) found on the score, enables the assignment of metric–strengths to each note and, by merging, to each schema–event. Regarding musical entities, the consideration of the outer–voice movements suffices as an initial step towards the identification of schema–voices. The identification of harmonic segments plays an important role in the identification of schema–events.

3.3.3 Identification of schematic elements

The stage of schematic–analysis begins after the musical contexts of tonality and measure organisation and the musical entities of outer–voices and harmonic segments have been identified. Utilising the above information, each action of this stage generates samples from the music notation found in the time–window for the following musical structures:

- Single schema–events;
- Single schema–voices;
- Schema–voice–pairs, and
- Schema–event progressions.

As discussed in the previous Chapter (*see* Section 2), schema–stages are considered to be temporal and pitch prolongations of the harmonic space of schema–events. Therefore, schema–stages were not considered as structural elements of the Galant representation, but modelled with schema–event 'transitional significance' (as is discussed next).

3.3.3.1 Modelling perceptive qualities

For the extraction of 'schematic' structures from music notation, notes are selected according to the qualities of their temporal and pitch relations. Three notions that describe such qualities are:

- Significance;
- Clarity, and
- Regularity.

The above notions facilitate the filtering by perceptive power of the notes and the musical structures that construct schematic elements, as well as the schematic elements themselves.

The notion of 'significance' attributes 'perceptive power' to single notes and structures thereof (by merging), with consideration of the temporal and pitch context in which they appear, and their local interrelations. In other words, the notion of 'significance' describes how musical structures, ranging from a single note up to a schema–event sequence, are able to distinguish themselves from similar structures around them. The two main qualities of this perceptive power concern the 'positional' and the 'transitional' significance of such elements. Positional significance concerns the metric properties of these elements. For example, when extracting schema–voices and in the process of selecting notes, if a note is in a weak metric position then its perceptive power is considered to be lower than that of a note that appears in a strong metric position. When note–relations are to be considered, such as schema–events or voices, the total significance of a structure is expressed by the average of the notes that comprise it.

While the quality of positional significance can filter out embellishing notes and chords, if it is the only criterion applied, then a single temporal lattice of quantised metric–strengths would mask all note–relations. In such a case, embellishments such as suspension and retardation cannot be identified by design. This issue is addressed by the 'transitional' significance described next.

The quality of transitional significance describes the importance of the change in musical properties that occurs between two elements. For example, when creating the sequence of schema–events and in the process of selecting the next schema–event in the sequence, where two adjacent schema–events are of the same harmonic quality, their transitional significance is considered to be low. In another example: when the ratio of durations between two adjacent schema–events is farther removed from one, e.g., where a four–beat in duration schema–event occurence is followed by a quarter of a beat in duration schema–event (i.e., 16/1), this relation is also considered to be of low transitional significance, suggesting the omission of the second schema–event from possible integration to the output sequence. The modelling of this notion is presented when extracting single schema–events (*see* Section 3.3.3.3).

3.3.3.2 Clarity of schema–voices and schema–events

Using findings from music perception and cognition (e.g., Bregman, 1994; Deutsch, 2013; Huron, 2001), this study considered the creation of a 'musically–meaningful' set of conditions that support the perceptive 'clarity' of *schema-voices* and *schema-events* (*see* Table 3.1). These conditions incorporate voice–leading and auditory streaming principles to restrict specific conditions that may be present between the matching of melody and bass schema–voices. For example, one such filter bans the crossing between *schema–voices* to maintain the clarity of each melodic movement. Similarly, schema–event overlapping was also prohibited because it decreases the clarity of each

Table 3.1: Clarity filter–set for schema–voices. `M` and `B` are melody and bass note–sequences and `X[i]` is their i^{th} element (datapoint). Each datapoint has the properties of pitch, onTime and offTime. `schemaEvent[i]` is the i^{th} *schema-event*.

Parameter	**Description**
no voice-crossings	M[i].pitch >B[i].pitch
no voice-overlaps	M.pitch >B.pitch
no event overlap	schemaEvent[i].offTime <= schemaEvent[i+1].onTime
no unisons in events	M[i].pitch != B[i].pitch
different pitch in events	M[i].scaleDegree != B[i].scaleDegree

schema-event. The rules shown in Table 3.1 play an important role in eliminating erroneous combinations of schematic–elements. However, since these rules regulate mainly pitch relations, they have minimal effect on temporal related issues. The latter are managed by qualitative and quantitative metrics of regularity, presented next.

Regularity is a fundamental quality of the music examined in this study, if not in music generally. Temporal regularity is explicitly imposed in common music notation with time–signatures and meter organisation. When extracting schematic structures, temporal regularity is utilised as a filter, by counting and thresholding the amount of regular (equal) temporal intervals on specific note–relations. As stated in Section 2.1.1, when Galant archetypes are instantiated in compositions, the notes and schema–events that construct them have preferable temporal arrangements/relations. For example, the schema–events of the *Prinner* are usually isochronous, while those of the the *Meyer* appear in pairs. In applying profiles of temporal regularity filters, such relations may be addressed. The modelling of this perceptive quality is discussed in the extraction of the schema–event sequence (*see* Section 3.3.4.1).

The following parts describe how the above notions are utilised for the extraction of schematic elements, presenting the methods for the extraction of single schema–events, and schema–event progressions in detail.

3.3.3.3 Extraction of single schema–events

The algorithm for the identification of schema–events begins with the outcomes of the score analysis stage, and examines the notes of the time–window by creating pairs of 'minimal segments' within local temporal bounds. A minimal segment (*see* Appendix D.1.1) is a continuous temporal segment with unchanged pitch properties. First, each minimal segment is assigned positional and (adjacent) transitional 'significance' ratings from two algorithms that weight parameters from the minimal segments' properties. Then, a merging function creates schema–event samples from pairs of minimal segments by combining and selecting values for the schema–event content–related information.

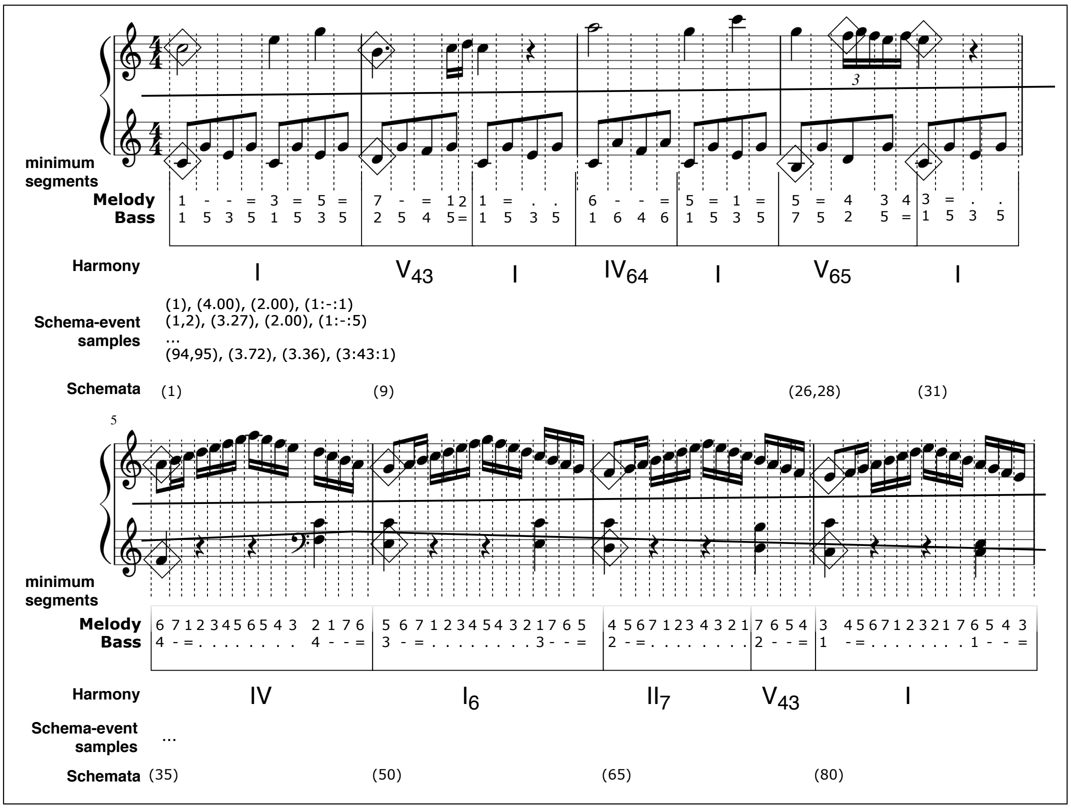


Figure 3.1: The musicXML example is first converted into minimal segments (1) and features such as harmony (2) are calculated using external algorithms. Next, two types of 'significance' are assigned on each minimal segment: *positional* (3), by rating their content in relation to the overall rhythmic and tonal contexts of the example, and *transitional*, by rating the difference between adjacent SPTS elements. The 'significance' values are calculated using the formulæ in 3.1 and 3.2 with the weights in Table 3.2.

Rating positional significance of minimal segments . The positional significance of each minimal segment is found considering formula 3.1 and the weights in Table 3.2:

$$poSig = \frac{1}{3}(dpQ + BStr + card) + outVp \tag{3.1}$$

Table 3.2: Parameters and their weights when calculating the *positional* 'significance' of a minimal segment.

Feature	Factor
Beat strength	2
Harmony	2
Cardinality	1.5
Outer voices	2
Complete datapoint in minimal segment	1
Starting datapoint in minimal segment	1
Ending datapoint in minimal segment	0.25
Middle datapoint minimal segment SPTS	0.125

where

$poSig$ is the positional significance of the minimal segment,

dpQ is the onset quality of datapoints in the minimal segment,

$BStr$ is the beat–strength value,

$card$ is the cardinality, and

$outVp$ the bonus from movement in outer voices.

Rating transitional significance between minimal segments . The transitional significance of minimal segments is calculated in a similar manner while also considering the changes in the properties of minimal segments, mainly in harmony. The transitional significance of a pair of minimal segments is found considering formula 3.2 and the weights in Table 3.2.

$$tranSig = \frac{1}{4}(harmT + dpQ + BStr + card) + outVt \tag{3.2}$$

where

$tranSig$ is the significance of the minimal segment, *harmT* is the transition quality of harmony between the minimal segment,

dpQ is the overall quality of datapoints in the minimal segment,

BStr is the beat Strength value,

card is the cardinality, and

outVt the bonus from movement in outer voices.

The transition between two harmonies (*harmT*) is quantified as follows:

$$harmT = \begin{cases} 0 & \text{if (Dm==TRUE) AND (Im)==TRUE)} \\ 0.25 & \text{if (Dm==TRUE) AND (Im)==FALSE)} \\ 1 & \text{if (Dm==FALSE) AND (Im)==FALSE)} \end{cases} \tag{3.3}$$

where

Dm is the matching between harmonic degrees, Boolean,

Im is the matching between harmonic inversion, Boolean.

The transition's value (*tranSig*) is found as the product of normalised significance values that are converted around value 1. This is to rate the transition only with a product. It will be important when a segment is extended and the adding of combined 'significances' takes place. The output of this process is a list with pairs indicating the position in the score, a dictionary with overall and per feature values of relative change.

For example, the excerpt in Figure 2.1 has 124 datapoints and 96 minimal segments. Thresholding minimal segments and samples using above–average positional 'significance' ratings, event–sampling returns a total of 107 samples. The first sample of the excerpt in Figure 2.1 is:

```
((1, 4/4, 1.0), [C5*, C4], (4.00), (2.00), (1:-:1))
```

where the values of 4.00 gives the positional significance, and the value of

2.00, that of adjacent transitional significance.

A schema–event sample has on–time and duration and all of the properties of a schema–event, but – and most importantly – each schema-event sample is rated with the combined ‘significance’ ratings of its minimal segments, facilitating further thresholding and grouping operations.

3.3.4 Extraction of schema–event sequences

In the final action of the schematic–analysis stage of the `SA` operation (`S2.A6`), an inference mechanism decides on which schema–event from the pool of rated schematic–elements is appended to the exported sequence of schema–events.

For ease of reference, the sequence of schema–events is represented as follows:

```
S[i].<property>
```

where

`S[i]` is the ith schema–event in the sequence,

`S[i].bass/melody]` is the scale degree for the bass or melody movement of the ith schema–event in the list,

`S[i].harmony]` is for the harmonic information,

`S[i].metric]` for the metric–strength information, and

`S[i].temporalInformation]` is an array with detailed temporal information for each note.

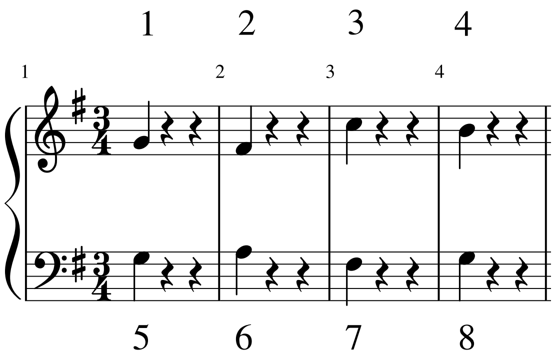
In this task, all three notions concerning the perceptive power of musical structures are participating. The notion of significance is applied in a similar manner as in schema–event extraction (*see* Section 3.3.3.3, above), whereas, in the place of minimal segments, the temporally sorted pool of extracted schema–events is used instead. In this case, the positional significance of the extracted schema–events is already calculated (as the average of the two minimal segments), and the transitional significance between local schema–events is calculated with the same function (*see* Formula 3.2). The quality of temporal regularity aids the selection of schema–events offering a metric for progressions of schema–events, and is described in detail in the rest of this segment.

3.3.4.1 Conforming the sequence of schema–events with thresholds of temporal regularity

In a previous work (Symons, 2012), a strict notion of temporal regularity (isochrony) was employed for the discovery of statistically significant monophonic schemata. In a recent study, Foubert et al. (2017) present the notion of a spectrum of (temporal) regularity, spanning highly regular (e.g., isochronous notes) to highly irregular (e.g., notes where no pair of start-time differences is the same as any other). Considering the temporal arrangements of each schema–type, these possess a certain type of temporal regularity — in other words, are situated in a certain region of a temporal regularity spectrum — that is rarely as regular as the excerpt shown in Example A but never as irregular as Example B.

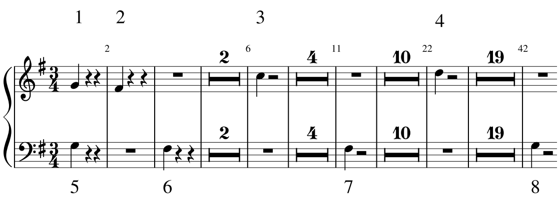
Example A

	1	2	3	4	5	6	7	8
1	0	3	6	9	0	3	6	9
2		0	3	6	-3	0	3	6
3			0	3	-6	-3	0	3
4				0	-9	-6	-3	0
5					0	3	6	9
6						0	3	6
7							0	3
8								0



Example B

	1	2	3	4	5	6	7	8
1	0	1	3	7	15	31	63	127
2		0	2	6	14	30	62	126
3			0	4	12	28	60	124
4				0	8	24	56	120
5					0	16	48	102
6						0	32	96
7							0	64
8								0



Example C

	1	2	3	4	5	6	7	8
1	0	1	6	7	-2	1	4	7
2		0	5	6	-3	0	3	6
3			0	1	-8	-5	-2	1
4				0	-9	-6	-3	0
5					0	3	6	9
6						0	3	6
7							0	3
8								0

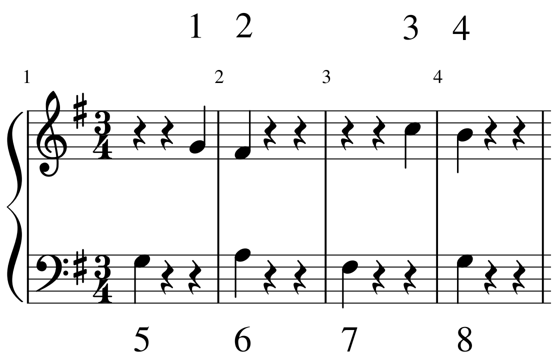


Figure 3.2: Musical examples indicating varying degrees of temporal regularity and their matrices for the temporal intervals between the notes in beats (the matrix is antisymmetric, that is `dij = -dji`, so we need show only the upper triangle). Examples A and B show the extremes in isochrony, perfect and none. Example C shows a more realistic case where specific intervals appear to be isochronous.

Previous work has established that when notes have `n` distinct ontimes (start times), the number of unique non-zero absolute differences between all pairs of ontimes may be as few as `n - 1` and as large as `n(n - 1)/2` (Foubert et al., 2017). Ontimes exhibiting close-to-the-minimum number of differences tend to be perceived as highly regular, whereas those exhibiting close-to-the-maximum number of differences tend to be perceived as highly irregular.

Figure 3.2.A shows notes with four distinct ontimes. The notes are labelled from one to eight (1-4 for melody and 5-8 for bass) and placed the difference in ontime between notes `i` and `j` in row `i` column `j` of the matrix shown alongside. Such difference matrices have been studied before (Collins et al., 2010; Meredith et al., 2002; Lewin, 2007). It can be seen that there are three unique non-zero absolute difference values in the matrix, which accords with the formula given above for the minimum number, `n - 1 = 4 - 1 = 3`. This excerpt would be heard as highly regular. It is the type of regularity in schemata occurrences assumed by Symons's approach. However, most schema occurrences are not this regular, and so requiring this regularity in the application of an algorithm to pieces may result in missed occurrences.

Figure 3.2.B is at the other end of the temporal regularity spectrum. It shows notes with eight distinct ontimes which, as in Figure 3.2.A, have been

labelled 1-8 and represented by a difference matrix alongside. There are 28 unique non-zero absolute difference values in the matrix, which accords with the formula given above for the maximum number, `n(n - 1)/2 = 8*7/2 = 28`. This excerpt would be heard as highly irregular.

Figure 3.2.C contains an instance of the *Meyer* schema, and the underlined elements in the accompanying matrix indicate that the temporal relationships between: notes 1 and 2 is preserved between notes 3 and 4; notes 1 and 5 is preserved between notes 3 and 7, and notes 5 and 6 is preserved between notes 7 and 8. The exact values of the differences are not important, only that $d_{1,2} = d_{3,4} = x$, $d_{1,5} = d_{3,7} = y$, and $d_{5,6} = d_{7,8} = z$, where x, y, z are real numbers. This abstract temporal regularity is indicated by the notated elements in Figure 3.3.

The temporal–difference matrices presented above demonstrate a formal method of measuring the amount of regularity within a pair of schema–voices; to an extent, using the formula from (Foubert et al., 2017), their significance in terms of perceptive strength can be calculated. One step further, utilising the temporal–difference matrices presented above, the temporal–arrangements for various schema–types may also be represented. In the matrix representation of temporal-interval differences, cells are categorised based on the following relations:

R.1 The intra–voice inter–event distances of adjacent datapoints;
R.2 The inter–voice intra–event distances, and
R.3 The inter–voice inter–event distance of adjacent schema-events.

Table 3.3: The grouping of the temporal relations between the notes of a schema–voice pair based on their type.

0	**R.1**	(1,3)	(1,4)	**R.2**	**R.3**	(1,7)	(1,8)
	0	**R.1**	(2,4)	**-R.3**	**R.2**	**R.3**	(2,8)
		0	**R.1**	(3,5)	**-R.3**	**R.2**	**R.3**
			0	(4,5)	(4,6)	**-R.3**	**R.2**
				0	**R.1**	(5,7)	(5,8)
					0	**R.1**	(6,8)
						0	**R.1**
							0

Table 3.3 shows the aforementioned types of temporal-relations (R.1, R.2 and R.3) on the corresponding interval–difference matrices' cells for *Prinner* and *Meyer* respectively, Figure 3.3 makes visual the two pairs of the *Meyer* instance in Figure 3.2.C with red (`R.1`), blue (`R.2`) and green (`R.3`) colours. With the grouping of these cells into categories, temporal–relations are formally represented as qualitative descriptions over quantitative amounts of regularity.

To filter the sequence of schema–events, two properties that group and manipulate selected temporal relations were considered:

- Schema–voice regularity (`VR`), considering only `R.1` relations. This is a thresholded version of Symon's regularity filter, and
- Schema–event pair regularity (`PR`), considering all types of `R.x` relations but for the first and last pairs of schema-events.

The voice-regularity filter (`VR`) uses a percentage threshold for the average amount of equal intervals present on each schema–voice separately or combined (averaged). The average amount of equal intervals for a single schema–voice is calculated by dividing `1` into the cardinality of the set of `R.1` values that correspond to this specific *schema–voice*. When the filter is applied to both *schema–voices*, their individual values are added and then divided by two to derive an average. For example, the `VR` values of the examples in 3.2.A, 3.2.B and 3.2.C for melody, bass and their combination are equal to:

```
A-melody = 1 / cardinality(set([3,3,3])) = 1/1 = 1
A-bass = 1 / cardinality(set([3,3,3])) = 1/1 = 1
A-melody-bass = (A-melody + A-bass) / 2 = 1

B-melody = 1 / cardinality(set([1,2,4])) = 1/3 = 0.33
B-bass = 1 / cardinality(set([16,32,64])) = 1/3 = 0.33
B-melody-bass = (B-melody + B-bass) / 2 = 0.33

C-melody = 1 / cardinality(set([1,5,1])) = 1/2 = 0.5
C-bass = 1 / cardinality(set([3,3,3])) = 1/1 = 1
C-melody-bass = 0.75
```

The pair–regularity filter (`PR`, visualised in Figure 3.3) utilises a scoring mechanism that benefits pair temporal formations and penalises the lack of `R.1` relations.

This filter was developed to formalise the temporal form of schemata such as the *Meyer*, but can be applied to any other schema-type with even number of schema-events (e.g., *Fonte*, or *Monte*). The scoring mechanism has two configurations:

- **`Normal`**, considering the `R.1` and `R.2` relations (blue and red lines, *see* Figure 3.3). The maximum score is `4` and this occurs when the `R.1` relations (*see* Figure 3.3-red) of the first and last pair of both voices are equal (`+1` for each voice), and similarly, when the `R.2` relations (*see*

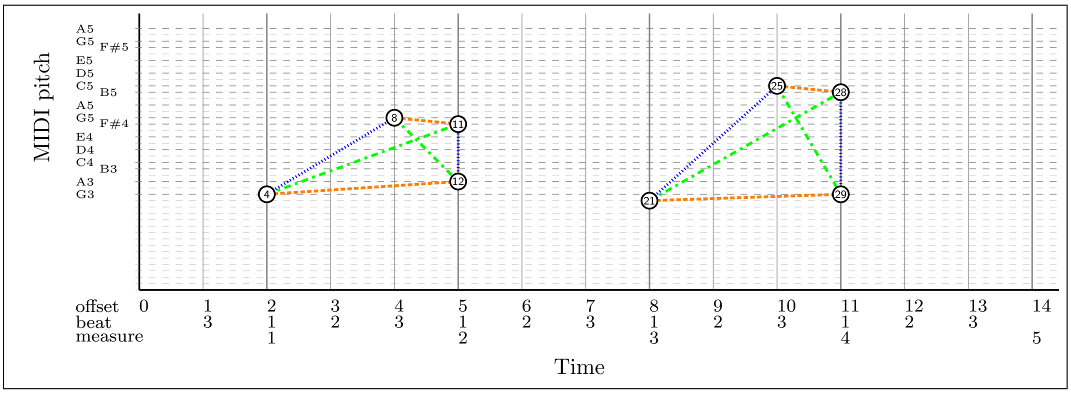


Figure 3.3: Counting regularity on intra–note–set temporal intervals.

0	***x***	•	•	***z***	•	•	•
	0	•	•	•	•	•	•
		0	***x***	•	•	***z***	•
			0	•	•	•	•
				0	***y***	•	•
					0	•	•
						0	***y***
							0

Figure 3.4: Schema-event pairs regularity form (e.g. for *Meyer* schema)

Figure 3.3-blue) of the 1st-3rd and 2nd-4th pairs of *schema-events* are equal (`+1` for each pair in score). There is a penalty of `-1` if there are no equal `R.1` relations.

- **`Extended`**, scoring has the same mechanism as `Normal` yet also considers `R.3` relations pairwise (*see* Figure 3.3–green), in a similar fashion to `R.1`.

The temporal-form scores for the examples in Figure 3.2 will be :

```
A-Normal = [R.1] + [R.2] =
 [(1,2)==(5,6) + (3,4)==(7,8)] + [(1,5)==(3,7) + (2,6)==(4,8)]  =
```

```
 [3==3 + 3==3] + [0==0 + 0==0] =
 [1 + 1] + [1 + 1] = 4
A-Extended = A-Normal + [R.3] =
 A-Normal + [(1,6)==(3,8) + (2,5)==(4,7)] =
 A-Normal + [(3==3) + (-3)==(-3)] =
 4 + [1 + 1] = 6
B-Normal = [R.1] + [R.2] =
 [(1,2)==(5,6) + (3,4)==(7,8)] + [(1,5)==(3,7) + (2,6)==(4,8)]  =
 [(1==16) + (4==64)] + [(15==60) + (30==120)] =
 [0 + 0] + [0 + 0] = 0
B-Extended = B-Normal + [R.3] =
 B-Normal + [(1,6)==(3,8) + (2,5)==(4,7)] =
 B-Normal + [(31==124) + (14==56)] =
 0 + [0 + 0] = 0
C-Normal = [R.1] + [R.2] =
 [(1,2)==(5,6) + (3,4)==(7,8)] + [(1,5)==(3,7) + (2,6)==(4,8)]  =
 [(1==1) + (3==3)] + [((-2)==(-2)) + (0==0)] =
 [1 + 1] + [1 + 1] = 4
C-Extended = C-Normal + [R.3] =
 C-Normal + [(1,6)==(3,8) + (2,5)==(4,7)] =
 C-Normal + [(1==1) + ((-3)==(-3))] =
 4 + [1 + 1] = 6
```

The above scoring mechanisms may be utilised not only for the thresholding of the exported sequence of schema–events, but also for the identification of categories of temporal arrangements. As shown in Section 2.1.1, some family–types appear in specific temporal arrangements, and the identification of their qualities in temporal regularity may facilitate the operations of discovery.

3.3.4.2 Schema–event sequence extraction

Considering the information available to this final action of the `SA` operation (Action `A6`), an inference system decides on which schema–event is added to the export sequence of schema–events. The inference system considers three factors:

- The status of the sequence, meaning the schema–events already extracted from the previous time–window (if any);
- A set of candidate extensions to the sequence, including the best–rated schema–event samples, according to transitional significance, and
- The values of temporal regularity thresholds for when the set of candidate schema–events are applied.

If the sequence is empty, the first schema–event is selected considering only positional significance. After the first schema–event, the next schema–event in the sequence is the one with the greatest transitional significance and, when added, fails to pass any of the temporal regularity thresholds that may be employed.

The method to handle the situations where schema–events of the same perceptive rating are present, forking is applied, and parallel sequences may be created. These forks of sequences are inhibited by the presence of switches that prohibit forking after a certain number of active parallel sequences that may become present. Such sequences also converge frequently, leaving only a few segments with multiple options. When only a single sequence needs to be extracted, the algorithm selects an arbitrary path.

3.3.5 Summary

The operation of schematic analysis (`SA`) offers the fundamental interpretation of music notation to the examined cognitive model in the form of the Galant musical surface. Aiming to explicate on this music–analytical operation, a thought–pattern was presented, suggesting the increases in awareness of the presence of musical structures of varying structural integration. These constructs are rated with perceptive power, concerning qualities such as positional and transitional significance, clarity, and temporal regularity. In the final act of the `SA` operation, an inference mechanism determines which schema–event is added to the exported sequence, considering the temporal regularity of the resulting sequence extension.

3.4 Example–based learning of Galant musical schemata archetypes

The operation of example–based learning can be activated whenever the model inputs musical schemata annotations. A musical schema annotation is a tuple, pairing a label of a musical schema family–class with a continuous temporal segment in music notation (in measure–ranges). Therefore, a training example is a family–class labelled segment in schematic analysis form.

Utilising the architecture of the examined cognitive model (*see* Section 3.2), the example–based learning of Galant musical schemata archetypes (`XL`) is translated into a set of interlocking routines between the `SA` operation and the `GMSC` management of family–classes. When performing the `XL` operation, information from the working memory is transferred to the long–term memory; hence, the focus shifts to the processes within the `GMSC` and the class–similarity function (`SF.CS`) of each family–class, in particular. When a training (labelled) example is to be considered by the model, its schematic analysis is converted into an exemplar, and is added to the exemplar–base of the annotated Galant family–class in the `GMSC`. Then, the class–similarity function (`SF.CS`) of the annotated family–class identifies the kinds of differences that may exist between the incoming exemplar and the archetype(s) of the labelled class, and performs accordingly one of the four modes of learning described earlier (*see* Section 2.1.2), namely: *accretion*, *tuning*, *restructuring*, or *accommodation*.

3.4.1 The thought–pattern for example–based learning

The music–analytical thought pattern for the `XL` operation is a series of decisions, considering both the outcomes of the class–similarity function over the new exemplar, and the subsequent actions that are taken. Those decisions result in one of the following modes of learning:

- Accretion, when the exemplar adds to the frequency–of–appearance of existing information;
- Tuning, when the class–similarity function is altered to include the content–related variances between the new exemplar and the (same–length) archetype(s) of the class;
- Restructuting, when a variant is added due to structure–related variances between the new exemplar and existing archetype and variants

(if any), and

- Accommodation, when the first of a kind is presented.

The following thought–pattern for the `XL` operation is a sequence of instructions that reference Cases (**C**), Questions (**Q**), Decisions (**D**), and Actions (**A**).

C1: When a training example inputs the model.
Q1: Does the annotated family–class exist in the GMSC?
D1 (**GMSC**: **FALSE**): Perform *accommodation*.
A11: Create a new family–class in the GMSC.
A111: Set the exemplar as the archetype of the class.
A112: Add the exemplar to the exemplar–base.
D1 (**GMSC**: **TRUE**): All other learning modes may apply (*accretion*, *tuning* and *restructuring*).
A12: Examine the relation between the new exemplar and the exemplar–base.
A121: Add the SA segment to the exemplar–base.
A122: Apply the class–similarity function for the exemplar–set, see **C2**.

C2: When an exemplar is added to a non–empty exemplar–base, and the class–similarity function is activated.
Q2 **SF.CS**: Has structure–related similarity threshold passed?
D2 (**TRUE**): Perform *restructuring*.
A2: Create a variant.
A21: Set archetypical form and exemplar–base.
A22: Update the class–similarity function to integrate the new structure– and content–related differences.
D2 (**FALSE**): Perform *accretion* or *tuning*, see **Q3**.
Q3: **SF.CS**: Has content–related similarity threshold passed?
D3 (**TRUE**): Perform *tuning*.
A31: Integrate the new conditional content–related difference to the class–similarity function.
D3 (**FALSE**): Perform *accretion*.
A32: Increase the frequency–of–appearance for the existing exemplar.

As shown in the instruction sequence above, the example–based learning operation of Galant archetypes concerns the categorisation of the differences between an incoming exemplar and the existing archetypes of a family–class, expressed by the class–similarity function (`SF.CS`). This simple approach in maintaining a classification of complex structures, such as the Galant archetypes, is facilitated by the highly–structured information of the training examples. The next part presents how these differences between sequences of schema–events may be classified through the class–similarity function (`SF.CS`).

3.4.2 The class–similarity function of a musical schema family–class

The class–similarity function of a family–class (`SF.CS`) extends the archetype–class–similarity function (*see* Section 3.2.1.4) by incorporating structure–related differences as the edit–operations of *addition* and *removal* of schema–events to the family–class archetype (`SF.AR`). To avoid confusion with terms and operations, the Exemplar–Difference–Function (`EDF`) that compares two exemplars (e.g., archetypes and variants), yields results expressed as a sequence of schema–event differences (the Exemplar–Difference–Vector, `EDV`, *see* example below). In the `XL` operation, the class–similarity function of a family–class (`SF.CS`) calculates the exemplar–difference–vector (`EDV`) of each new training example, and through the `XL` thought–pattern, it integrates them into a hierarchical representation of differences.

Utilising edit–operation statements in combinations with the content–related conditional statements, the class–similarity function of a family–class may express all its variants through a single exemplar. This representation is particularly useful when examining the class of an exemplar: if that does not return a clean `EDV` result from the class–similarity function of a family–type, i.e., `EDF(S,E) = 0` (*see* below), then it is not of that sort. In brief, the `SF.CS` function expresses the exemplar–base of a family–class through an archetype–exemplar and a collection of hierarchical conditional statements of edit–operations.

The following example displays the transformations of the class–similarity function when aggregating variant exemplars from the *Romanesca* family–class. Each row is one of the followings:

`A`: The archetype of the family–class;

`E`: The incoming exemplar;

`S`: The status of the class–similarity function, integrating the representations of variants hierarchically;

`V`: The variant, expressed as edit–operations on the archetype (`A`), with details about the differences of their schema–events, and

`EDV`: The Exemplar–Difference–Vector (`EDV`), resulting from `EDF(E,E)`.

The edit–operations are represented by the following symbols, before the vector of content–related differences between schema-–events:

=, same valued event or content;

v, only voicing differences (inversions of the same harmonic class events);

h, harmonic class difference;

+, addition of an event, and

-, removal of an event.

The weight of importance for each of the above operations is hierarchically assigned. Since the operations of *addition* and *removal* change the structure of the archetype, these are considered to be more important than the other two content–related differences. This means that any count of these operations is considered always to be greater than content–related differences. No differentiation of importance between *addition* and *removal* is considered. When content–related differences are considered, these are also hierarchical,

Table 3.4: Example integration of class–similarity differences for the *Romanesca* family–class. The values of each schema–event are for: bass, harmony, melody.

	Schema–events					
	1	2	3	4	5	6
A	1, 53, 1	7, 63, 5	6, 53, 1	3, 63, 1		
E_1	1, 53, 3	5, 53, 2	6, 53, 1	3, 53, 7	4, 53, 6	1, 53, 5
V_1=EDF(A, E_1)	v (= , = , +2)	v (-2, -1i, -3)	=	h (=, +2c, -1)	+ (4, 53, 6)	+ (1, 53, 5)
E_2	1, 53, 3	7, 63, 5	6, 53, 1	5, 63, 7	4, 53, 6	3, 63, 5
V_2=EDF(A, E_2)	v (= , = , +2)	v (=, =, -3)	=	h (+2, +2c, -1)	+ (4, 53, 6)	+ (1, 53, 5)
EDF(V_1, V_2)	=	v (+2, +1i, =)	=	v (+2, =, =)	=	v (+2, +1i, =)

valuing differences in the change of harmonic class more highly than inversions and voicing variations. These ratings considerations are applied when forming the class–similarity function (`SF.CS`).

Aggregating ***Romanesca*** variants

Table 3.4 shows the differences between the family–class archetype (`A`, the *Galant Romanesca*) and incoming exemplars, E_1 (*Leaping bass* variation) and E_2 (*Step–wise bass*) variation. In the first comparison, EDF(A,E_1), the two exemplars have: two schema–events with equal harmonic class (1 and 2), one with the same content (3), one with different harmonic class (4), and the addition of two new (5 and 6). The total number of edit–operations in this comparison are therefore: two additions, one major alteration, and two minor alterations. At this point, since this is the first variant, the class–similarity function simply incorporates the `EDV` vector.

Next, the new exemplar is compared through the `EDF` function against both: the archetype, and any variant of the annotated schema–family–type, through the `EDV` vectors incorporated in the family–class–similarity function (`SF.CS`). With this method, if a variant (`V`), in the form of an `EDV` in the `SF.CS`, yields a lower `EDV` score in edit–operations than that of the archetype, i.e., EDF(V,E) < EDF(A,E), then the new exemplar is expressed as an EDV on that variant. This method connects the variants of a family–class by creating hierarchical relations through considering minimal edit–operations in their alterations. In the example case presented above, this transformation of the SF.CS function is observed when the second exemplar (E_2) is compared against the archetype (A) and the SF.CS function (through its variant representations, in this case V_1). Counting the edit–operations of the EDF(A,E_2), the result is the same as with E_1. When counting the edit–operations of the EDF(V_1,E_2), the result is three minimal content alterations. Therefore, the new exemplar (E_2) is integrated into the SF.CS representation as an EDV of EDF(V_1,E_2).

To summarise: in the XL operation, the class–similarity function of a family–class (SF.CS) and its schema–event sequence difference–function (EDF) offer a straightforward and comprehensible mechanism for the representational integration of variant exemplars. This representation is utilised in the recognition process when the classification task is performed (*see* Chapter 5).

3.4.3 Summary

The example–based leaning of Galant archetypes (XL) can be performed by the model whenever annotations are provided. In that case, after the SA operation, the labelled segments in schematic analysis form are converted into exemplars, and are added to the exemplar–base of the annotated family–class. This triggers the class–similarity function SF.CS that compares the newly considered exemplar, on the one hand, and on the other the existing archetype and variants (if any). The SF.CS compares exemplars utilising an Exemplar–Difference function (EDF) that identifies and categorises the differences between two sequences of schema–events, resulting into an Exemplar–Difference–Vector (EDV). Following an analytical thought–pattern, decisions upon the types of differences are taken, leading to four modes of learning (i.e., *accretion*, *tuning*, *restructuring*, and *accomodation*). Regardless of which learning mode is performed, the EDV of a training example is integrated into the GMSC, either creating a family–class, or by adding to or altering the annotated SF.CS.

3.5 Discovery of Galant musical schemata

The operation of Galant musical schemata discovery (SD) concerns the extraction and maintenance of Galant family–classes into the long–term representation of the GMSC. To perform the SD operation, the examined model utilises and extends the high–level operations of schematic analysis (SA, *see* Section 3.3) and example–based learning (XL, *see* Section 3.4). In two main stages, the SD operation first processes a score in an attempt to segment and categorise SA segments and exemplars, then to integrate these into the long–term classification of the GMSC. The SD operation is, therefore, considered as the integration/concentration of score–wise classifications of SA segments into family–classes in the GMSC. As such, the music–analytical thought–pattern of this operation models the notion of *equilibrium* within the long–term Galant musical schemata in the GMSC; it acts as the regulatory logic that selects which learning mode should be performed whenever a repetitive SA–segment is found, considering qualities such as the diversity among discovered family–

classes and the variability among their exemplars.

In the first stage of the `SD` operation, the `SA` operation is extended to perform the Intra–Score Exemplar Discovery process (`SA.ISED`, *see* Section 3.5.1, below). The `SA.ISED` process inputs the schematic analysis form of a score (`SA.SS`), and attempts to segment it into non–overlapping exemplars. The steps of the `SA.ISED` process over the schematic analysis of a score involve: the creation of a discovery–space of possible exemplars; the identification of *similar* non–overlapping exemplars (if any), the identification of already discovered family–types (if any), and the extraction of a plausible segmentation of non–overlapping `SA`–segments and discovered/found exemplars, according to a set of preference rules for musical schemata progressions (*see* Section 3.5.1.4). The `SA.ISED` process results in at least one segmentation/classification (`SA.CLE`), that of the complete score in `SA` form. In the case where repetitive segments or exemplars are found, these are stored as family–classes (`SA.CLE.CL`), and any non–repetitive ones into a set of 'uncategorised' exemplars (`SA.CLE.UN`). When performing the `SD` operation, therefore, the available information from a score is now a plausible segmentation/classification of the score into 'uncategorised' `SA`–segments, and repetitive exemplars (if any) either score–wise or in relation to already discovered family–classes.

In the second stage of the `SD` operation, the `GMSC` integrates the outcome of the score–wise segmentation/classification process (`SA.ISED`) through the Exemplar–Recognition–Routine (`ERR`). Considering the above enhancements, the discovery–space of the model, where exemplars may be matched/discovered, consists of the following repositories:

- The (discovered) family–classes in the GMSC (GMSC.SF);
- The 'uncategorised' SA–segments in the GMSC.UN;
- The Intra–Score Exemplar–Classification of the SA (SA.CLE), including:
  - Discovered family–classes (SA.CLE.CL, if any), and
  - 'Uncategorised' SA–segments (SA.CLE.UN).

The repositories with discovered family–classes (GMSC.SF and SA.CLE.CL) are similar to the GMSC from the XL operation, each storing family–classes (GMSC.SF) with an archetype, its exemplar–base and a class–similarity function. The repositories with 'uncategorised' SA–segments (X.UN) simply aggregate those SA–segments that do not match with neither other parts of the score nor discovered family–classes.

Utilising the aforementioned repositories, the model integrates exemplars into the GMSC through the Exemplar–Recognition–Routine (ERR). The ERR compares the contents of the four repositories in pairs, aiming to discover any repetitive exemplars. If a pair of similar exemplars is discovered, according to a discovery–similarity threshold, the GMSC integration routine is activated, calling in turn the process of *equilibrium* to decide which learning method should be performed. After the comparisons of the contents of all of the repositories, the corresponding repositories are merged into the GMSC: the GMSC.SF integrates similar and repetitive exemplars from the SA.ISED operation (in the SA.CLE.CL), and the GMSC.UN integrates those 'uncategorised' SA–segments of the SA.CLE.UN. These two stages are repeated whenever a score inputs the model.

The rest of this section describes the processing steps of the above functions, ending with the methodological explication of the SD operation through the examined model architecture and its operations.

3.5.1 The Intra—Score Exemplar—Discovery process

The Intra—Score Exemplar—Discovery process (`SA.ISED`) inputs the schematic analysis representation of a score (`SA.SS`), and outputs a segmentation of non–overlapping `SA`–segments and exemplars (`SA.CLE`). This transformation is achieved through the following steps:

1. The creation of the score–wise discovery–space of exemplars;
2. The identification of *similar* and non–overlapping exemplars from the discovery–space, according to a discovery matching threshold (`DMT`);
3. The identification of exemplars *similar* to discovered family–classes, according to a classification matching threshold, and
4. The extraction of a segmentation/classification of the `SA` form, according to a set of preference rules for exemplar relations.

3.5.1.1 Creating the discovery–space of exemplars

The discovery–space of exemplars is extracted by the `SA` representation of a score with the application of a grouping mechanism that considers the following parameters:

- A range of minimum and maximum number of schema–events;
- The maximal temporal interval between two successive schema–events of the same exemplar, and
- A temporal regularity threshold.

Considering the properties of the Galant musical schemata, the range of minimum and maximum schema–events for an exemplar–sample was considered to be between three and six. The maximal temporal interval between two successive events of the same exemplar is set adaptively, corresponding to extracted features such as the harmonic and chordal density. The temporal regularity threshold that was selected to filter the generated schema–event sequences is the *total* –average to the length of schema–events– threshold utilised in the `SA` operation (*see* Section 3.3.4.1). The generated samples of exemplars form the discovery–space of the `SD` operation.

3.5.1.2 Identification of repeated exemplars

The next step in the `SA.ISED` process attempts the identification of similar and non–overlapping exemplars in the discovery–space. The sampled exemplars are first categorised based on their length (in schema–events). The exemplars of each group are then compared in pairs, creating a matrix of Exemplar–Difference–Vectors (`EDV`). Applying the Discovery–Matching–Threshold (`DMT`) on each `EDV`, repetitive exemplars may be found. The exemplars of a matching pair are then tagged and stored in the `SA.CL` repository to be utilised by the final stage of the `IS.ISED` process to decide upon which segmentation of the score to extract.

3.5.1.3 Identification of discovered Galant family–classes

After the attempt to identify any non–overlapping intra–score exemplar repetitions, the next step classifies the exemplars of the discovery–space according to already discovered family–classes. Considering same schema–event length, each sampled–exemplar inputs the family–class function of a (previously discovered) family–class in the `GMSC` (if any), yielding a list of Exemplar–

Difference–Vectors (EDV). From the resulting list of EDVs, if the minimum EDV is below the classification–recognition threshold, then the exemplar in the discovery–space is also tagged with the discovered family–class. The classification of exemplars at this stage aims to facilitate the extraction of a segmentation for the examined score, and matching exemplars found in this stage are added later to the GMSC on the second stage.

3.5.1.4 Extracting segmentations of exemplars

The final step of the SA.ISED process utilises the exemplars of the discovery–space and the extracted information concerning possible exemplar repetitions and similarity with the family–classes in the GMSC to select a segmentation of non–overlapping exemplars. The segmentation task is considered a point/score optimisation problem where each exemplar of the discovery–space is rated based on its relation with other and with the GMSC. For example, those exemplars that match with previously discovered family–classes in the GMSC may be valued the highest, followed by those with intra–score repetitions, with any 'uncategorised' exemplars adding no points. Thus, the segmentation algorithms creates sequences of non–overlapping exemplars, preferring those exemplars that are tagged as similar to those in the GMSC (if any), then those repetitive in the discovery–space. The segmentation with the highest score is selected. In the case where no repetition or similarity with the GMSC is found within the exemplar discovery–space, as is often the case where small excerpts are considered, then the complete SA–segment is added to the SA.CLE.UN repository.

Summarising, the SA.ISED process extends the SA operation by attempt-

Table 3.5: The indices of the pairs of repositories compared by the Exemplar–Recognition–Routine. GMSC.SF maintains family–classes in the GMSC, GMSC.UN maintains 'uncategorised' SA–segments from previous SA.UNs repositories, SA.CL contains the (score–wise) repeated exemplars, SA.UN contains the (score–wise) uncategorised exemplars. Numbers do not indicate priority.

	GMSC.SF	GMSC.UN	SA.CLE.CL	SA.CLE.UN
GMSC.SF	1	2	3	4
GMSC.UN		5	6	7
SA.CLE.CL			8	9
SA.CLE.UN				10

ing to segment the SA representation of a score into a sequence of non–overlapping exemplars.

3.5.2 The Examplar–Recognition–Routine

The Exemplar–Recognition–Routine (ERR) integrates the outcomes of the SA.ISED process into the long–term representation of the GMSC. The ERR process compares the contents of all four repositories in pairs, resulting in ten pairs of repository comparisons (indexed in Table 3.5). When any two repositories are compared, a matrix of minimal Exemplar–Difference–Vectors is generated (EDV). The discovery of family–classes occurs when any of the resulting values from a repository comparison (EDVs) passes the Discovery–Matching–Threshold (DMT). Then, the GMSC integration routine is activated (GMSC.IR.D) and the process of *equilibrium* (GMSC.EQ) determines the learning mode (*see* below). Each pair of repositories is compared under different occasions. All intra–repository comparisons (cases 1, 5, 8 and 10 in Table 3.5) are activated when their contents are altered, i.e., after the discovery of an exemplar–pair in an inter–repository case (cases 2, 3, 4, 6, 7 and 9 in Table 3.5). After the completion of the ERR process, any remaining SA–segments in the SA.CLE.UN are merged into the GMSC.UN, and the two stages of the SD operation repeat.

The ERR function may by summarised by the following equation:

$$ERR(R^1, R^2) = \sum_{i=1}^{|R^1|} \sum_{j=1}^{|R^2|} EDF(R_i^1, R_j^2)$$

where

ERR is the Exemplar–Recognition–Routine (ERR),

$R^1, R^2 \in$ [GMSC.SF, GMSC.UN, SA.CLE.N.CL, SA.CLE.N.UN],

EDF is the Exemplar–Difference–Function (EDF) yielding Exemplar–Difference–Vectors (EDVs), and

R_n^X is the n^{th} Exemplar of the X repository.

The EDVs of the above matrix indicate the difference between the exemplars of any two repositories (in pairs). The EDF function is directional, but since the differences of the pitch values and the harmonic distance are complementary, only the upper triangle of the generated matrices is considered, converting values into their minimal form.

3.5.2.1 The GMSC Integration–Routine

The GMSC Integration–Routine (GMSC.IR.D) is the core process of the SD operation as it alters the state of the discovered family–classes (GMSC.SF). The GMSC.IR process is activated whenever a new family–class is discovered by the matching of two exemplars during an ERR process. Any newly discovered family–class (exemplar–pair) is compared with existing family–classes (in the GMSC.SF) resulting in a set of Exemplar–Difference–Vectors concerning structure– and content–related differences. In the `XL` operation, due to the presence of annotations, the results of such `EDV`s were directly related to learning modes (i.e., *accretion*, *tuning*, and *restructuring*). In the `SD` operation, the resulting `EDV`s are interpreted through the `GMSC.EQ` process, considering the state of the existing classification of Galant family–classes.

When performing the `SD` operation, the `GMSC` aggregates classifications of score–wise exemplars (`SA.CLE`) from the results of the `SA.ISED` process on `SA` forms of music notation. This constant increase of exemplars may erode the family–class–similarity function of a Galant family–class by gradually expanding its threshold. This may lead to uneven distributions of the exemplars into family–classes, creating highly variable family–classes that integrate most of the exemplars found. In another extreme, there is the possibility of populating the classification with similar family–classes whose exemplars do not match because of borderline crossing of their variability thresholds.To prevent such issues and in an attempt to formalise the maintenance of the classification of long–term Galant musical schemata, the process of *equilibrium* utilises two thresholds that relate to the qualities of the *diversity* of the discovered family–classes within the `GMSC`, and the *variability* within each of those family–classes.

The *diversity* of the discovered classification of family–classes is expressed by the minimum Exemplar–Difference–Vector among them. This minimum `EDV` indicates the lowest differentiation among the Galant family–classes in the `GMSC.SF` repository. In a similar manner, the *variability* of a discovered family–class is found by calculating the maximum `EDV` among an archetype/variant and its exemplar–base. This maximum `EDV` indicates the greatest differentiation among the exemplars of a family–class in the `GMSC.SF` repository. Both of the above thresholds may be expressed with conditional statements that are fixed, relative to specific values, or adaptive to certain conditions of the classification.

Therefore, whenever the `GMSC` Integration–Routine is activated (`GMSC.IR`), the process of *equilibrium* (`GMSC.EQ`) examines the outcome of each potential learning mode, and prefers those that do not pass the thresholds of diversity and variability. The explication of the `SD` operation describes the logic and the actions of the `GMSC.EQ` process, when determining on which learning mode to perform.

3.5.3 The music–analytical thought–pattern for the discovery of Galant archetypes

The enhanced operations of the examined model facilitate the explication of the `SD` operation, and the latter is expressed as the examination of three main cases:

- **C1** The processing of a new score;
- **C2** The synchronisation of repositories through the Exemplar–Recognition–Routine (`ERR`), and
- **C3** The integration of discovered family–classes through the process of equilibrium (`GMSC.EQ`).

Similarly to the previous thought–patterns (*see* Section 3.3 and Section 3.4) the sequence of instructions references Cases (**C**), Questions (**Q**), Decisions (**D**), and Actions (**A**).

C1: Considering a new score.
A1: Perform Intra–Score Exemplar Detection (`SA.ISED`).
A11: Create the discovery–space of exemplars.
A12: Identify repetitive non–overlapping exemplars.
A13: Identify discovered family–classes.
A14: Find optimal segmentation.
A15: Activate Case **C2**.

C2: Synchronisation of repositories.
A2: Perform repository comparisons.
A21: Perform inter–repository comparisons (*see* Table 3.5, cases 2, 3, 4, 6, 7, 9).
D21: If an exemplar–match is discovered.
A211: Activate Case **C3**.
D211: If inter–repository comparisons are completed.
A212: Merge corresponding repositories (`SA.CLE.CL` with `GMSC.SF` and `SA.CLE.UN` with `GMSC.UN`)
A22: Perform intra–repository comparisons (*see* Table 3.5, cases 1, 5, 8, 10).
D22: If an exemplar–match is discovered.
A221: Activate Case **C3**.

C3: The integration of discovered family–classes.
A3: Find the minimal Exemplar–Difference–Vectors (`EDV`s) for the newly discovered family–class and all of the previously discovered family–classes in the `GMSC.SF`.
D3: If all generated `EDV`s do not pass the classification similarity threshold.
D31: Consider the `GMSC.EQ` diversity lower threshold.
D311: If adding the new class does not pass the threshold.
A31: Create a new family–class in the `GMSC` performing the *accomodation* learning mode and exit.
D312: If adding the new class passes the threshold.
A312: Add the new class to the discovered family–class with the lowest `EDV` and exit.
D4: If there exist `EDV`s within the classification similarity threshold.
D41: If the matching is exact.
A41: Perform *accretion* and exit.
D42: If the matching is approximate.
A42: Consider the `GMSC.EQ` variability upper threshold for the family–classes with minimal `EDV`.
D421: Exclude the family–classes whose `EDV` exceeds the variability upper threshold.
D43: If there exist family–classes in the `EDV` matrix.
A43: Select the family–class with minimum distance and apply *tuning*.
A5: Exit to Case **C2**.

The above thought–pattern is examined in Chapter 6 with the task of Galant archetype extraction.

3.5.4 Summary

The discovery of Galant musical schemata archetypes is modelled by the SD operation of the examined model. The SD operation utilises and enhances the two previously examined operation of schematic analysis and example–based learning, for the gradual adaptation of the long–term musical schemata representations to incoming score–wise classifications. In two stages, the SD operation first, performs score–wise analysis to identify potential segmentations of the score into exemplars (through the SA.ISED process, *see* Section 3.5.1), then compares the findings of the segmentation with existing information (through the ERR process, *see* Section 3.5.2) to integrate them into the GMSC (through the GMSC.IR.D process, *see* Section 3.5.2.1). This constant aggregation of score–wise classifications is regulated by the process of *equilibrium* (GMSC.EQ), determining which learning mode to apply for a discovered family–class, considering the diversity and the variability of the discovered classification.

3.6 Summary

This chapter presented the cognitive model designed to facilitate the explication of the analytical processes involved in the automatic identification of musical schemata. The first section described the architecture of the model, and the separation among the music–notation (MD), the long–term memory classification of Galant musical schemata (GMSC), and the operations of the working memory with the schematic analysis operation (SA). The next three sections described the three high–level music–analytical operations that gradually perform more autonomous musical schemata identification operations. First, the operation of schematic analysis (`SA`, *see* Section 3.3) was explicated through a series of analytical operations that increase the awareness of (Galant) musical schemata elements, and eventually, extract the `SA` form of a score, i.e., a sequence of schema–events. Next, the operation of (Galant archetype) example–based learning was presented (`XL`, *see* Section 3.4), analysing and formalising the learning steps through a composite representation for Galant archetype family–classes. The final operation of Galant musical schemata discovery (`SD`, *see* Section 3.5), utilised and expanded the previous two high–level operations to accomodate 'uncategorised' exemplars, and through a method that maintains specific conditions among the discovered classes (the `GMSC.EQ` process), employs the learning methods of the `XL` operation to discover and maintain Galant family–classes. All three high–level operations were explicated with thought–patterns, a sort of analytical sequences of instructions, suggesting a high–level methodology for each operation. The model and its operations is examined through computational implementations in the following three chapters (*see* Chapters 4, 5, and 6).

4 Galant musical schemata search

This chapter presents the task of identifying Galant musical schemata in scores. The method was employed to evaluate the efficacy of the schematic analysis operation (`SA`, *see* Section 3.3) in extracting the most (perceptually) prominent sequence of schema–events from music notation. The first section describes the search task within the examined model architecture, and how its outcomes are utilised for the evaluation of the `SA` operation. The next section presents the computational implementations of the `SA` operation and the search workflow, through the `AES` system (*see* Section D.2). In the following section, the parameters and the results of the computational experiments that tested the `SA` operation are shown. The last section discusses findings regarding the effects of the grouping mechanisms in the performance of the `SA` operation.

4.1 Task description

In the examined cognitive model architecture, the operation of the schematic analysis is performed through the gradual identification and integration of musical structures that are created and rated by models of powers of perception. To examine whether this complex operation produces the intended outcome, the performance of the `SA` model was measured by searching for annotated Galant archetype instantiations in schematic analysis representation:

- If an annotated Galant archetype instantiation is found (a *true–positive*), that indicates good performance from the `SA` operation;
- If an annotated Galant archetype instantiation is not found (a *false–positive*), that is regarded as a performance issue of the model, or its computational implementation.

If most of the annotated Galant archetypes are indeed found, that would signify efficiency from the `SA` operation in reducing the music notation, leaving out non–structural/schematic notes. However, if most of the annotated Galant archetypes are not found, this indicates that either the model or its computational implementation has issues, possibly excluding information that it should not ignore. By performing the `SA` operation with different configurations, the parameters of the `SA` operation could be examined both in isolation and in combination with others.

Since this task examines the performance of the `SA` model and not the identification process *per se* (*see* Chapter 5), only the 'exact' matching of the Galant archetypes was considered. The exact matching of the Galant archetypes in the `SA` representation means that all of the pitch–related properties of the sequences of schema–events of the former must be equal to schema–events found in the latter. Within the context of the examined model architecture, this identification process may be viewed as an extreme top–down bias, where only 'known' information, i.e., the exemplars in the `GMSC`, is considered. More specifically, when searching for an archetype representation in the schematic reduction form of a time–window, i.e., the identification process, a search–space is created, comprising of all same–length (in schema–events) sequences that may be compared with the archetype. The only parameter through which to create the search–space of the identification process is the maximum temporal interval between two successive events. Then, each sequence from the search–space is examined by the class–similarity function of an archetype to identify an exact match. This 'local' time–window of the identification process is gradually moving towards the end of the time–window information–state of the 'active–plane', until a new one is to be processed; the process thus repeats.

4.2 Computational implementations by the `AES` system

As stated in the Introduction, the examined cognitive model is computationally implemented by the `AES` system (*see* Section D.2). In brief, the `AES` system performs high–level workflows (operation–modes) that control the information exchange among three classes: music notation and annotations (`MD`), the classification of Galant musical schemata archetypes (`GMSC`, *see* Section 3.2.1), and the operation of schematic analysis (`SA`, *see* Section 3.3). Low–level operations with symbolic music information are facilitated by a library made for this study (*see* Section D.1). The current section describes the computational implementation of the `SA` operation and its parameterisation. Then, the search workflow and its parameters are shown.

4.2.1 Implementing the schematic analysis operation

As described in the previous Chapter, the `SA` module simulates the working memory by incorporating a mechanism to access the model–wise information–states (through the 'active plane', *see* Section 3.2.2.1) in order to perform the schematic analysis operation. The `AES` system implements the `SA` module with a class (`AES.SA`) that updates its information states (`AES.SA.AP`) by interacting with the `GMSC` and the `MD` classes. Having access to a score element, the first stage in the `SA` operation (`AES.SA.ScA`) concerns the extraction of tonality and rhythmic context, and the identification of the outer voices and harmonic segments. The second stage (`AES.SA.SA`) is where samples of schematic structures are extracted and selected to form the final output of the operation, the sequence of schema–events (`SA.SS`). The following parts present the implementation details of these stages.

4.2.1.1 The score analysis stage

An `AES.SA` object initiates with a score element (in MusicXML, Good, 2001) and its accompanying annotations (if any). The information of the music notation is then converted into two operable representations: datapoints and 'minimal segments' (*see* Appendix D.1.1). A datapoint is a vector of values regarding the temporal and pitch properties of a note, i.e., the measure and position within it, and an absolute pitch value. Minimal segments are continuous segments of the score with stable pitch properties. Both of these representations are extracted from scores utilising the `music21` framework (Ariza and Cuthbert, 2011). Thus, a datapoint storing note information has (at minimum) the following values:

```
[ measure, beat, duration, pitch ]
```

A minimal segment is storing information about its position in the score, and the notes present.

```
[ measure, beat, duration, [pitch-list] ]
```

The above representations are utilised throughout the `SA` operation.

The first step in score analysis concerns the extraction of score–wise tonality and rhythmic organisation. Tonality is extracted considering key–signature readings from the score, in combination with a probe–tone mode detection algorithm (Krumhansl, 1990). At this point, both datapoints and minimal segments may be represented by their pitch–related information in scale–degrees, and temporal–positions with metric–strengths.

The next step in score analysis concerns the extraction of the outer melodic movements and the harmonic segmentation. The outer melodic movements are found by simply considering the outer–notes of each 'minimal–segment'. To maintain clarity between the two melodic movements, a mechanism from stream–segregation was adopted (Katsiavalos and Cambouropoulos, 2012, *see* Section D.1.2). When selecting the proceeding note for any of the outer voices, a voicing mechanisms measures the 'distance' of possible following notes within a temporal time–window, by counting their pitch and temporal interval. Then, it selects the next note for the voice considering the one with the lowest 'distance'. By considering different weights for the pitch and temporal distances, an equation between the maximal pitch–difference and the maximal temporal–difference is formed, acting as a threshold ratio. For example, by equating the temporal interval of two measures with the pitch interval of an octave, this means that if the melody movement stops and the bass movement continues, the melody movement will prefer selecting notes that are further distant in time, but closer in pitch. This means that cross voices, assuming that these are at least one octave apart, are highly unlikely.

The harmonic segmentation is performed by the *HarmAn* algorithm (Pardo and Birmingham, 2002). The *HarmAn* algorithm aggregates 'minimal segments' and compares their combined pitch–profile with templates of triadic chords. The segmentation process occurs through a scoring mechanism that calculates the total duration of the pitch–classes present in a segment. If a segmentation threshold is passed, a new harmonic segment is created and labelled according to the aforementioned mechanism.

At this point, the `AES.SA` has access to the following information:

- Music notation in two operable representations, datapoints and 'minimal segments';
- Score–wise tonality and rhythm properties;
- Outer melodic movements, and
- Harmonic segments.

The score analysis stage of the `SA` operation concludes with the embedding of the extracted information into the operable representations of datapoints and minimal segments. Therefore, the datapoints now include information regarding harmonic and voicing context:

```
[ measure, beat, duration,
  scale-degree, metric-strength,
  voice, harmony ]
```

Similarly, scale–degrees, voicing and harmonic information is embedded into minimal segments.

4.2.1.2 Implementation and parameterisation of the schematic analysis stage

The next stage of the SA operation is where the schematic elements of schema–events, –voices, –voice–pairs, and eventually schema–event progressions are created. Each schematic structure is created by its own grouping mechanism (*see* Appendix D.2.1.3), adapting to the perceptual models of positional and transitional significance and temporal regularity (*see* Section 3.3). Those grouping mechanisms are utilising the information obtained from the score analysis stage (`AES.SA.ScA`) for the extraction of features that may make their operation adaptive to the local context of a time–window.These features describe the temporal and pitch qualities of the music notation that is found on an examined time–window as follows:

- Chordal (vertical) density;
- Temporal (horizontal) density, and
- Harmonic diversity.

Chordal density expresses the average note cardinality that is found on the minimal segments, also considering their metric strength. Thus, chordal density is represented with three values: an average cardinality for the complete time–window, and two averages for the cardinality of the minimal segments in strong– and weak–metric positions. This quality is utilised to apply adaptive weights to positional significance (*see* below). Temporal density expresses the average number of onsets *per* measure. This quality is calculated for the active time–window, considering minimal segments, and for each voice, considering the onsets per measure of each melodic movement separately. Harmonic diversity expresses the number of harmonic qualities that are found in the time–window, and the ratio of their temporal duration.

The above features are utilised by the grouping mechanisms to adjust the weights of positional and transitional significance, resulting in behaviour adaptive to the local (temporal) contexts.

Regarding the parameterisation of the `SA` operation, the following factors were tested with different configurations:

- Schematic integration (*harmonic*, *complete*);
- Positional significance (*none*, *fixed*, *adaptive*);
- Transitional significance (*none*, *fixed*, *adaptive*), and
- Regularity thresholds (*none*, *strict*, *total*, *extended*).

The parameter of the schematic integration of the `SA` operation concerns the information considered when processing/matching a schema–event. Two kinds of schematic integration were tested considering only the harmonic information of schema–events, and the complete schematic integration of a schema-event, comprising harmonic and voicing information. This parameter enables the comparison of the `SA` performance between the more generalised harmonic–only information of schema–event, and the more specific complete schematic integration of schema–events that includes both harmonic and voicing information.

As stated in the previous chapter (*see* Section 3.3), the positional significance relates to the metric position of an element. To examine the effect of this parameter within the grouping mechanisms of the `SA` operation, it was considered *in absentia*, both with fixed weights, and with adaptive weights. When adaptive weights are utilised, the weights are adjusted according to values of temporal density. If temporal density is low, e.g., a chord *per* measure, the positional significance is decreased, under the assumption that there is not enough musical content to reduce, and that all content present is equally significant. On the other hand, if temporal density is high, e.g. a chord *per* beat, the positional significance is increased, aiming to distinguish between ornament and passing notes and chords, and potentially schematic content.

The parameter of transitional significance is adjusted through taking into consideration all of the three 'contextual' features presented above. Lower chordal density increases the weights of transitional significance, under the assumption that the content is sparse. Higher temporal density decreases the weight of transitional significance, assuming that more chords are present, possibly ornamentations and/or passing. The same assumption is made when harmonic diversity is also high, and transitional significance is also lowered.

The temporal regularity thresholds were tested with four configurations:

- *none*, *in absentia*, where no such thresholds are considered;
- *strict*, where only isochronous relations are considered;
- *total*, when a single threshold is used for all temporal relations, and
- *extended*, adding specificity to the 'total' threshold above, permitting only specific temporal forms.

As shown in Table 4.1, examining selected combinations of values for the above parameters, the performance of the SA operation, and the impact of each parameter, were examined. The rest of this segment describes the algorithms for the extraction of each structure.

Extraction of schema–events . The schema–event extraction algorithm begins with a minimal segment (base), and examines its relation with other minimal segments in a local temporal window (pivots), less than a measure ahead, creating pairs of base–pivot minimal segments. For the pair of minimal segments (base–pivot), the combined positional significance and the transitional significance are found with the formulæ shown in Section 3.3. To avoid the extraction of all of the possible schema–events that may spawn from this schema–event sampling mechanism, two thresholds were applied. A positional significance threshold filters out pairs of minimal segments of which the combined positional significance is below the average of the examined time–window. If the transitional threshold of a base–pivot pair is passed, implying a change in harmony, then the pairing process stops, allowing only previous pairs of minimal segments to be considered as possible schema–events. Applying the above operations on the minimal segments of a time–window results in a set of schema–event samples, rated with positional significance.

Extraction of schema–voices . The schema–voice extraction algorithm inputs the list of datapoints from any of the outer–voices, and updates the properties of each datapoint to include a potentiality score for participation on the extracted schema–voice. This schema–voice potentiality score is based on their respective positional significance, transitional significance, and regularity. This reduction process initiates by considering the first three datapoints in order to determine which one has the maximal positional significance, and to be the first of the sequence. In the case where the `SA` continues the sequence from previous extraction, the process continues from the last datapoint.

After deciding on the first note of a schema–voice in a time–window, the next datapoint is selected with the application of its positional and transitional significance, and the temporal regularity it adds to the trail of previously selected datapoints. The positional and transitional significance of a melodic pair of notes is found similarly to when calculating minimal segments, also applying similar thresholds. Additional processing includes the regularity thresholds that are applied when considering the trail of datapoints that is created if an incoming datapoint is accepted. Since the algorithm does not extract a single (perceptually) valid sequence, but reduces the music notation to allow all (perceptually) valid sequences, this regularity filter may not hold true when all previous datapoints are considered. What is important in this process is to find at least one set of previous datasets that will incorporate the incoming datapoint and will not pass the regularity threshold.

Extraction of schema–voice–pairs . The algorithm for the extraction of schema–voice pairs inputs the lists of datapoints for the melodic and bass movements, and updates the schema–voice potentiality value of their datapoints considering cross–voice combinations with temporal regularity thresholds. Similarly to when extracting each schema–voice, the first datapoints for the movements of melody and bass are selected. Then, the algorithm rates all possible combinations between the datapoints of these two voices, adding to the potentiality of each datapoint whenever it appears on a valid combination, according to the temporal regularity filter.

Extraction of schema–event–progression . In the final step of the schematic analysis stage of the schematic analysis operation, the final sequence of schema–events is extracted. The first event is selected similarly to the first note of a schema–voice, as described above. Then, the next schema–event to be added to the sequence is selected based on the combination of highest scores in transitional significance, and the combined total scores of voice potentiality of the notes in the schema–events. The method is similar to when extracting schema–events, with a base schema–event creating pairs with schema–events within the local temporal context.

4.2.2 The `AES` search workflow

The search workflow of the `AES` system begins with the initialisation of a `GMSC` object, loading the set of targeted Galant archetypes, and an `SA` object, loading configurations regarding the schematic analysis operation. Next, the `SA` object inputs score elements and annotations, to perform schematic analysis according to the selected configurations. After the completion of the `SA` operation upon a time–window, the archetypes stored in the `GMSC` are then searched for on the extracted sequence of schema–events.

In the identification process of the search task (*see* Appendix D.2.2.1), the pitch–related properties of the Galant archetypes in the `GMSC` are projected onto the sequence of schema–events, aiming to find equally valued schema–events, also in the correct order. The only parameter of this 'strict' recognition process concerns the maximal distance between two successive schema–events. Therefore, the identification of a Galant archetype is the matching between an archetype and its conditional variations, and a set of temporally constraint schema–events (maximum schema–event temporal interval).

A found case is recorded as a pair of a measure–range with a Galant archetype name. These found cases are then compared with annotations, and the values of true– and false–positive cases are utilised to calculate the value of recall.

4.3 Computational experiments

4.3.1 Parameters of the experiments

The computational experiments tested seventeen configurations for the schematic analysis operations, when searching for seven Galant family–types (including variants).

4.3.1.1 Schematic analysis configurations

Table 4.1 (below) presents the `SA` configuration profiles that were examined. These aim to isolate the effects of most of the parameters on the overall performance of the `SA` operation. Beginning from the top, the first seven `SA` configurations (`HXXX`) extract schema–events by considering only the harmonic information. The `Hnnn` (`SA` configuration) is the most generic setting considered, utilising only the extracted harmonic information. The next three configurations (`HnnS`, `HnnT`, and `HnnE`), examine the effects of different temporal regularity settings, gradually becoming more elaborate. The following two configurations (`HFFT` and `HAAT`) examine the effects of fixed and adaptive weights on harmonic segments, both under *total* temporal regularity thresholds. The final test of harmonic schema–events performs with completely adaptive behaviour in 'significance' weights and the most possible elaborate temporal regularity filters.

Table 4.1: Examined configurations for the schematic analysis operation.

	Significance		
	Positional	Transitional	Regularity
Hnnn	none	none	none
HnnS	none	none	strict
HnnT	none	none	total
HnnE	none	fixed	extended
HFFT	fixed	fixed	total
HAAT	adaptive	adaptive	total
HAAE	adaptive	adaptive	extended
SFFn	fixed	fixed	none
SFAn	fixed	adaptive	none
SAFn	adaptive	fixed	none
SAAn	adaptive	adaptive	none
SFFS	fixed	fixed	strict
SFFT	fixed	fixed	total
SFFE	fixed	fixed	extended
SAAS	adaptive	adaptive	strict
SAAT	adaptive	adaptive	total
SAAE	adaptive	adaptive	extended

The next ten `SA` configurations are extracting complete schema–events, i.e., including both harmonic and voicing information. The first four configurations (`SFFn`, `SFAn`, `SAFn`, and `SAAn`) examined the effects of fixed and adaptive weights, when no temporal regularity filters are considered. The following three configurations (`SFFS`, `SFFT`, and `SFFE`) examined the effect of temporal regularity, applying gradually more elaborate versions. Then, the last three `SA` configurations (`SAAS`, `SAAT`, and `SAAE`) examined the `SA` operation in adaptive 'significance' weights, also gradually elaborating the threshold of temporal regularity.

4.3.1.2 Examined Galant archetypes

From the more than 20 documented Galant musical schemata prototypes that are found in the book *Music in the Galant Style* (Gjerdingen, 2007), the experiments presented here focused on the following seven Galant family–types:

- The *Romanesca*, an opening schema (*see* Appendix A.1);
- The *Do–Re–Mi*, an opening schema (*see* Appendix A.4);
- The *Meyer*, an opening and thematic schema (*see* Appendix A.5);
- The *Fenaroli*, an intermediate schema (*see* Appendix A.6);
- The *Prinner*, a parallel passing of thirds from the fifth to the tonal, a pre–cadential schema (*see* Appendix A.7);
- The *Quiescenza*, a prolongation, pre–cadential schema (*see* Appendix A.6), and
- The *Clausula*, the ending schema of a phrase/part (*see* Appendix A.9).

The main reason for selecting to search the aforementioned musical schemata

Table 4.2: Annotations from the *Galant schemata* dataset.

Family–type	Annotations
Romanesca	58
Galant	32
Step–wise bass	14
Leaping–bass	11
Meyer	22
Generic	10
Variant–bass (3271)	6
Variant–bass (1251)	6
Do-Re-Mi	48
Generic	38
Do–Re...Re-Mi	10
Fenaroli	68
Prinner	105
Quiescenza	63
Clausula	141
Complete	31
Cudworth	13
Deceptive	9
Converging	17
Half	22
Comma	25

family–types concerned the availability of annotations. While most of the selected musical schemata types have more than 50 annotations in Gjerdingen's book, an additional dataset was created, annotating piano sonatas with the *Meyer*, *Prinner*, and *Clausula* instances (*see* Appendix B.2). The archetypes and variants searched for and their respective quantities are shown in Table 4.2. From the *Sonatas schemata* dataset (in Section B.2) the three musical schemata classes considered were: the *Meyer* (158 instances); the *Prinner* (92 instances), and the *Clausula* (260 instances).

Details about the Galant musical schemata archetypes can be found in

Appendix A. Information about the examined datasets can be found in Appendix B.

4.3.1.3 Performance metric

As stated earlier in this Chapter, the performance of the `SA` operation was measured by the metric of recall. This is a binary comparison of the measure–ranges between annotations and identified Galant archetype instances, counting true–positives (*TP*, the correct findings) and false–negatives (*FN*, the incorrect omissions). Utilising the values of *TP* and *FN*, the metric of recall is calculated with the following equation:

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

where
TP are the True Positives,
FN are the False Negatives.

4.3.2 Results

In Figure 4.1, the search results for the seven Galant family–types in the seventeen SA configurations are shown in recall.

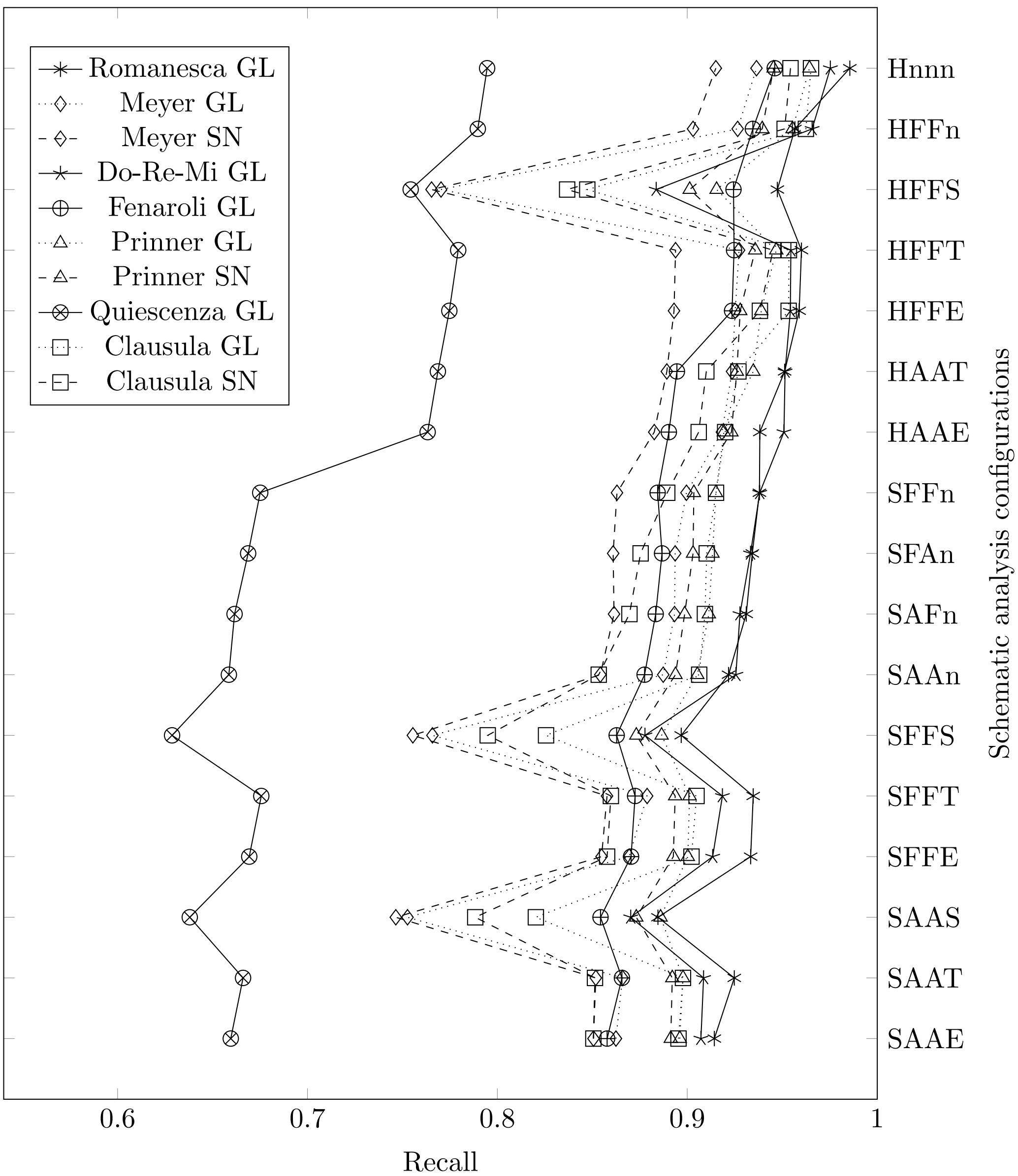


Figure 4.1: Search results. SN indicates findings from the *Sonatas* schemata dataset, and GL from the *Galant* schemata dataset.

The numerical values of the `SE` computational experiments are shown in Table E.1.

4.3.2.1 Results description

Before analysing the search results, recall that the goal of the search task was to evaluate the performance of the various `SA` configurations. Ideally, recall would be perfect in all configurations. That would mean that even the most sophisticated `SA` configuration (`SAAE`) would be capable of creating the sequence of schema–events without excluding any 'schematic' information.

The results of this experiment are analysed considering:

- The structural integration;
- The `SA` configuration;
- The characteristics of each archetype, and
- The dataset examined.

Considering structural integration, the first seven configurations that tested harmonic schema–events (the `HXXX` configurations), all performed better in comparison to those utilising complete schema–events (the `SXXX` configurations). This was expected as the omission of voicing information from schema–events enables generality.

Comparing the results among different datasets (for the three common archetypes of *Meyer*, *Prinner* and the *Clausula*), the *Galant schemata* dataset had higher recall than the *Sonatas schemata*, regardless of schematic integration or threshold configuration. This was also expected since keyboard sonatas are more elaborated than are works in the Galant period, exhibiting additional passing and ornamental notes and chords, making it more difficult for the `SA` algorithm to decide on which schema–events to keep.

Considering the performance in finding the various Galant archetypes, this was mainly subject to the temporal arrangements of each schema type.

The generally lowered results of the *Quiesenza* were due to the frequent inability of the *HarmAn* algorithm to detect specific harmonies. The mainly isochronous schemata (the *Romanesca*, the *Prinner*, the *Fenaroli*, and the *Quiesenza*) all performed well regardless of the temporal regularity filter applied. On the other hand, archetypes with freer temporal arrangements (such as the *Do-Re-Mi* and the *Cadence*) or with schema–event pair–arrangements (the *Meyer*) were subject to the thresholds of temporal regularity and had diminished performance, especially when the *strict* thresholds were utilised (`XXXs`). The transition from *total* thresholds to *extended* also had minimal effect, increasing the performance of only the *Meyer* schema.

Starting from the more generalised `SA` configurations and moving towards more elaborate thresholds and weights (*see* Figure 4.1 from top to bottom), as it was expected, recall drops. The first configuration (`Hnnn`) is matching only harmonic information, and is basically testing the harmonic extraction algorithm. The next three configurations (`HnnS`, `HnnT`, and `HnnE`), examined the effects of temporal regularity filters, gradually becoming more elaborate. The *strict* filter was the most problematic for reasons related with the temporal arrangements of each schema. The *total* and *extended* configurations of the temporal regularity threshold absorbed any exclusions from the *strict* configuration, resulting in perfect recalls. In the following two configurations (`HFFT` and `HAAT`), neither setting had a negative effect, maintaining recall mostly perfect for all schema types.

The next ten `SA` configurations (`SXXX`) examined the application of significance and regularity on complete schematic integration of schema–events in most possible cases. The performance of the first four configurations utilising complete schema–events (`SFFn`, `SFAn`, `SAFn`, and `SAAn`) indicates that both positional and transitional significance, under any configuration (*fixed* or *adaptive*), do not lower the recall value. This means that these filters threshold the generated schema–event samples, while maintaining perfect recall. Applying the three levels of temporal regularity thresholds in schematic structures (`SAAS`, `SAAT`, and `SAAE`), these had a similar effect as to when harmonic segments were utilised, i.e., being affected by the temporal arrangement of each schema type.

In total, the value of average recall for the most elaborate configuration of the schematic analysis operation (`SAAE`) was 0.866. The schematic configuration (`SXXX`) with the highest average recall on all Galant archetypes was `SFFn`, 0.884.

4.4 Findings

The schematic analysis operation creates the Galant information environment of the examined model. Due to the importance of this operation, its internal mechanisms had to be tested in isolation, to decide upon which of its parameters may have had a disadvantageous effect. Performing the search task for Galant archetypes under different schematic analysis configurations, such effects were revealed.

Therefore, the use of schema–events over harmonic information is a relatively safe option, as the highest total recall value of such configuration (`SFFn`) was 0.884. The use of positional significance parameters did not lower the performance of the `SA` in any configuration tested. The use of transitional significance caused minor decreases in recall when examining the keyboard works with *adaptive* configurations; however, in general, it is also considered safe to use. Temporal regularity is utilised as a *total* threshold, and additional temporal profiles were added.

As stated in the introduction to this Chapter, the goal of this search task was to determine whether the `SA` operation and its internal mechanisms are performing adequately for the following operation. As shown from results, even the most elaborate configuration (`SAAE`) suffices for the task.

4.5 Summary

This Chapter has presented the task of searching for Galant archetypes in sequences of schema–events that are extracted through the operation of schematic analysis. The main goal of the task was to examine the performance of the `SA` operation and its methods for the extraction of schema–event sequences. A series of configurations for the `SA` were examined, testing its parameterisation. Results indicate a robust mechanism, capable of reducing the score into 'schematic' content.

5 Classification of Galant musical schemata

The task of Galant musical schemata classification examined the example–based learning operation of the proposed model (`XL`, *see* Section 3.4), through a recognition process that identifies 'trained' family–classes in schematic analysis form (i.e., the outcome of the `SA` operation, *see* Section 3.3). The first section describes the classification task, outlining the processing stages within the context of the examined cognitive model, and its relation with the search task, examined in the previous chapter (*see* Section 4). The next section overviews the recognition process that classifies segments in `SA` with 'trained' family–classes, describing its approximation method. Then, the four computational experiments that tested different classification configurations are presented, followed by results and analysis. The final chapter discusses findings regarding the example–based leaning method of the model and its recognition process.

5.1 Task description

Within the architecture of the examined cognitive model (*see* Chapter 3), the (Galant archetype) classification task involves similar processes to the search task (*see* Section 4), and extends it in the following regards:

- The family–class definitions are given through annotations, performing the `XL` operation;
- The similarity models for musical schemata structures allow approximations, and
- The ‘unknown’ classes may now be identified.

In the classification task, after the completion of the `SA` operation for a time–window, if annotations are present, the model performs the `XL` operation (*see* Section 3.4) and integrates the information of the training example into the annotated family–class exemplar–base in the `GMSC` (*see* Section 3.4.2). When no annotations are given to the model, the recognition process identifies the ‘trained’ classes in the `SA` form. In this recognition process, all exemplars in the `GMSC` are projected through their archetype(s) and class–similarity functions onto the schematic analysis form of the time–window so that matching may be performed. The recognition process extends the ‘exact’ matching of the identification process in the search task (*see* Section 4.2.2) to include approximation similarity models. Therefore, all schema–events in the schematic analysis form are also tagged and temporally allocated thresholds; this time, however, doing so considers approximations for pair comparisons of single schema-events, and complete schema–event progression thresholds.

The results of the recognition process are in the form of continuous `SA` segments, labelled with a ‘trained’ family–class from the `GMSC`. The main issue on the recognition results concerns the overlapping of found segments. Overlapping may occur among instances of different family–types and also among multiple instances of the same family–class, given the repetitions of same schema–events. These issues are handled by the recognition process with consideration of temporal forms, for the case of multiple instances of the same family–type, as well as a mechanism that first attempts to avoid overlapping by selecting other schema–events. If that is not possible, it selects the greater (in duration) instance. After the recognition process, those segments in the SA that remained unlabelled are categorised as 'unknown' classes.

Similarly to the search task, the found family–classes are recorded as pairs of a measure–range with a Galant archetype family–class name. Since in the classification task the recognition process is examined, the found cases are compared with annotations to measure the values of true– and false–positive cases. This performed so as to calculate the value of recall, as well as the false–positives (i.e., the incorrect identifications), thus to calculate the value of precision. The recall metric reflects on how well the class similarity function integrates the information from the training examples. The precision metric reflects on the accuracy of the SA process in extracting the schematic elements, and how well the recognition process performs in excluding false identifications.

5.2 Computational implementations by the AES system

The AES system implements the classification task similarly to the search task, embedding the XL operation whenever annotations are found, and by enabling approximations in the recognition process. The XL operation and the transformation of the class–similarity function are described in Section 3.4.

This operation is implemented by the `GMSC` class (*see* Section 3.2.1), where the class–similarity function of a family-class integrates a training–example. The recognition process, and its approximation methods, are now described.

5.2.1 The recognition process

The recognition process of a musical schema family–class identifies the archetype(s) of the classes stored in the `GMSC` in schematic analysis form. This function inputs the schematic analysis of a time–window information–state from the active–plane (`SA.TW.SS`) and the family–classes (i.e., the archetype and the class–similarity function of each). The recognition processing is applied in three stages.

First, all of the kinds of schema–events that are stored in the exemplar–base of each family–class are identified in the examined schematic analysis form. These pairs of schema–events may match exactly or approximately (*see* below), when selected properties of the schema–events are considered. If such a schema–event–pair is considered a match (i.e., identified), the schema–event in the schematic analysis is tagged with the family–class label and (schema–event) order (i.e., the position within the archetype's sequence of schema–events).

After the tagging of the schema–events in the schematic analysis representation, these are bound temporally by a maximal (temporal) interval between successive schema–events. All combinations of schema–event sequences that may be formed within these bounds are considered valid recognitions. Through this process, all of the classes in the `GMSC` may be identified simultaneously within the data scope of a time–window.

The last stage of the recognition process involves the selection from the (possibly) multiple and overlapping instances that are recognised as being from the same family–class, and the overlapping of different family–class instances. When an instance of a family–class has repetitions of one of its schema–events, this will cause the recognition process to identify multiple overlapping schema–event sequences. To identify which of these sequences may be the most prominent, the extended forms of temporal regularity are applied (*see* Section 3.3). These impose a stricter filtering (than the 'total' regularity threshold) on the temporal relations among the schema–events that compose a found instance. In the event where any of the multiple instances of a family–class that may be found are overlapping with instances of different family–types, these are discarded, and the non–overlapping instance is preferred.

The recognition process is the same as in the search task when exact matching is considered; all content–related properties of the comparing schema–events must be the same. In contrast, when approximate matching is considered, two thresholds may be applied, addressing the differences between same–order schema–event relations and the sum of these differences on complete sequence of schema events. The approximation of the similarity between schema–events is implemented as a generalisation, considering only harmonic class. Thus, all of the inversions of a harmonic class are considered equal, while the voicing information is discarded. The approximation of complete musical schemata instances is considered a threshold on the total number of schema–event approximations. Therefore, when approximate recognition of a family–type is applied, this is expressed as the percentage of the total number of schema–events of a family–class that may have only the same harmonic class. Any other type of approximation, such as omission or harmonic substitutions, was considered to be too general for it to maintain the characteristics of a family–class.

With the utilisation of the approximation method above, the aim is to balance between performance metrics. The greater the approximation, the more likely recall will increase, and precision will fall. Accordingly, the lower the approximation, the more likely recall will decrease, and precision will increase.

5.3 Computational experiments

In total, sixteen classification configurations were tested, organised into four groups each with four cases, for three Galant family–classes (*Meyer*, *Prinner*, and *Clausula*). The performance metrics of recall and precision were utilised, and the results are shown in *F1* score diagrams.

5.3.1 Parameters of the experiments

The parameters of the computational experiments that examined the `XL` operation and the recognition process, concerned the configuration of the classification, the examined Galant family–types, and the performance metrics.

5.3.1.1 Examined Galant archetypes

For the classification task, the three Galant family–types considered were:

Table 5.1: Galant archetypes participating in the classification task and their annotations.

	Galant	Sonatas
Meyer	35	158
Prinner	145	92
Clausula	48	260

- The *Meyer*, a structurally stable schema with voice variants, represented by three variants (*see* Appendix A.5);
- The *Prinner*, a stable schema in all regards, represented by one main archetype (*see* Appendix A.7), and
- The *Clausula*, a highly variant schema, represented by seven variants (*see* Appendix A.9).

Table 5.1 displays the number of annotations for the above three Galant archetypes schemata in the two examined datasets, the *Galant schemata* dataset (*see* Appendix B.1), and the *Sonatas schemata* dataset (*see* Appendix B.2).

5.3.1.2 Classification configurations

The parameters that form the configuration of each classification experiment concerned the following:

- The `SA` configuration;
- The training example form;
- The handling of annotations, the training/testing configuration, and
- The recognition similarity.

`SA` configurations . Since the `SA` form is the information environment of the examined model, two `SA` configurations were tested. The `SA` configuration of SFFT utilises fixed weights of significance and a *total* regularity threshold. The SA configuration of SAAE utilises adaptive weights of significance and an *extended* regularity threshold. These two SA configurations are tested to examine whether the specificity that is added to the SA operation with the SAAE configuration through adaptive weights of significance and extended temporal regularity filters affects the precision performance of the model.

Form of training examples . The second parameter concerns the form of the training examples, meaning these may be represented as segments of schematic analysis form, or, exemplars, comprising of a single sequence of schema–events. As is described in Section 3.3, the final form of schematic analysis may include more than a single sequence of schema–events. This adds additional processing to the class–similarity functions, and may also lead to erroneous results. Utilising sequences of single schema–events, i.e., exemplars, the performance of the XL operation and the recognition function may be examined, resulting in the avoidance of any issues that may be caused from training with schematic analysis segments. The comparative evaluation of this configuration with those using schematic analysis segments for training will also indicate any faults the latter may cause when class-similarity is applied.

Managing annotations in training and testing groups . A common problem in classification tasks is the usually limited number of expert annotations, i.e., training examples. When artificial neural networks are utilised for classification, the lack of annotations raises issues such as ‘overfitting’ and ‘bias’, meaning that only trained information may be identified. In the case of musical schemata identification, and in the context of the examined cognitive model, the above issues, although they may exist, seem irrelevant. Since both the training and the testing of the Galant archetypes utilise music information in the schematic analysis form, the specificities of a Galant archetype instantiation (e.g., embellishments) are excluded. Besides that, a Galant musical schema is a highly structured unit of information, and even a single (highly–structured) training example in `SA` may serve as a family–class representative.

However, the use of methods that split the training and testing examples was considered a parameter that could test the performance of the class–similarity function of a family–class. Therefore, the following separation of training and testing examples was considered:

- Intra–dataset;
- Inter–dataset, and
- Cross–dataset.

In the intra–dataset configuration, both the training examples and the testing examples are from the same dataset. This is the simplest form through which to examine training and testing with `SA` segments. The method of N–fold cross–validation (Friedman et al., 2001) was applied in order to overcome the limited amount of annotations. Applying N–fold cross–validation, each collection of annotations for the selected musical schemata classes was split into N non–overlapping sets. Next, the model is configured with one of these sets, leaving the other N–1 sets for testing. The results from N run–times for each schema–type are then averaged. Regarding N–fold cross–validation, the ideal number of sets (N) would be K, where K is the number of examples in the dataset. However, this approach (also called 'leave–one–out') is computationally demanding. In this study, to balance performance and model–bias of the classifiers, N was arbitrarily selected to be one sixth of the size of annotations for each family–class.

The Inter–dataset configuration is when training examples from both datasets are considered. In this case, although annotations number more than in the Intra–dataset configuration, the method of cross–validation is again applied.

In the Cross–dataset configuration, the classifier is utilising training examples from one dataset (e.g., the *Galant* schemata dataset) then to perform classification on the other dataset (e.g., the *Galant* schemata dataset), and *vice–versa*. Utilising different sources for training and testing, the schematic analysis of these may differ. Thus, if the classification results are the same when training examples from different datasets are utilised, this finding would indicate robustness in the class–similarity function.

Recognition approximation . The parameterisation of the recognition process concerns the two modes of similarity, for exact and approximate matching between the family–classes and the musical schemata instances in `SA` form (*see* Section 5.2.1). This differentiation aims to examine whether the selected approximation method may facilitate the recognition of the variants of a family–class.

Summarising, the configurations of the classification computational experiments include the following parameters and values:

- The configuration of the schematic analysis operation (`SA`):
  - `SFFT`, and

  - `SAAE`.
- Training example format:
  - Exemplars, and
  - Segments in `SA` form.
- Training and testing data sources:
  - Inter–dataset;
  - Intra–dataset, and
  - Cross–dataset.
- Recognition similarity model:
  - Exact, and
  - Approximate.

The above parameters and values may be expressed by the following set:

```
(SA-config(SFFT,SAAE),
 TrainingFormat(Exemplar/SA-config),
 TrainTestSeparation(Inter, Intra, Cross),
 Matching(Exact, Approximate))
```

These parameters were organised and codified in four groups as follows:

A Baseline Exemplar training:

```
SA-config(1,2), Exemplar, Inter, Similarity(Exact, Approximate)
```

B Schematic Intra–dataset training:

```
SA-config(1,2), SA-config(1,2), Intra(GS,SS), Similarity(Exact, Approximate)
```

C Schematic Inter–dataset training:

```
SA-config(1,2), SA-config(1,2), Inter, Similarity(Exact, Approximate)
```

D Schematic Cross–dataset training:

```
SA-config(1,2), SA-config(1,2), Cross, Similarity(Exact, Approximate)
```

The above parameters result in 24 configurations. Each row below presents the abbreviation code for the configuration (e.g., S22IA), and the parameter values of the configuration. Analytically, the configurations are:

```
A. SA-config(1, 2), Exemplar, Inter, Similarity(Exact, Approximate)
 S1EIE: SA1, Exemplar, Inter, Exact
 S1EIA: SA1, Exemplar, Inter, Approximate
 S2EIE: SA2, Exemplar, Inter, Exact
 S2EIA: SA2, Exemplar, Inter, Approximate
B. SA-config(1,2), SA-config(1,2), Intra(GS,SS), Similarity(Exact, Approximate)
 S11WE: SA1, SA1, Intra(GL/SS), Exact
 S11WA: SA1, SA1, Intra(GL/SS), Approximate
 S22WE: SA2, SA2, Intra(GL/SS), Exact
 S22WA: SA2, SA2, Intra(GL/SS), Approximate
C. SA-config(1,2), SA-config(1,2), Inter, Similarity(Exact, Approximate)
 S11IE: SA1, SA1, Inter, Exact
 S11IA: SA1, SA1, Inter, Approximate
 S22IE: SA2, SA2, Inter, Exact
 S22IA: SA2, SA2, Inter, Approximate
D. SA-config(1,2), SA-config(1,2), Cross, Similarity(Exact, Approximate)
 S11CE: SA1, SA1, Cross, Exact
 S11CA: SA1, SA1, Cross, Approximate
 S22CE: SA2, SA2, Cross, Exact
 S22CA: SA2, SA2, Cross, Approximate
```

The first group of configurations (A) offers a Baseline configuration for both the XL operation and the recognition process, as exemplars are utilised for training. This configuration examines how the class–similarity function operates with 'clean' training examples, and whether the recognition process may approximate these to find instances. Utilising exemplars to train the model, this configuration extends the search task. Comparing the precision between the two `SA` configurations may indicate which `SA`–configuration is more selective in the extraction of schema–events. The second group of configurations (`B`) is the first 'true' example–based learning method, as the training examples are now given in schematic analysis form. Such a configuration uses the simplest training scenario, that of performing intra–dataset classification. The third group of configurations (`C`) utilises training examples from both datasets, also to classify them in both datasets. The last group (`D`) utilises annotations from different datasets.

5.3.1.3 Performance metrics

The performance of the classification task was measured by the metrics of *recall* and *precision*. Similarly to the search results, the task performs binary comparisons between the classification results and annotations, to calculate the values of true–positives (*TP*, the correct findings), false–positives (*FP*, the incorrect findings) and false–negatives (*FN*, the incorrect omissions). Utilising the values of *TP*, *FP* and *FN*, the metrics of *recall*, *precision*, and *F1* score were calculated with the following equations:

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \qquad \text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \qquad \text{F1} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

where

TP are the True Positives,

FP are the False Positives, and

FN are the False Negatives.

Utilising the metrics of *recall* and *precision*, the results of the classification experiments are displayed with a single *F1* score diagram for all the parameter cases of a configuration family (i.e., `A`, `B`, `C`, and `D`). All results are discussed at the end of this section.

5.3.2 A. Training with exemplars

5.3.2.1 Examined configurations

The first configuration examined the case where the model is trained with exemplars. The parameters tested include:

```
( SA-config(1, 2), Exemplar, Inter, Similarity(Exact, Approximate) )
```

Analytically, these are:

A1 `S1EIE`: (SA1, Exemplar, Inter, Exact);

A2 `S1EIA`: (SA1, Exemplar, Inter, Approximate);

A3 `S2EIE`: (SA2, Exemplar, Inter, Exact);

A4 `S2EIA`: (SA2, Exemplar, Inter, Approximate).

5.3.2.2 Results

Total illustration of recall and precision values for group `A` classification configurations .

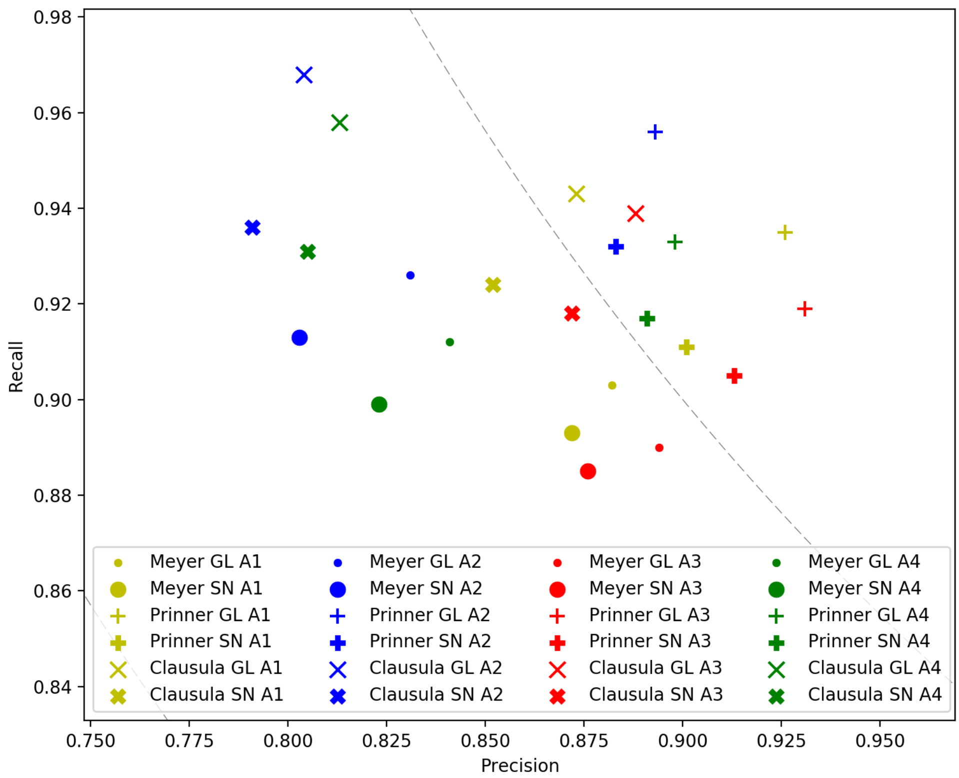


Figure 5.1: Recall and precision for classification experiment A. F1 scores.

In terms of `F1` scores, the best value was achieved by the `S1EIE` configuration (*see* Figure 5.1, thin yellow cross, 'A1 SFFT Exemplar Prinner GL', 0.930). The lowest `F1` score was yielded by the `S1EIA` configuration (*see* Figure 5.1, thick blue circle, 'A2 SFFT Exemplar Approximate Meyer SN', 0.854). The average `F1` score for this configuration group (`A`) was 0.892.

In terms of *precision*, the best value was yielded by the `S2EIE` configuration (*see* Figure 5.1, thin red cross, 'A3 SAAE Exemplar Exact Prinner GL', 0.931). The lowest *precision* value was returned by the `S1EIA` configuration (*see* Figure 5.1, thick blue x–mark, 'A2 SFFT Exemplar Approximate

Clausula SN', 0.791). The average *precision* value for this configuration group (`A`) was 0.864.

In terms of *recall*, the best value was returned by the `S1EIA` configuration (*see* Figure 5.1, thin blue x–mark, 'A2 SFFT Exemplar Approximate Clausula GL', 0.968). The lowest *recall* value was given by the `S2EIE` configuration (*see* Figure 5.1, thick red circle, 'A3 SAAE Exemplar Exact Meyer SN', 0.885). The average *recall* value for this configuration group (`A`) was 0.922.

Comparing the performance among the three Galant family–types, in each of the four configurations (defined by color): the *Prinner* (*see* Figure 5.1, crosses) had the highest `F1` score and *precision*; the *Clausula* (*see* Figure 5.1, x–marks) had the second best `F1` scores, but the best *recall* values, and, the *Meyer* (*see* Figure 5.1, circles) had the lowest *recall* values, but *precision* greater than that of the *Clausula*.

Comparing the performance between different datasets (*see* Figure 5.1 thin marks for the *Galant schemata* dataset and thick marks for the *Sonatas schemata*), in all of the configurations, and for all family–types, the *Galant schemata* dataset had both, greater recall and precision.

Comparing the performance between similarity models, the *exact* matching (*see* Figure 5.1, yellow and red colors) had generally greater precision values for all of the family–classes, than those of the *approximate* matching (*see* Figure 5.1, blue and green colors). When approximation in similarity was applied, in comparison to their corresponding family–class with 'exact' matching, recall increased and precision dropped.

Considering the two `SA` configurations, the *SFFT* configuration (*see* Figure 5.1, yellow and blue colors), had both greater recall and lower precision, than their corresponding configurations using the *SAAE* configuration (*see* Figure 5.1, red and green colors).

5.3.3 B. Training with schematic analysis, intra–dataset classification

5.3.3.1 Examined configurations

The second configuration examined the case where the model is trained with SA–segments, performing Intra–dataset classification. The parameters tested include:

```
( SA-config(1,2), SA-config(1,2), Intra(GS,SS), Similarity(Exact, Approximate) )
```

Analytically, these are:

B1 `S11WE`: (SA1, SA1, Intra(GL/SS), Exact);

B2 `S11WA`: (SA1, SA1, Intra(GL/SS), Approximate);

B3 `S22WE`: (SA2, SA2, Intra(GL/SS), Exact);

B4 `S22WA`: (SA2, SA2, Intra(GL/SS), Approximate).

5.3.3.2 Results

Total illustration of recall and precision values for group `B` classification configurations .

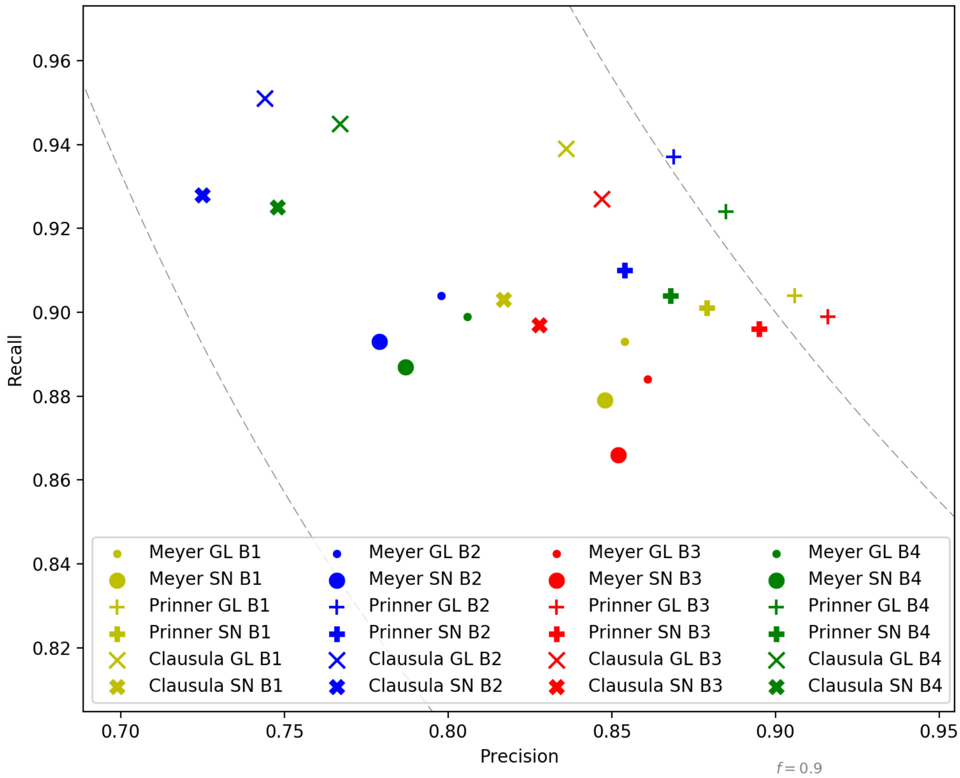


Figure 5.2: Recall and precision for classification experiment B. F1 scores.

In terms of `F1` scores, the best value was achieved by the `S22WE` configuration (*see* Figure 5.2, thin red cross, 'B3 SAAE SAAE Exact Prinner GL', 0.907). The lowest `F1` score was yielded by the `S11WA` configuration (*see* Figure 5.2, thick blue x–mark, 'B2 SFFT SFFT Approximate Clausula SN', 0.814). The average `F1` score for this configuration group (`B`) was 0.867.

In terms of *precision*, the best value was yielded by the `S22WE` configuration (*see* Figure 5.2, thin red cross, 'B3 SAAE SAAE Exact Prinner GL', 0.916). The lowest *precision* value was returned by the `S11WA` configuration (*see* Figure 5.2, thick blue x–mark, 'B2 SFFT SFFT Approximate Clausula

SN', 0.725). The average *precision* value for this configuration group (`B`) was 0.832.

In terms of *recall*, the best value was returned by the `S11WA` configuration (*see* Figure 5.2, thin blue x–mark, 'B2 SFFT SFFT Approximate Clausula GL', 0.951). The lowest *recall* value was given by the `S22WE` configuration (*see* Figure 5.2, thick red circle, 'B3 SAAE SAAE Exact Meyer SN', 0.866). The average *recall* value for this configuration group (`B`) was 0.908.

Comparing the performance of the Galant family–types, the different datasets, the similarity models, and the `SA` configurations, these are the same as in computational experiment `A`, but with lower values.

5.3.4 C. Training with schematic analysis, inter–dataset classification

5.3.4.1 Examined configurations

The third configuration examined the case where the model is trained with `SA`–segments, performing Intra– and Inter–dataset classification. The parameters tested include:

```
( SA-config(1,2), SA-config(1,2), Inter, Similarity(Exact, Approximate) )
```

Analytically, these are:

C1 `S11IE`: (SA1, SA1, Inter, Exact);

C2 `S11IA`: (SA1, SA1, Inter, Approximate);

C3 `S22IE`: (SA2, SA2, Inter, Exact);

C4 `S22IA`: (SA2, SA2, Inter, Approximate).

5.3.4.2 Results

Total illustration of recall and precision values for group C classification configurations .

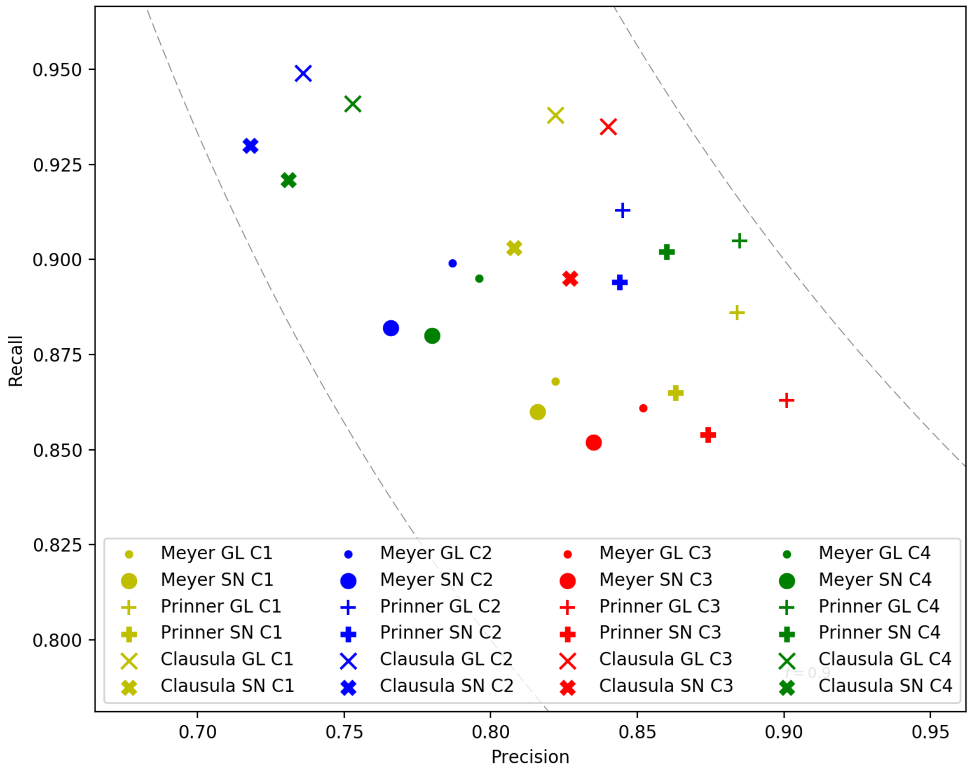


Figure 5.3: Recall and precision for classification experiment C. F1 scores.

In terms of `F1` scores, the best value was achieved by the `S22IA` configuration (*see* Figure 5.3, thin yellow cross, 'C4 SAAE SAAE Approximate Prinner GL', 0.894). The lowest `F1` score was yielded by the `S11IA` configuration (*see* Figure 5.3, thick blue x–mark, 'C2 SFFT SFFT Approximate Clausula SN', 0.810). The average `F1` score for this configuration group (`C`) was 0.853.

In terms of *precision*, the best value was yielded by the `S22IE` configuration (*see* Figure 5.3, thin red cross, 'C3 SAAE SAAE Exact Prinner GL', 0.901). The lowest *precision* value was returned by the `S11IA` configuration (*see* Figure 5.3, thick blue x–mark, 'C2 SFFT SFFT Approximate Clausula SN', 0.718). The average *precision* value for this configuration group (`C`) was 0.818.

In terms of *recall*, the best value was returned by the `S11IA` configuration (*see* Figure 5.3, thin blue x–mark, 'C2 SFFT SFFT Approximate Clausula GL', 0.949). The lowest *recall* value was given by the `S22IE` configuration (*see* Figure 5.3, thick red circle, 'C3 SAAE SAAE Exact Meyer SN', 0.852). The average *recall* value for this configuration group (`C`) was 0.895.

The performance differences among the Galant family–types, the different datasets, the similarity models, and the `SA` configurations are analogous to the previous computational experiments (`A` and `B`), but with lower values than those of computational experiment `B`.

5.3.5 D. Training with schematic analysis, cross–dataset classification

5.3.5.1 Examined configurations

The last configuration examined the case where the model is training with SA–segments, to perform Cross–dataset classification. The parameters tested include:

```
( SA-config(1,2), SA-config(1,2), Cross, Similarity(Exact, Approximate) )
```

Analytically, these are:

D1 `S11CE`: (SA1, SA1, Cross, Exact);

D2 `S11CA`: (SA1, SA1, Cross, Approximate);

D3 `S22CE`: (SA2, SA2, Cross, Exact);

D4 `S22CA`: (SA2, SA2, Cross, Approximate).

5.3.5.2 Results

Total illustration of recall and precision values for group `D` classification configurations .

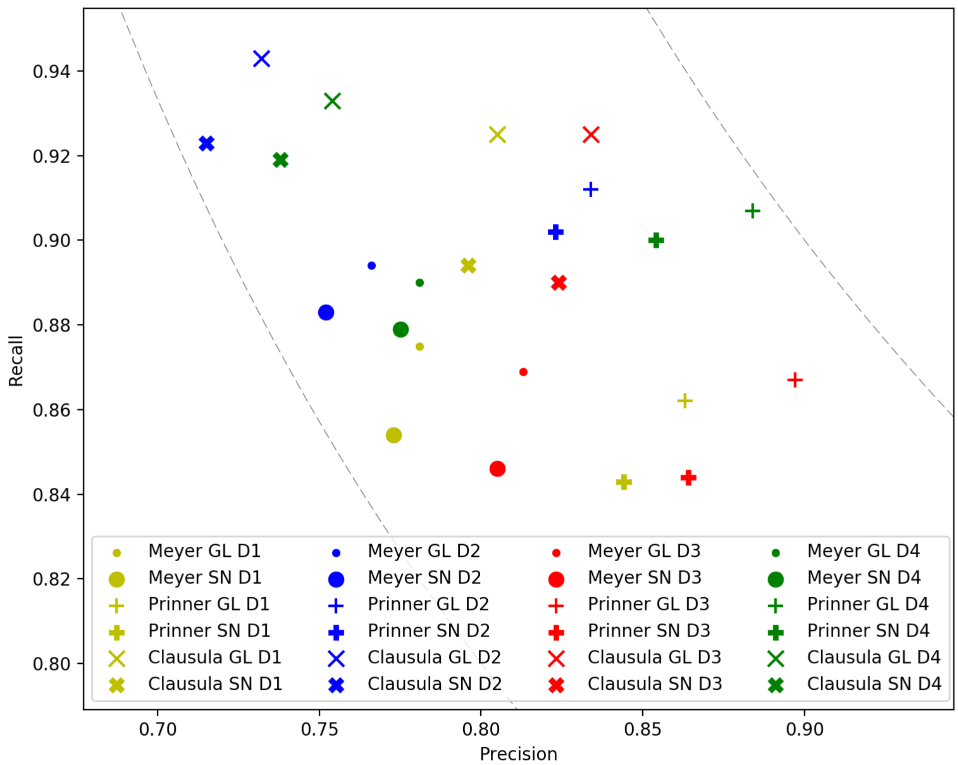


Figure 5.4: Recall and precision for classification experiment D. F1 scores.

In terms of `F1` scores, the best value was achieved by the `S22CA` configuration (*see* Figure 5.4, thin green cross, 'D4 SAAE SAAE Approximate Prinner GL', 0.895). The lowest `F1` score was yielded by the `S11CA` configuration (*see* Figure 5.4, thick blue x–mark, 'D2 SFFT SFFT Approximate Clausula SN', 0.805). The average `F1` score for this configuration group (`D`) was 0.844.

In terms of *precision*, the best value was yielded by the `S22CE` configuration (*see* Figure 5.4, thin red cross, 'D3 SAAE SAAE Exact Prinner GL', 0.897). The lowest *precision* value was returned by the `S11CA` configuration (*see* Figure 5.4, thick blue x–mark, 'D2 SFFT SFFT Approximate Clausula SN', 0.715). The average *precision* value for this configuration group (`D`) was 0.804.

In terms of *recall*, the best value was returned by the `S11CA` configuration (*see* Figure 5.4, thin blue x–mark, 'D2 SFFT SFFT Approximate Clausula GL', 0.943). The lowest *recall* value was given by the `S11CE` configuration (*see* Figure 5.4, thin yellow cross, 'D1 SFFT SFFT Prinner SN', 0.843). The average *recall* value for this configuration group (`D`) was 0.890.

Similarly to the previous computational experiments (i.e., `A`, `B` and `C`), the performance differences among the Galant family–types, the different datasets, the similarity models, and the `SA` configurations are analogous, including the lowering of values in comparison to the previous computational experiment (`C`).

5.3.6 Results description

Considering the results of the classification experiments above, general remarks can be drawn. Through out all of the configurations, similar relations among the results were observed. Comparing the performance of the Galant family–types, the *Prinner* (crosses) had the highest F1 scores and *precision*. The *Clausula* (x–marks) had the second best F1 scores, but the best *recall* values. The *Meyer* (circles) had the lowest *recall* values, but *precision* greater than that of the *Clausula*. Comparing the performance between different datasets, in all configurations, the *Galant schemata* dataset (thin stroke marks) had both, greater recall and precision than those in the *Sonatas schemata* dataset (thick stroke marks). Comparing the performance between similarity models, the *exact* matching (yellow and red colors) had generally greater precision values for all of the family–classes, than those of the *approximate* matching (blue and green colors), which had recall increased and precision dropped. Considering the two SA configurations, the SFFT configuration (yellow and blue colors) had both greater recall and lower precision, than their corresponding configurations using the SAAE configuration (red and green colors). Comparing the different training configurations, the first configuration (A, using exemplars) had the best performance. Comparing the SA–segment training groups (i.e., B, C and D), B (Intra–dataset) had the highest F1 score, followed by C (Inter–dataset), with the Cross–dataset training configuration (D) coming last in all cases. Comparing the N1 configurations (i.e., A1, B1, C1, and D1), the values of recall and precision gradually fall for all family–classes. This decrease in values is observed for all such sets of the XL computational experiments' configurations (i.e., A2–D2, A3–D3, and A4–D4).

5.4 Findings

The results of the classification computational experiments suggest a robust class–similarity function, capable of transforming `SA`–segments into family–class definitions. However, the decrease in performance when `SA`–segments are utilised (instead of exemplars) indicates that the class–similarity function of a Galant family–class considers all of the possible schema–event sequences contained in a training example. Since not all of these sequences are part of an archetype definition, though, this resulted in the decrease of precision.

Similarly, the decrease in recall suggests that either the class–similarity is failing to create the correct difference–vector from a training example, or that the `SA` configuration is not extracting all of the valid schema–events.

Regarding the approximation method and the discard of voicing information, the results indicate that for family–classes with four schema–events (such as the *Meyer* and the *Prinner*), the impact on precision is low. However, when the *Clausula* is considered, this kind of approximation significantly lowers precision, as the only three harmonies (or more, depending on the variant) that comprise this family–class appear frequently in both datasets.

5.5 Summary

This chapter has presented the task of Galant musical schemata classification. The task examined the example–based learning operation (`XL`) of the proposed model (*see* Section 3.4) through the recognition of 'trained' classes in schematic analysis. By considering a set of parameters for the classification task, the computational experiments tested the capacity of the model to extract and identify family–classes under different circumstances. In general, the model performed adequately, managing to extract classes from training examples, and successfully to recognise them in schematic analysis form.

6 Discovery of Galant musical schemata

This chapter presents the Galant musical schemata discovery task that tested the discovery operation of the examined model (SD, *see* Section 3.5). The first section describes the discovery task within the context of the examined model architecture, overviewing the processes and the workflow of the SD operation. The next section presents the computational experiments that tested the SD operation through the Adaptive Expert System (AES) implementation. The last section discusses findings evaluating the results of the computational experiments.

6.1 Task description

The discovery of Galant musical schemata is examined as the process of regulating the accumulation of *similar* exemplars into the classification of the Galant family–classes, the GMSC. The SD operation is performed in two stages, with each stage expressed by a high–level process:

1. The Intra–Score Exemplar Discovery process (SA.ISED), and
2. The Exemplar–Recognition–Routine (ERR).

In the first stage of the SD operation, the schematic analysis of a score (SA.SS) is segmented into exemplars through the Intra–Score Exemplar Discovery process (SA.ISED). The SA.ISED process inputs the SA form of a score and first, creates the discovery–space of exemplars: schema–event progressions sampled with constraints concerning their minimum and maximum length (in both, schema–events and temporal interval values), and a maximal eIVIE distance (*see* Figure 2.2). Next, if there exist discovered family–classes in the GMSC, these are identified in the discovery space of exemplars through the Exemplar–Recognition process (ERP). Then, the Discovery–Matching–Threshold (DMT) is applied for non–overlapping exemplars of the same (schema–event) length, aiming to identify possible repetitions. In the final step of the SA.ISED process, a scoring mechanism selects the segmentation with the most 'valuable' exemplars, according to a point system that favours the selection of exemplars that are recognised from previously discovered ones, and repetitions. If no recognitions or such repetitions are found, the complete SA form of the score is appended to the 'uncategorised' repository of the schematic analysis module (SA.UN). If exemplar repetitions are found and a segmentation is selected, these are added to the repository of classified exemplars of the schematic analysis module (SA.CL). Thus, the outcome of the SA.ISED process is a segmentation of a score into non–overlapping exemplars, and their classification into repetitive and 'uncategorised'.

In the second stage of the SD operation, the Exemplar–Recognition–Routine (ERR) integrates the outcome of the Intra–Score Exemplar Discovery process (SA.ISED) into the long–term musical schemata representations in the GMSC through comparing all of the exemplars among the four repositories (i.e., the GMSC.SF, the GMSC.UN, the SA.CL, and the SA.UN). Whenever there is a match, according to the Exemplar–Discovery–Process (EDP), the discovery Integration–Routine (`GMSC.IR.D`) is activated, and the process of equilibrium (`GMSC.EQ`) determines which learning mode will integrate the newly found family–class (the exemplar–pair) into the `GMSC`. After the completion of the `ERR` process and the integration of any *similar* exemplars from the `SA` into the `GMSC`, the repository of 'uncategorised' exemplars of the `SA.ISED` process (`SA.UN`) is merged with its corresponding repository of 'uncategorised' exemplars in the `GMSC` (`GMSC.UN`), and a new score is considered.

6.2 Computational Experiments

6.2.1 Parameters of the experiments

The parameters of the computational experiments that may test the `SD` operation concern the following:

- The configuration of the schematic–analysis operation (`AES.SA`);
- The value of the Discovery–Matching–Threshold (`DMT`);
- The configuration of the equilibrium process (`GMSC.EQ`), and
- The type of input data (excerpts or full parts).

The complete set of `SD` configurations that stemmed from the above parameterisation is shown in Table 6.2. The computational experiments examined a subset of selected configurations from the aforementioned set of configurations.

6.2.1.1 Schematic–analysis configuration

To examine whether the quality of the extracted schematic analysis form affects the discovery operation, two `SA` configurations were tested: the `SFFT`, where schema–events are extracted with *fixed* 'positional' and 'transitional significance', applying a *total* regularity threshold, and, the `SAAE`, where schema–events are extracted with *adaptive* 'positional' and 'transitional significance', and an *extended* regularity threshold. The selected `SA` configurations were also utilised in the classification computational experiments (*see* Chapter 5).

6.2.1.2 The Discovery–Matching–Threshold

The Discovery–Matching–Threshold (`DMT`) is the maximum permissible difference between two exemplars, for the Exemplar–Discovery–Process (`EDP`) to considered them of the same family–class. The `DMT` is expressed by a conditional statement for content–related differences between two exemplars. The two examined `DMT` values concern the *exact* and the *approximate* matching among all of the pitch–related properties of exemplars with equal schema–event length. The *exact* `DMT` value means that the Exemplar–Difference–Vector (`EDV`) of two exemplars, resulting from an `EDP` process during the `ERR`, must be zeroed. Since the `DMT` value allows the creation of a family–class, selecting an *approximate* threshold should consider minimal differentiations between exemplars. The examined *approximate* `DMT` value was selected to be the exact matching of harmonic–classes, requiring only the half of the voicing information to also be equal. This means that exemplars with schema–events of the same harmonic class (also in the same position) will be considered as equal, only if at least half of the voicing information of the complete exemplars is also equal.

6.2.1.3 The process of equilibrium

When a new family–class is discovered by the Exemplar–Discovery process (`EDP`) during the second stage of the `SD` operation (`ERR`), the discovery Integration–Routine (`GMSC.IR.D`) is activated, calling the `GMSC.EQ` process to determine which learning mode should be performed. The parameterisation of the `GMSC.EQ` process concerns the assignment of values for the two thresholds of *diversity* and *variability*. These two thresholds may be configured with values that are static/fixed throughout a discovery session, or with values that adapt to the changes of certain qualities in the highly dynamic `GMSC` and its discovered family–classes.

To examine whether the `GMSC.EQ` performs as it should, this study examined three pairs of static threshold values (*see* Table 6.1). The selected diversity threshold values (`EDV`s) express the minimum differentiation among discovered family–classes as a percentage of the total binary comparisons of the harmonic–classes of schema–events in the same–position. In other words, the percentage of the Hamming distance between two exemplars performing binary harmonic–class comparison. Similarly, the selected variability threshold values (`EDV`s) express the maximum differentiation among the exemplars of a discovered family–class as a percentage of the total binary comparisons of the voicing information in schema–events of the same–position, i.e., the average of two Hamming distances, one for each voice. Recall that the `GMSC.EQ` process prohibits the creation of family–classes that are below the diversity threshold value. Therefore, a low value on the diversity threshold will favour the creation of new family–classes, while a high value will have the opposite effect. Accordingly, a high value on the variability threshold favours

Table 6.1: The configurations of the `GMSC.EQ` thresholds

	Diversity (minimum harmonic–class differentiation)	Variability (maximum voicing differentiation)
Loose	Single	Any
Strict	All	Half
Balanced	Half	Half

the expansion of existing discovered family–classes, while a low value hinters such action. The three configurations of the `GMSC.EQ` threshold values (Table 6.1) aimed to examine three different 'behaviours' of the (discovered family–class) integration regulatory mechanisms. The *Loose* configuration examined thresholds that enable both, the creation of new family–classes through a minimal diversity value (a single harmonic class differentiation), and the expansion of discovered family–classes with exemplars of any voicing differentiation. The *Strict* configuration examined thresholds that limit the creation of new family–classes, unless all of their schema–events are of different harmonic–class from those existing, and the inclusion of variations that have only half of their voicing information different (equal to the `DMT` approximate configuration). The *Balanced* configuration examined thresholds with values between the two extreme cases of *Loose* and *Strict* configuration.

6.2.1.4 Input data organisation

In the first stage of the `SD` operation, the `SA.ISED` process attempts to segment the examined score. To examine whether the size of an examined score, i.e., an excerpt or a full part, affects the performance of the `SD` operation, the following three input–data scenarios were considered:

- The fully annotated *Galant schemata* dataset of excerpts/*partimenti* (*see* Appendix B.1);

- The sparsly annotated *Sonatas schemata* dataset (*see* Appendix B.2), and
- A mixed dataset utilising scores from both of the above datasets.

6.2.1.5 Performance measurements

To measure the performance of the discovery task and examine whether the extracted by the `SD` operation family–classes are valid, two methods were considered:

- Identifying known Galant archetypes in the discovered family–classes, and
- Identifying those (known) discovered family–classes in annotated works.

Since the discovered Galant family–classes reside in the `GMSC.SF`, the first method examines whether any of the existing (known) family–classes are discovered. If known Galant family–classes are found in the discovered family–classes, that would indicate a successful discovery operation.

The second method examines whether the discovery of the known classes was efficient, performing the search task (with exact matching) to obtain the values of recall and precision. If a search for a discovered family–class yields high recall, that would mean that the correct definition of that class was extracted. If a search for a discovered family–class yields high precision, that would mean that mostly the correct definition of that family–class was extracted. Additional performance indicators include the number of discovered family classes, a value that should reflect on the thresholds of the equilibrium process.

6.2.1.6 Run–time configurations

The possible configurations of the `SD` operation computational experiments include the following parameters and values.

- The configuration of the Schematic Analysis operation (`SA`, *see* Section 4.3):
  - `SFFT`, and
  - `SAAE`.
- The Discovery–Matching–Threshold:
  - Exact, and
  - Approximate.
- The Equilibrium thresholds configuration:
  - Strict;
  - Loose, and
  - Balanced.
- The input–data configuration:
  - Excerpts;
  - Parts, and
  - Mixed.

The above parameters and values may be expressed by the following set:

```
(SA-config(SFFT, SAAE),
 Discovery matching threshold(Exact, Approximate),
 Equilibrium thresholds(Balanced, Strict, Loose))
 Data(Excerpts, Parts, Mixed)
```

Excluding the input–data configuration, the above parameter set and their values yield twelve `SD` configurations (*see* Table 6.2). Each of the above `SD`

Table 6.2: The configurations of the `SD` operation.

	Schematic analysis configuration	Discovery Matching	Equilibrium thresholds
`SFFT-EB`	`SFFT`	Exact	Balanced
`SFFT-EL`	`SFFT`	Exact	Loose
`SFFT-ES`	`SFFT`	Exact	Strict
`SFFT-AB`	`SFFT`	Approximate	Balanced
`SFFT-AL`	`SFFT`	Approximate	Loose
`SFFT-AS`	`SFFT`	Approximate	Strict
`SAAE-EB`	`SAAE`	Exact	Balanced
`SAAE-EL`	`SAAE`	Exact	Loose
`SAAE-ES`	`SAAE`	Exact	Strict
`SAAE-AB`	`SAAE`	Approximate	Balanced
`SAAE-AL`	`SAAE`	Approximate	Loose
`SAAE-AS`	`SAAE`	Approximate	Strict

configurations may be tested by one of the three input–data configurations. The examined configurations are shown when presenting each computational experiment (*see* below).

6.2.1.7 Results format

The results of the `SD` operation are displayed in the form shown in Table 6.3. Each row represents a configuration of the `SD` operation, and each column, the archetype of the discovered family–class that is most similar to a known Galant archetype. In addition, each cell includes the total number of different exemplars in the exemplar–base, the search results (with exact matching) of the discovered family–class in the examined dataset (in terms of recall and precision), and the Exemplar–Difference–Vector between the discovered archetype and the Galant.

Table 6.3: The format of the Galant discovery computational experiments.

	< Galant schema family–class >
< `SD` configuration > < Number of discovered family–classes >	<Number of exemplars > *Search results* < Recall >, < Precision> *Most similar to the Galant discovered family–class* < melody–line > < intervals from bass > < bassline > *The Exemplar–Difference–Vector,* ***`EDV`*** < Differences *per* schema–event >

6.2.2 A. Discovery of Galant family–classes from excerpts

The first computational experiment utilised the *Galant schemata* dataset of excerpts (*see* Appendix B.1) and examined the different `SA` and `DMT` configurations under the same `GMSC.EQ` configuration (*Balanced*, *see* Table 6.1). The four `SD` configurations examined with this dataset were:

- `SFFT-EB`;
- `SFFT-AB`;
- `SAAE-EB`, and
- `SAAE-AB`.

The examined Galant family–types include:

- The *Romanesca*;
- The *Do–Re–Mi*;
- The *Meyer*;

Table 6.4: The results of the first musical schemata discovery computational experiment (Table 1 of 2).

	ROMANESCA	DO–RE–MI	MEYER
`SFFT-EB` (21)	(9) 0.764, 0.648 3 2 1 7 6 5 53 53 53 53 53 53 1 5 6 3 4 1 0 0 0 0 0 0	(8) 0.732, 0.682 1 2 3 53 63 53 1 7 1 0 0 0	(9) 0.712, 0.710 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0
`SFFT-AB` (27)	(11) 0.812, 0.583 3 2 1 7 6 5 53 53 53 53 53 53 1 5 6 3 4 1 0 0 0 0 0 0	(13) 0.797, 0.598 1 2 3 53 63 53 1 7 1 0 0 0	(17) 0.736, 0.698 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0
`SAAE-EB` (18)	(8) 0.732, 0.712 3 2 1 7 6 5 53 53 53 53 53 53 1 5 6 3 4 1 0 0 0 0 0 0	(6) 0.758, 0.674 1 2 3 53 63 53 1 7 1 0 0 0	(5) 0.744, 0.724 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0
`SAAE-AB` (20)	(10) 0.786, 0.562 3 2 1 7 6 5 53 53 53 53 53 53 1 5 6 3 4 1 0 0 0 0 0 0	(9) 0.814, 0.582 1 2 3 53 63 53 1 7 1 0 0 0	(10) 0.788, 0.680 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0

- The *Fenaroli*;
- The *Prinner*, and
- The *Clausula*.

6.2.2.1 Results description

In the first computational experiment that tested the `SD` operation, all of the six examined Galant family–classes were found in the discovered family–classes, regardless of the `SD` configuration tested. However, the search results of the discovered family–classes show low values for the metrics of recall (maximum value of 0.822 for the *Prinner* with `SFFT-AB`) and precision (maximum value of 0.724 for the *Meyer* with `SAAE-EB` configuration). Considering the schematic analysis configurations, the `SFFT-xx` configurations yielded more family–classes and more exemplars *per* family–class than their corre-

Table 6.5: The results of the first musical schemata discovery computational experiment (Table 2 of 2).

	PRINNER				FENAROLI				CLAUSULA		
SFFT-EB (21)	(8) 0.798, 0.715				(9) 0.712, 0.702				(17) 0.780, 0.665		
	6	5	4	3	4	3	7	1	3	2	1
	53	63	63	53	65	53	63	63	53	63	53
	4	3	2	1	7	1	2	3	4	5	1
	0	0	0	0	0	0	0	0	0	0	0
SFFT-AB (27)	(13) 0.822, 0.694				(15) 0.740, 0.680				(22) 0.812, 0.604		
	6	5	4	3	4	3	7	1	3	2	1
	53	63	63	53	65	53	63	63	53	63	53
	4	3	2	1	7	1	2	3	4	5	1
	0	0	0	0	0	0	0	0	0	0	0
SAAE-EB (18)	(6) 0.765, 0.720				(7) 0.698, 0.712				(15) 0.754, 0.652		
	6	5	4	3	4	3	7	1	3	2	1
	53	63	63	53	65	53	63	63	53	63	53
	4	3	2	1	7	1	2	3	4	5	1
	0	0	0	0	0	0	0	0	0	0	0
SAAE-AB (20)	(11) 0.815, 0.682				(10) 0.732, 0.658				(20) 0.772, 0.612		
	6	5	4	3	4	3	7	1	3	2	1
	53	63	63	53	65	53	63	63	53	63	53
	4	3	2	1	7	1	2	3	4	5	1
	0	0	0	0	0	0	0	0	0	0	0

sponding `SAAE-xx` configurations. The `SFFT-EB` configuration returned 21 family–classes while the `SAAE-EB` 18, and the `SFFT-AB` configuration yielded 27 family–classes while the `SAAE-AB` resulted in 20. Comparing the performance of the different `DMT` configurations, the *exact* configurations (`SFFT-EB` and `SAAE-EB`) resulted in lower numbers for both, the discovered family–classes and the exemplars *per* family–class than their corresponding *approximate* configurations (`SFFT-AB` and `SAAE-AB`). The above result patterns were observed for all the examined family–classes.

6.2.3 B. Discovery of Galant family–classes in keyboard sonata parts

The second computational experiment of the `SD` operation utilised the *Sonatas schemata* dataset of complete sonata parts (*see* Appendix B.2) . This computational experiment examined the different configurations of the `GMSC.EQ` thresholds mainly in the `SAAE SA` configuration, also utilising the *exact* `DMT` configuration.

The four `SD` configurations examined with this dataset were:

- `SFFT-EB`;
- `SAAE-EB`;
- `SAAE-EL`, and
- `SAAE-ES`.

The examined Galant family–types include:

- The *Meyer*;
- The *Prinner*, and
- The *Clausula*.

6.2.3.1 Results description

In the second computational experiment, all three Galant family–classes were also discovered regardless of the `SD` configuration. Similarly to the first computational experiment, the search results of the discovered family–classes show low values for the metrics of recall (maximum value of 0.802 for the *Prinner* with the `SFFT-EL` configuration) and precision (maximum value of 0.712 for the *Meyer* with the `SAAE-ES` configuration). Considering the `GMSC.EQ` configurations, the *Strict* configuration (`SAAE-ES`) resulted in

Table 6.6: The results of the second musical schemata discovery computational experiment.

	MEYER	PRINNER	CLAUSULA
SFFT-EB (22)	(9) 0.798, 0.698 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0	(8) 0.798, 0.708 6 5 4 3 53 63 63 53 4 3 2 1 0 0 0 0	(20) 0.792, 0.7 3 2 1 53 63 53 4 5 1 0 0 0
SAAE-EL (28)	(17) 0.774, 0.638 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0	(13) 0.802, 0.686 6 5 4 3 53 63 63 53 4 3 2 1 0 0 0 0	(26) 0.8, 0.692 3 2 1 53 63 53 4 5 1 0 0 0
SAAE-ES (19)	(5) 0.790, 0.712 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0	(6) 0.772, 0.692 6 5 4 3 53 63 63 53 4 3 2 1 0 0 0 0	(18) 0.794, 0.698 3 2 1 53 63 53 4 5 1 0 0 0
SAAE-EB (21)	(10) 0.782, 0.688 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0	(11) 0.794, 0.690 6 5 4 3 53 63 63 53 4 3 2 1 0 0 0 0	(22) 0.796, 0.695 3 2 1 53 63 53 4 5 1 0 0 0

less family–classes than all the other configurations (19), also with the least number of exemplars in the exemplar–bases of the discovered family–classes for all of the discovered family–classes. The *Loose* configuration yielded the most family–classes (28), also with the most exemplars in their exemplar–bases. The *Balanced* thresholds configuration (SAAE-EB) resulted in number of family–classes and number of exemplars *per* family–class that are between the *Loose* and the *Strict* configurations. Comparing the different SA configurations (SFFT-EB and SAAE-EB), similarly to the first computational experiment, the latter yielded less family–classes and exemplars *per* family–class. Again, the above results were consistent regardless of the family–class examined.

6.2.4 C. Discovery of Galant archetypes in mixed datasets

The third computational experiment of the `SD` operation utilised scores from both datasets (i.e., the *Galant schemata* and the *Sonatas schemata* datasets). The examined configurations are the same as those in the second (`SD`) computational experiment (*see* Section 6.2.3):

- `SFFT-EB`
- `SAAE-EL`
- `SAAE-ES`
- `SAAE-EB`

The examined Galant family–types include:

- The *Meyer*;
- The *Prinner*, and
- The *Clausula*.

6.2.4.1 Results description

Similarly to the previous computational experiments, all three Galant family–classes were also discovered. Likewise, when searching for the discovered family–classes, the metrics of recall and precision were also low. When searching for the discovered family–classes, the `SAAE-EL` configuration returned the best recall value (0.784 for the *Prinner*) and the `SAAE-ES` yielded the best precision value (0.698 for the *Meyer*). Considering the `GMSC.EQ` configurations, the *Strict* configuration (`SAAE-ES`), similarly to the second computational experiment, also resulted in less family–classes than all the other configurations (19), also with the least number of exemplars in the exemplar–bases of the discovered family–classes. The *Loose* configuration (`SAAE-EL`)

Table 6.7: The results of the third musical schemata discovery computational experiment.

	MEYER	PRINNER	CLAUSULA
`SFFT-EB` (22)	(9) 0.764, 0.622 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0	(10) 0.765, 0.678 6 5 4 3 53 63 63 53 4 3 2 1 0 0 0 0	(17) 0.752, 0.628 3 2 1 53 63 53 4 5 1 0 0 0
`SAAE-EL` (25)	(17) 0.755, 0.638 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0	(13) 0.784, 0.652 6 5 4 3 53 63 63 53 4 3 2 1 0 0 0 0	(22) 0.738, 0.654 3 2 1 53 63 53 4 5 1 0 0 0
`SAAE-ES` (19)	(5) 0.702, 0.698 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0	(7) 0.742, 0.670 6 5 4 3 53 63 63 53 4 3 2 1 0 0 0 0	(15) 0.744, 0.626 3 2 1 53 63 53 4 5 1 0 0 0
`SAAE-EB` (23)	(10) 0.736, 0.654 1 7 4 3 53 63 65 53 1 2 7 1 0 0 0 0	(11) 0.758, 0.662 6 5 4 3 53 63 63 53 4 3 2 1 0 0 0 0	(20) 0.732, 0.638 3 2 1 53 63 53 4 5 1 0 0 0

yielded the most family–classes (25), also with the most exemplars in their exemplar–bases. The *Balanced* thresholds configuration (`SAAE-EB`) returned 23 family–classes. Comparing the different `SA` configurations (`SFFT-EB` and `SAAE-EB`), similarly to the previous computational experiments, the latter yielded less number of family–classes (23) and exemplars *per* family–class. The above results were consistent regardless of the family–class examined.

6.2.5 Total results description

The `SD` operation was tested through a subset of the configurations that stemmed from the parameterisation of the task (*see* Table 6.2). In all three computational experiments, the examined Galant family–classes were discovered. Comparing the search results among the three computational experiments, the first computational experiment (*see* Section 6.2.2) achieved the best results, followed by the third computational experiment (*see* Section 6.2.4), with the results of the second computational experiment (*see* Section 6.2.3) coming last. Considering the different `SA` configurations, the `SFFT` configuration returned more family–classes than the `SAAE` for all of the examined Galant family–classes, also in all of the computational experiments. The same behaviour was observed for the *approximate* `DMT` value over the *exact*.

6.3 Findings

The results from the three computational experiments that tested the `SD` suggest a functional model with consistent behaviour, though with low performance. Despite the fact that all of the examined family–classes were discovered in all of the computational experiments, when searching for the discovered family–classes, both metrics of recall and precision were low. This suggests that either the `SD` operation or its computational implementation is not performing as it should. In regards to the `SA` configurations, the `SAAE` configuration, yielding less schema–events than the `SFFT` configuration, seems to be more appropriate for the `SD` task. Considering the performance of the `SD` operation under different datasets, when excerpts are processed, search results were better than when complete parts were considered. This indicates that when the Intra–Score Exemplar Discovery process (`SA.ISED`) is activated, it does not have a positive effect. Considering the effects of the various configurations of the `GMSC.EQ` configurations, in spite of the consistent behaviour throughout all of the computational experiments, further adjustments are required so that discovered family–classes are kept minimal. This issue might be addressed considering *adaptive* thresholds instead of the *static* values that were utilised throughout each computational experiment.

6.4 Summary

This chapter presented the computational experiments that tested the operation of Galant family–class discovery (`SD`, *see* Section 3.5). The examined model is functional and does yield correct results, discovering all of the examined Galant musical schemata family–types. However, when searching for the discovered family–classes, the values of both recall and precision were low, most probably due to the numerous variants that were integrated on each family–class. Regardless of the low performance of the model, this was an implementation of a novel system, and the fact that it is functional, enables further experimentation and improvements.

7 Conclusions

This study aimed to explicate on the mental processes involved in the automatic identification of musical schemata, suggesting music–analytical methodologies. Under the premises of the Galant Musical Schemata Theory (`GMST`) and schemata in Psychology, the goal of this study was examined by considering a cognitive architecture facilitating music–analytical operations with polyphonic patterns (*see* Chapter 3). First, in a reductionistic approach, the automatic identification of musical schemata was segmented into three accumulative 'identification' scenarios concerning: the interpretation of music notation into salient (schematic) structures (`SA`), the example–based learning of Galant archetypes (`XL`), and the discovery of Galant family–classes (`SD`). Then, utilising the 'infrastructure' of the cognitive model, each of the above operations was translated into a music–analytical syllogism. To verify whether the suggested cognitive model and its music–analytical syllogisms performed the intended tasks, the 'technological' approach was employed. The computational implementation of the cognitive model and its operations, therefore, enabled computational experiments, and the association of performance metrics with specific parts of the model.

The operation of schematic analysis (`SA`, *see* Section 3.3) examined a method for the reduction of music notation that yields the Galant 'musical surface' of schema–event progressions. The `SA` operation was deemed essential for the examined model, as the operations of learning and discovery rely on such a highly–structured representation to function. The main idea of the `SA` was to 'construct' the Galant surface by increasing the awareness of musical contexts and entities; initiating with music–theoretic elements to identify Galant archetype elements (schema–voices and –events). This approach was facilitated by grouping functions based on the notions of 'clarity', 'positional' and 'transitional significance', and on models that quantify and classify temporal regularity. The `SA` operation was examined through the search task (*see* Chapter 4) for seven Galant family–classes (i.e., the *Romanesca*, the *Meyer*, the *Prinner*, the *Do–Re–Mi*, the *Fenaroli*, the *Quiesenza*, and the *Clausula*). Analysing the results for a set of `SA` configurations, the method achieved the desired goal of extracting schema–event progressions, but with decreasing performance when elaborated. This outcome suggests that the examined method may perform needless processes. Although some steps in the `SA` operation seem reasonable when applying them by thought (e.g., the extraction of schema–voices), in (computational) practice, such over–specificity may be avoided through integration (e.g., by the extraction of schema–event progressions).

The example–based learning operation (`XL`, *see* Section 3.4) was analysed into a set of learning modes from Developmental Psychology. Utilising the `SA` representation, the `XL` operation introduced concepts such as the *exemplar* and the *Exemplar–Difference–Function/–Vector*, and the *family–class* and its *class–similarity* function. The main idea of the `XL` operation was to maintain a composite representation for Galant family–classes, each comprising of: an archetype (exemplar), an exemplar–base of previous training examples, and a class–similarity function that aggregates Exemplar–Difference–Vectors from the exemplar–base. The learning process in the `XL` operation was modelled with the association of the three modes of learning, namely *accretion*, *tuning* and *restructuring*, to the kinds of differences that may appear among the training examples of a family–class (the categorisation of Exemplar–Difference–Vectors, `EDV`s). With this method, a Galant family–class aggregates (annotated) exemplars, and adjusts its class–similarity function to the new training information. The `XL` operation was examined with the classification task (*see* Chapter 5). The performance of the model was examined for three Galant family–classes (i.e., the *Meyer*, the *Prinner*, and the *Clausula*) through three parameters concerning: different `SA` configurations, exact and approximate recognition, and training scenarios (i.e., with exemplars, intra–, cross–, and inter–dataset). Analysing the results of the classification computational experiments, the model appears to have a consistent behaviour in all of the training configurations. However, the constant decrease of performance when the model elaborates, as in the case of the `SA` operation, suggests that parts of either the model or its computational implementation do not function as they should. Despite the relatively low performance of the computational implementation of the model when performing the classification task (failing to identify one in five true–positives), a functional workflow was presented, enabling future improvements.

The discovery of – similar to the Galant – musical schemata family–types (`SD`, *see* Section 3.5) was analysed by utilising and extending the previously examined operations of `SA` and `XL`. The main idea in the `SD` operation was first, to identify repeated exemplars within a score, so as to create a classification of repetitive (if any) and 'uncategorised' exemplars, then to compare these with discovered family–classes and 'uncategorised' exemplars from previous intra–score analyses, and possibly discover or integrate new repetitions. The above method aggregates score–wise classifications, and the integration of new information was controlled by the process of *equilibrium* and its two thresholds for the minimum diversity among discovered family–classes, and the maximum variability among the exemplars of each exemplar–base. The `SD` operation was examined with the Galant family–class discovery task (*see* Chapter 6). The parameters of the `SD` computational experiments concerned the configuration of the `SA` operation, the value of the Discovery–Matching–Threshold (`DMT`), the configuration of the thresholds of the equilibrium process (*diversity* and *variability*), and the selection of input data. The performance of the `SD` operation was measured by identifying known Galant archetypes in the discovered family–classes, and by searching the latter on the examined dataset to obtain the values of recall and precision. Analysing the results of the discovery task, the model is functional and consistent, discovering all of the known Galant archetypes. However, when searching for the discovered family–classes in the dataset from which they were extracted, the precision was generally low, indicating that these family–classes contain additional variants that are not valid.

The examined approach is a prototypical work, and as such, it has great potential for all kinds of improvements. This study has focused on the information unit of an exemplar, and examined a range of methods for its identification in scores, their comparison, and their categorisation in family–classes.

The inter–relation of exemplars, as well as their phrase–related properties were not examined. By considering such information, morphological analysis may be enabled. For example, tracking the appearance of the *Clausula* family–class may facilitate the identification of phrases. In turn, this may facilitate the score summarisation task (Hirata and Matsuda, 2003), a useful automation for both musicologists and performers. In general, this work established a computational methodology providing a base model for further experimentation, and an alternative to the machine–learning methodologies.

Concluding, the main challenge of this study was the multiplicity of the parameters and the processing stages that stemmed from the analytical process. Human–generated complexity is prone to error, and in particular, in the absence of definitive theories about the machinations within the cognitive realm, the modelling of high–level cognitive operations, such as the automatic identification of musical schemata, resorts to speculation. This approach led to overwhelming complexities and unavoidable assertions. Despite these inherent difficulties, the author strongly believes that such approaches, where human knowledge is to be codified, should be preferred over 'industry–level' approximations, and should benefit from collective efforts, such as crowd–sourcing. Such approaches may overcome the overwhelming complexities of the examined tasks and facilitate the formalisation of human intellect.

Appendices

A
Examined Galant musical schemata family–classes

In his book, *Music in the Galant style* (Gjerdingen, 2007), Gjerdingen documents more than 20 family–classes of Galant music schemata. This study examined the following:

- The *Romanesca*;
- The *Do–Re–Mi*;
- The *Meyer*;
- The *Fenaroli*;
- The *Prinner*;
- The *Quiescenza*, and
- The *Clausula*.

The next segments define the archetypes of the above Galant family–classes in their schematic form (*see* Section 2.1.1, Gjerdingen, 2007). The numerous variants are displayed in a tabular format, where: each column represents a schema–event; the top row (`M`) shows the scale–degrees of the melody; the second row (`H`) displays the harmony of the schema–event as intervals from bass, and the last row (`B`) shows the scale degrees of the bass. Dots ('.') indicate multiple values and slashes ('/') suggest alternative values. When multiple scale–degrees are present, these indicate melodic movement within the schema–event. Schema–events in parentheses are permissible omissions.

A.1 The *Romanesca* family–class

The *Galant Romanesca* archetype

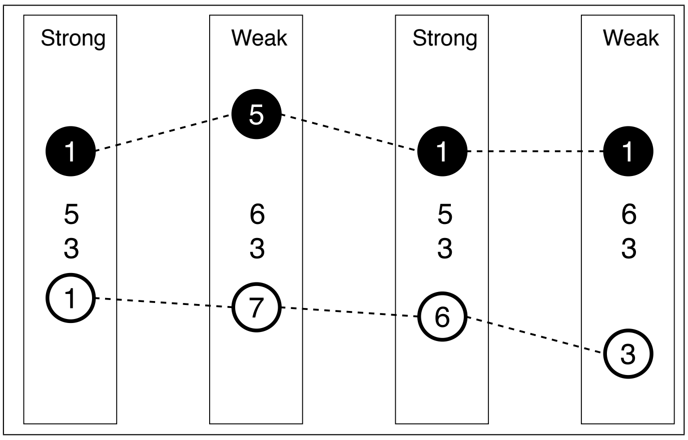


Figure A.1: The *Galant Romanesca* archetype in schematic form.

The *Stepwise bass Romanesca* archetype

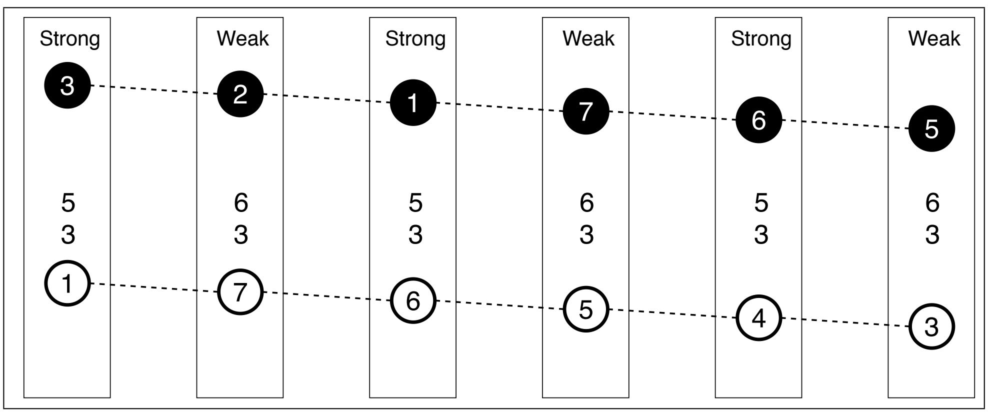


Figure A.2: The *Stepwise bass Romanesca* archetype in schematic form.

The *Leaping bass Romanesca* archetype

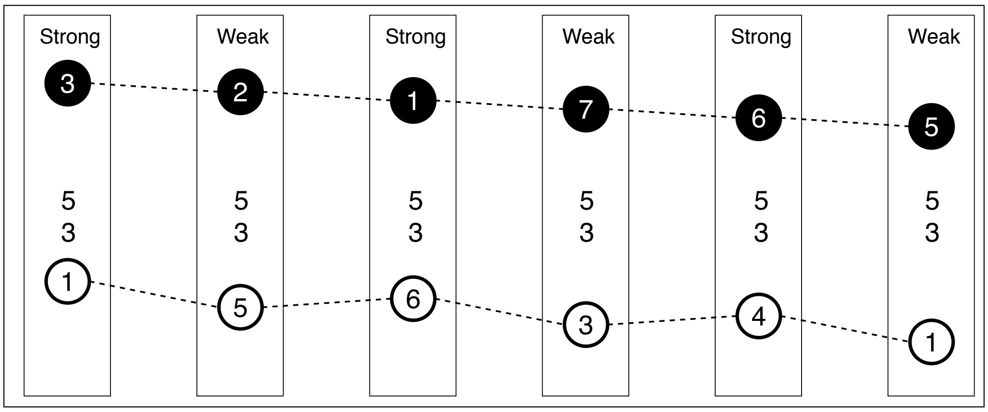


Figure A.3: The *Leaping bass Romanesca* archetype in schematic form.

A.2 The *Do–Re–Mi* family–class

Archetype

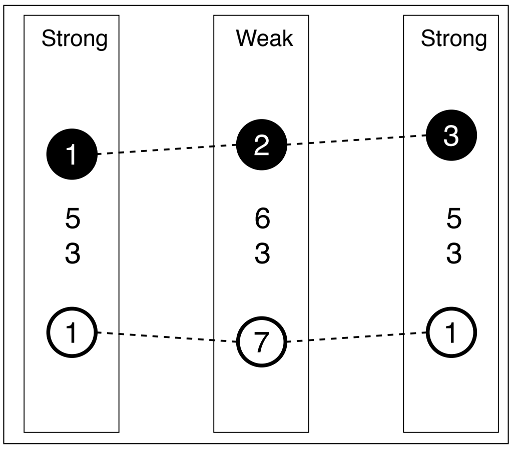


Figure A.4: The *Do–Re–Mi* archetype in schematic form.

The *Do–Re...Re–Mi* variant

M	1	2	2	3
H	53	63	63	53
B	1	7	7	1

A.3 The *Meyer* family–class

Archetype

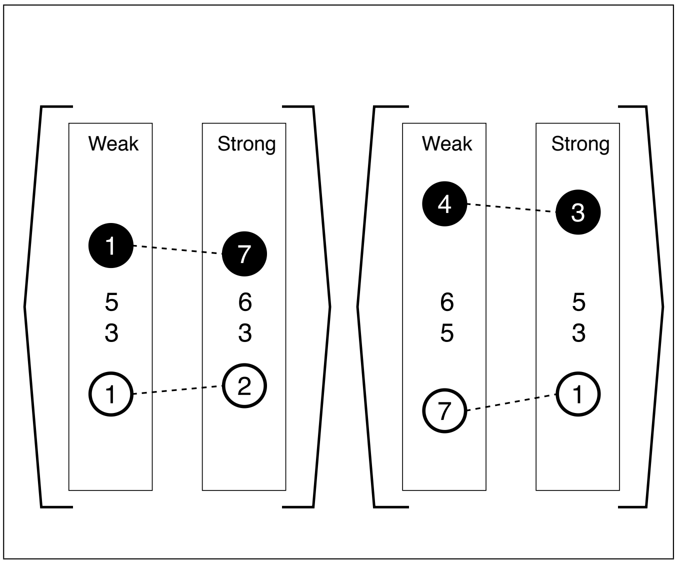


Figure A.5: The *Meyer* archetype in schematic form.

The *Aprile* variant

M	1	7	2	3
H	53	43	65	53
B	1	2	5	1

A.4 The *Fenaroli* family–class

Archetype

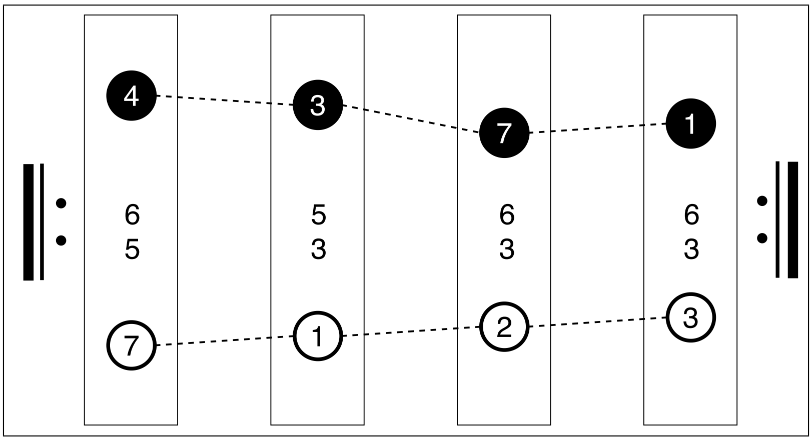


Figure A.6: The *Fenaroli* archetype in schematic form.

A.5 The *Prinner* family–class

Archetype

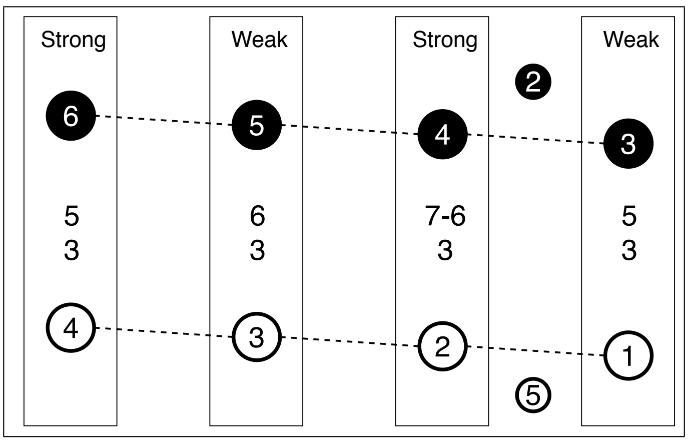


Figure A.7: The *Prinner* archetype in schematic form.

A.6 The *Quiescenza* family–class

Archetype

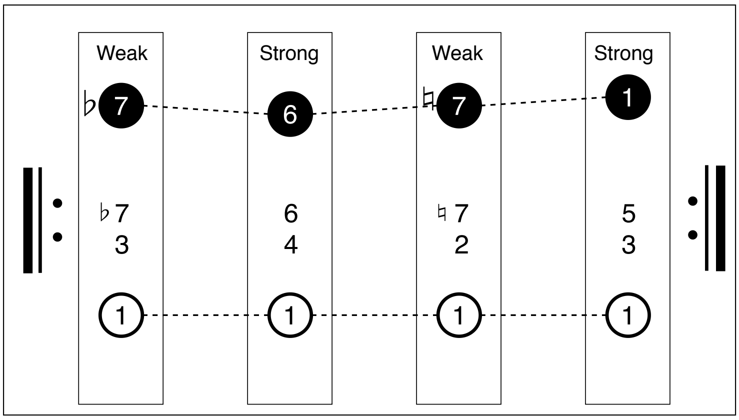


Figure A.8: The *Quiesenza* archetype in schematic form.

A.7 The *Clausula* family–class

A.7.1 Bass movement 5-1

Simple and Compound forms of the standard Galant *Clausula*

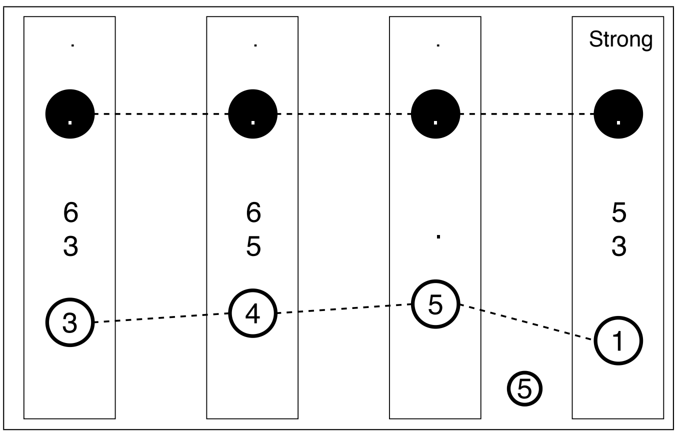


Figure A.9: *Simple* and *Compound* forms of the standard *Galant Clausula*. The Compound form has an additional schema–event before the last.

Simple Galant *Clausula* with the *Mi–Re–Do* melodic line

M	.	3	2	1
H	63	65	.	53
B	3	4	5	1

The *Complete cadence*

M	2	1
H	.	53
B	5	1

The *Perfecta / MI–RE–DO*

M	2	2	1
H	653	75	53
B	4	5	1

The *Perfectissima DO–SI–DO*

M	1	7	1
H	53	753	53
B	4	5	1

Compound Galant *Clausula* with the *Mi–Re–Do* melodic line

M	(1/3)	2	2	7	1
H	(.)	65	64	53	53
B	(3/1)	4	5	5	1

The *Grand Galant Cadence*

M	1	6	5	2	1
H	63	53	53	53	53
B	3	4	5	5	1

The *Cudworth Cadence*

M	1	7654	3	2	1
H	53	.	64	53	53
B	3	4	5	5	1

The *Imperfecta / INCOMPLETE*

M	.	4	3
H	53	7#3	5#3
B	4	5	1

A.7.2 Bass movement 7-1

The *Comma*

M	(5/2)4	3
H	.	.
B	7	1

The *Long Comma*

M	5(4)	4(2)	3
H	.	.	53
B	6	7	1

The *Cantizans*/ JOMMELLI

M	6	5
H	753	#3
B	7	1

The *Cantizans*/ COMMA

M	54	5
H	65	#3
B	7	1

A.7.3 Bass movement 2-1

The *Clausula Vera*

M	1	7	1
H	753	#643	53
B	2	2	1

The *Deceptive Clausula*

M	.	.	3	3/5
H	6	65	.	63
B	3	4	5	6

Half Cadence

M	1/3	(.)	3	2
H	63/53	(53)	64	53
B	3/1	(4)	5	5

A simplified *Converging Cadence*

M	3	2	1	7
H	53	63	65	53
B	3	4	#4	5

Converging Cadence **variant**

M	1	7654	3	(3)2
H	6(65)	63	65	53
B	3	4	#4	5

Converging Cadence **variant**

M	3/5/1	6543	21	7
H	53	63	65	53
B	3	4	#4	5

Converging Cadence **variant**

M	4	3	2	2	1
H	53	75	65	5#3	5#3
B	4	#4	#4	5	1

Converging Cadence **variant**

M	6	6	5	4	3
H	53	65	5#3	75#3	5#3
B	4	#4	5	5	1

B
Annotated datasets

This section overviews the datasets of this study regarding the Galant musical schemata annotated corpora. Each annotated dataset is a collection of digital scores containing complete parts or excerpts in MusicXML format (Good, 2001), accompanied by expert–annotations as Galant musical schemata, family–type and variant, labelled measure–ranges (in JSON format). These are the following:

- *Galant schemata*, an almost complete digitisation of the score examples in *Music in the Galant style* (Gjerdingen, 2007), and
- *Sonatas schemata*, a corpus of keyboard sonatas by Haydn, Mozart and Beethoven, with annotations for three Galant musical schemata prototypes, namely the *Meyer*, the *Prinner*, and the *Cadence*.

B.1 The *Galant schemata* dataset (Gjerdingen, 2007)

The *Galant schemata* dataset is a digitisation of the examples found in *Music in the Galant style* Gjerdingen (2007), containing 304 examples, of which 288 are excerpts a few measures in length and 16 are complete parts. In total, the *Galant schemata* dataset includes 998 annotation entries for 25 musical schemata family–types and variants. Table B.1 presents the number of generic and variant forms of each musical schema type in the dataset.

Table B.1: The *Galant schemata* dataset of musical schemata annotations.

Family type	**Total**	**Generic**	**# of variants**
Aprile	2	2	0
Circle of 5ths	1	1	0
Clausula	337	31	110
Coda	8	8	0
Do–Mi–Sol	4	4	0
Do–Re–Mi	52	38	3
Do–Si–Do	5	5	0
Fenaroli	71	68	2
Folia	1	1	0
Fonte	51	49	2
High–2–Drop	1	1	0
Indugio	24	24	0
Jupiter	8	8	0
Meyer	35	10	12
Mi–Re–Do	15	15	0
Monte	38	27	2
Pastorella	6	6	0
Ponte	36	25	1
Prinner	149	105	6
Quiescenza	67	63	2
Romanesca	58	33	2
Sol–Fa–Mi	27	23	1
Triadic Flourish	1	1	0

The following is an example annotation entry in the JSON annotation–file:

```
{
 "info": "Annotations from Gjerdingen, R. (2007). Music in the galant style.
         Oxford University Press.",
 "annotations" :
```

```
[
 {
   "file_name": "ex2--1",
   "file_type": "excerpt",
   "file_info": "Tritto, from a partimento in F major, m. 1 (ca. 1810--20)
                 Example 2--1, p.26, Gjerdingen, R. (2007)",
   "schemata":
     [
      {
         "schema_family": "Romanesca",
         "schema_variant": "Leaping bass",
         "measures": [1,7]
      },
      { ,... another schema annotation entry }
     ]
  },
  {        ... another schema annotation entry }
]
}
```


B.2 The *Sonatas schemata* dataset

The *Sonatas schemata* dataset includes 50 keyboard and piano sonata parts from Haydn (15 parts), Mozart (15 parts) and Beethoven (20 parts), and Galant schemata annotations for three musical schemata types: the *Meyer*, the *Prinner*, and the *Clausula*. The quantities of musical schemata annotations are shown in Table B.2 and the sonata parts are listed in Table B.3.

The following is an example annotation entry in the JSON annotation–file:


```
{
[
  {
   "file": "beethoven_ludwig_van_piano_sonata_no_2_op_2_no_2_a_...",
```

Table B.2: Sonatas schemata annotations.

Composer	Haydn		Mozart		Beethoven		ALL	
Parts	15		15		20		50	
Schemata	Uniques	All	Uniques	All	Uniques	All	Uniques	All
Meyer	2	74	7	46	12	48	21	168
Prinner	2	24	9	58	2	8	13	90
Cadence	7	87	26	156	6	27	39	270
ALL	11	185	42	265	20	87	73	537

```
   "TimelineAnno" :
       [
        [[1,8], "p1_Meyer_variant_A"],
        [[7,8], "p2_Clausula_Bass_51"],
        [[8,12], "p3_unknown_texture_p1"],
        [[12,17], "p4_unknown_texture_p3"],
        [[18,19], "p5_Clausula_Bass_51"],
        [[19,23], "p6_Meyer_variant_B"],
        [[23,24], "p7_unknown_texture_p6"],
        [[25,30], "p8_unknown_texture"],
        [[32,40], "p9_Meyer_variant_A_repeat_p1"],
        [[39,40], "p10_Clausula_Bass_51_repeat_p2"],
        [[40,44], "p11_Clausula_Bass_1451"],
        [[45,46], "p12_unknown"],
        [[47,48], "p13_unknown_repeat_p12"],
        [[49,52], "p14_Clausula"],
        [[53,54], "p15_unknown_repeat_p12"],
        [[55,56], "p16_unknown_repeat_p12"],
        [[57,60], "p17_unknown_repeat_p14"],
        [[61,62], "p18_unknown_repeat_p12"],
        [[62,63], "p19_unknown_repeat_p12"],
        [[64,65], "p20_unknown_repeat_p12"],
        [[67,68], "p21_Clausula_Bass_451_Final"]
      ]
  },
  {       ... another sonata part annotations list }
]
}
```

Listing of sonata parts

Table B.3: Keyboard sonatas dataset part listing.

Haydn, J., Partita Hob.XVI:2 in B–flat major, 2nd mvt. (G minor), Largo
Haydn, J., Divertimento Hob.XVI:12 in A major, 1st mvt., Andante
Haydn, J., Partita Hob.XVI:14 in D major, 2nd mvt., Minuet
Haydn, J., Keyboard Sonata Hob.XVI:18 Op.53 No.3 in B–flat major, 1st mvt., Allegro moderato
Haydn, J., Keyboard Sonata Hob.XVI:20 Op.30 No.6 in C minor, 3rd mvt., Finale. Allegro
Haydn, J., Keyboard Sonata Hob.XVI:21 Op.13 No.1 in C major, 1st mvt., Allegro
Haydn, J., Keyboard Sonata Hob.XVI:23 Op.13 No.3 in F major, 1st mvt., Allegro
Haydn, J., Keyboard Sonata Hob.XVI:23 Op.13 No.3 in F major, 3rd mvt., Finale. Presto
Haydn, J., Piano Sonata Hob.XVI:27 Op.14 No.1 in G major, 3rd mvt., Finale. Presto
Haydn, J., Keyboard Sonata Hob.XVI:33 Op.41 No.2 in D major, 1st mvt., Allegro
Haydn, J., Keyboard Sonata Hob.XVI:34 in E minor, 3rd mvt., Finale. Molto vivace
Haydn, J., Piano Sonata Hob.XVI:38 Op.30 No.4 in E–flat major, 2nd mvt. (C minor), Adagio
Haydn, J., Keyboard Sonata Hob.XVI:48 Op.70 in C major, 1st mvt., Andante con espressione
Haydn, J., Keyboard Sonata Hob.XVI:49 Op.69 in E–flat major, 1st mvt., Allegro
Haydn, J., Keyboard Sonata Hob.XVI:52 Op.92 in E–flat major, 3rd mvt., Finale. Presto
Mozart, W. A., Piano Sonata No.1 in C major K.279/189d, 3rd mvt., Allegro
Mozart, W. A., Piano Sonata No.2 in F major K.280/189e, 2nd mvt., Adagio
Mozart, W. A., Piano Sonata No.3 in B–flat major K.281/189f, 1st mvt., Allegro
Mozart, W. A., Piano Sonata No.4 in E–flat major K.282/189g, 2nd mvt., Menuetto I
Mozart, W. A., Piano Sonata No.5 in G major K.283/189h, 1st mvt., Allegro
Mozart, W. A., Piano Sonata No.6 in D major ("Durnitz") K.284/205b, 2nd mvt., Rondeau en polonaise
Mozart, W. A., Piano Sonata No.7 in C major K.309/284b, 1st mvt., Allegro con spirito
Mozart, W. A., Piano Sonata No.7 in C major K.309/284b, 2nd mvt., Andante un poco Adagio
Mozart, W. A., Piano Sonata No.8 in A minor K.310/300d, 1st mvt., Allegro maestoso
Mozart, W. A., Piano Sonata No.9 in C major K.311/284c, 3rd mvt., Allegro con spirito
Mozart, W. A., Piano Sonata No.11 in A major ("Alla Turca") K.331/300i Op.6 No.2, 1st mvt., Tema con variazione, Allegro
Mozart, W. A., Piano Sonata No.13 B–flat major ("Linz") K.333/315c, 1st mvt., Allegro
Mozart, W. A., Piano Sonata No.14 in C minor K.457 Op.11, 3rd mvt., Molto allegro
Mozart, W. A., Piano Sonata No.16 in C major K.545 ("Semplice"), 1st mvt., Allegro
Mozart, W. A., Piano Sonata No.17 in D major K.576 ("Trumpet" or "Hunt"), 3rd mvt., Allegretto
Beethoven, L. v., Piano Sonata No.2, Op.2 No.2 in A major, 3rd mvt., Scherzo. Allegretto
Beethoven, L. v., Piano Sonata No.3, Op.2 No.3 in C major, 1st mvt., Allegro con brio

Beethoven, L. v., Piano Sonata No.5, Op.10 No.1 in C minor ("Little Pathétique"), 1st mvt., Allegro molto e con brio
Beethoven, L. v., Piano Sonata No.6, Op.10 No.2 in F major, 1st mvt., Allegro
Beethoven, L. v., Piano Sonata No.7, Op.10 No.3 in D major, 3rd mvt., Minuet. Allegro
Beethoven, L. v., Piano Sonata No.7, Op.10 No.3 in D major, 4th mvt., Rondo. Allegro
Beethoven, L. v., Piano Sonata No.11, Op.22 No.2 in B–flat major, 2nd mvt. (E–flat major), Adagio con molto espressione
Beethoven, L. v., Piano Sonata No.12, Op.26 in A–flat major ("Funeral March"), 1st mvt., Andante con variazioni
Beethoven, L. v., Piano Sonata No.13, Op.27 No.1 in E–flat major ("Quasi una fantasia"), 1st mvt., Andante
Beethoven, L. v., Piano Sonata No.13, Op.27 No.1 in E–flat major ("Quasi una fantasia"), 3rd mvt. (A–flat major), Adagio con espressione
Beethoven, L. v., Piano Sonata No.14, Op.27 No.2 in C–sharp minor ("Moonlight"), 2nd mvt. (D–flat major), Allegretto – Trio
Beethoven, L. v., Piano Sonata No.17, Op.31 No.2 in D minor ("The Tempest"), 2nd mvt. (B–flat major), Adagio
Beethoven, L. v., Piano Sonata No.18, Op.31 No.3 in E–flat major ("The Hunt"), 3rd mvt., Minuet. Moderato e grazioso – Trio
Beethoven, L. v., Piano Sonata No.20, Op.49 No.2 in G major ("Leichte Sonata"), 1st mvt., Allegro ma non troppo
Beethoven, L. v., Piano Sonata No.20, Op.49 No.2 in G major ("Leichte Sonata"), 2nd mvt., Tempo di minuet
Beethoven, L. v., Piano Sonata No.21, Op.53 in C major ("Waldstein"), 2nd mvt. (F major), Introduzione. Adagio molto
Beethoven, L. v., Piano Sonata No.26, Op.81a in E–flat major ("Les adieux"), 3rd mvt., Das Wiedersehen. Vivacissimamente (im Lebhastesten Zeitmasse)
Beethoven, L. v., Piano Sonata No.27, Op.90 in E minor, 2nd mvt. (E major), Nicht zu geschwind und sehr singbar vorgetragen
Beethoven, L. v., Piano Sonata No.29, Op.106 in B–flat major ("Hammerklavier"), 1st mvt., Allegro
Beethoven, L. v., Piano Sonata No.31, Op.110 in A–flat major, 1st mvt., Moderato cantabile molto espressivo

C Music as information

There are various kinds of music representations, each serving different usage needs. These range from: physical sound encodings (e.g., WAV and AIFF); transmission codes (e.g., MIDI and OSC), and digital music scores (e.g., kern and MusicXML). Byrd and Crawford (2002) suggest a categorisation based on the extremes of minimum and maximum structure (audio and score), and positions time–stamped events (MIDI events) in the middle. Honing (1993) discusses a number of issues on the representation of time and structure in music, concluding that 'it would be best to construct representations of music so as to be as declarative, explicit and formal as possible'. Honing (1993) also highlights the usefulness of having multiple representations of the same 'world' and associating musical structure with time intervals. In that context, the MUSITECH framework (Weyde, 2005) suggests an integrated representation enabling the modelling of cognitive and analytic musical structures.

C.1 Musical score digital file formats

The Musical Instrument Digital Interface protocol (MIDI[1]) is an industry standard created to pass information between digital instruments, and MIDI

[1]https://www.midi.org/

information can be streamed and stored in files. The protocol considers a time x pitch plane, where notes are tracked with timestamped (absolute temporal values or delta) note–on and –off events for different channels. While this kind of information is suitable for capturing performance data, for music analysis MIDI files are too primitive; these lack basic information such as rhythm and tonality. The MusicXML (Good, 2001) encoding is based on the eXtensible Markup Language (XML[2]) and was created to capture the information of a musical score, including aspects of its visual appearance.

C.2 Operational music encodings

While encoding formats are useful for the storage and playback of musical information, when examining music–analytical tasks, algorithm–friendly representations are more appropriate. The two main categories of music representations that are suitable for algorithmic processing are the so–called *symbolic* and *geometric* or 'numeric' (Mouton and Pachet, 1995). *Geometric* music representations consider notes as 'atoms' in the *time–pitch* plane (often referred to as 'datapoints'), with which composite groups (e.g., chords and melodies) are extracted by applying temporal and pitch constraints on their relations. The most popular *geometric* format is the so–called 'pianoroll' (*see* Figure C.1, F), which is a visualisation of MIDI note–events.

An important differentiation factor in geometric representations is whether the temporal information of the datapoints is quantised or not. For music–theoretic analytical purposes, temporally quantised datapoints are preferred as these most closely represent the score; for performance analysis, non–

[2]http://www.xml.org/

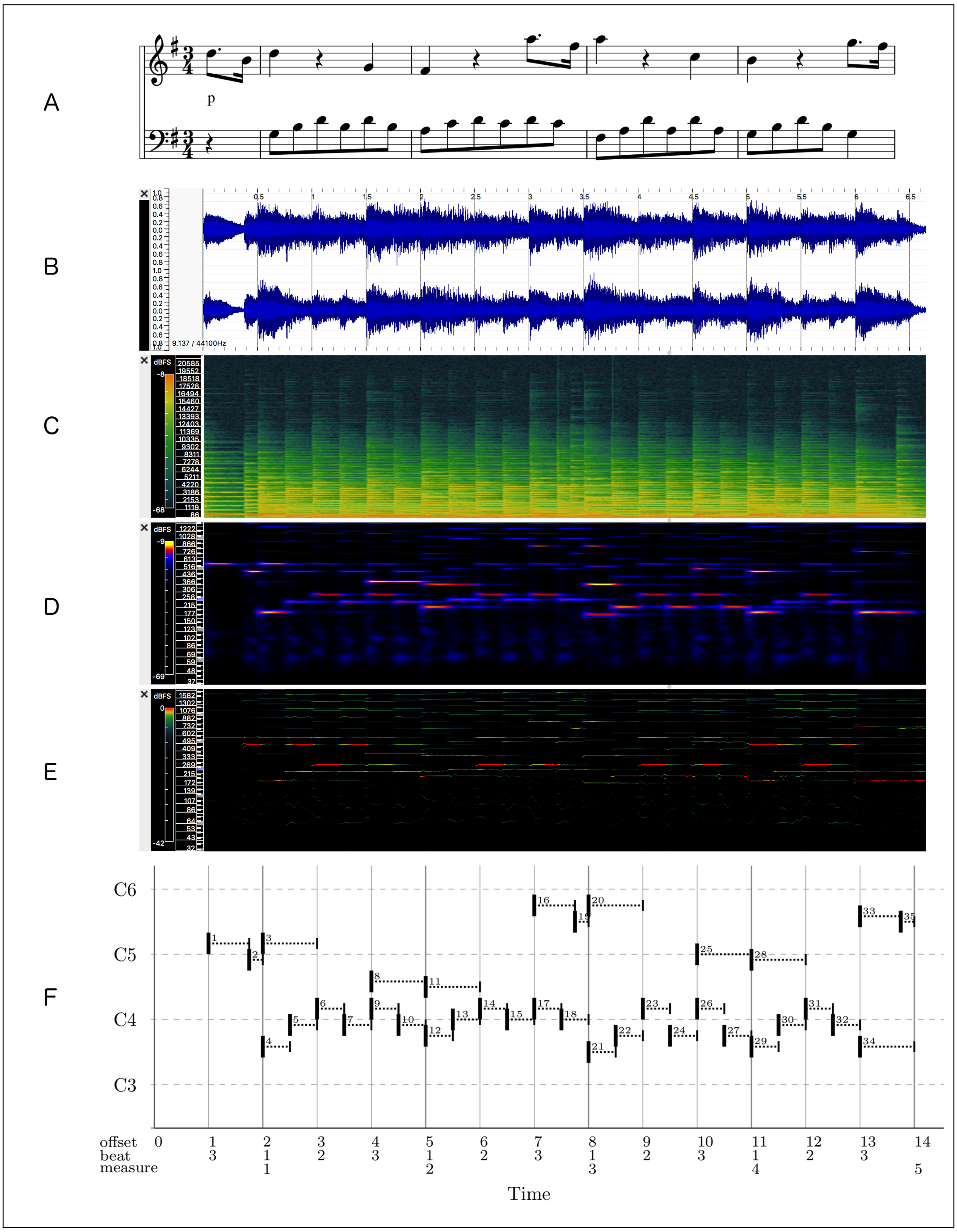


Figure C.1: Music views as information for the first 5 measures of Mozart Sonata N.5 K.283. (A) Common music notation, (B) Waveform, (C) Spectrogram, (D) Melodic Range Spectrogram, (E) Peak Frequency Spectrogram, and (F) MIDI Note events (visualisation from Sonic Visualizer software, Cannam et al., 2010).

quantised information is preferred.

C.2.1 Geometric representations

The utilisation of note elements from music encodings enables ‘direct’ processing of music data, without the need for extraction of contextual information (e.g., tonalities and voices). Starting with geometric representations, Lewin (2007) developed the ‘generalised interval system’, where he examined relations between groups of ‘datapoints’. Another formalism based on datapoints was developed by Conklin and Anagnostopoulou in the so–called ‘viewpoints’ representation (Conklin and Anagnostopoulou, 2001). Viewpoints may be *multiple*, *linked* and *thresholded*, by considering pitch and temporal constraints on ‘datapoint’ relations (Conklin, 2002; Conklin and Anagnostopoulou, 2006; Conklin and Bergeron, 2008, 2010). Geometric representations offer a straightforward mathematical representation and for that reason have been further utilised for monophonic and polyphonic pattern recognition (e.g., ‘Viewpoints’, Conklin, 2002; ‘Directed Interval Class’, Cambouropoulos, 2012) and discovery (‘Structured Induction Algorithm – Transition Equivalent Class’, Meredith et al., 2002; Collins et al., 2016).

In addition to datapoint representations, another common geometric representation is ’Stable Pitch Temporal Segment’ (hereafter SPTS), also known as ‘simultaneity’, ‘chordification’ (Marsden, 2010) and ‘minimum segment’ (Pardo and Birmingham, 2002). The main benefit of SPTS representation is that polyphonic music is represented as a single sequence of SPTS, which Pickens (2001) refers to as ’monophonic reduction’.

Table C.1: Music datapoints. The first five datapoint–entries of the Mozart excerpt in C.1, A, using key and meter information from the musicXML format. The features in parentheses are extracted from the music piece using `music21.streams` (Ariza and Cuthbert, 2011).

	Temporal info			Pitch info	
Unique id	Global offset	Measure	(Beat)	MIDI pitch	(Pitch class)
1	0	0	3	74	D
2	0.75	0	3.75	71	B
3	1	1	1	74	D
4	1	1	1	55	G
5	1.5	1	1.5	59	B
...	...	...	...	...	...

C.2.2 Symbolic music representations

Symbolic music representations use categorical values, hierarchical organisation, and contextualised music information, enabling operations with complex data structures (Smaill and Wiggins, 1990; **?**; Witten and Conklin, 1990). For example, the use of scale degrees instead of MIDI note numbering is regarded as a symbolic representation for pitch information on notes, since the scale context is implied. Similarly, the use of Roman numerals for chord descriptions is also a symbolic representation since the scale context is also implied. In addition to contextualisations of numerical music information, symbolic representations include composite data–structures for musical elements that encompass multiple types of music information. An example of such type of composite music data–construct is the *schema–event* described in Gjerdingen (2007), where a continuous temporal segment is defined by two scale degrees: one for melodic and bass movements, a harmonic description in terms of diatonic intervals from bass, and an overall metric strength, based on the local rhythmic context.

Although any type of non–geometric representation may be classified as symbolic, the complexity of information that may be encoded with some representation schemes exceeds the domain of information representation and is better described within the knowledge representation domain. Such representations include multifeature index structures (Lee and Chen, 1999), logical specifications for abstract representations of music, e.g., 'CHARM' (Pearce, 2002), and ontologies for chord sequences (Wissmann, 2012).

C.3 Frameworks for music–analytical operations

The increasing popularity of personal computers and the continuous improvement of programming languages has facilitated the development of specialised frameworks for music information processing. There are numerous software frameworks for music–related operations in a variety of programming languages. Early frameworks resulted from task–specific studies (e.g., 'Melisma', Sleator and Temperley, 2001 for harmonic analysis), then later frameworks supported more generic music–related operations. Such (public) projects include 'Humdrum' (Huron, 2002), written in C programming language; 'MIR toolbox' (Lartillot et al., 2008; Schedl et al., 2014; Shen, 2007), written in Matlab, 'jMIR' (McKay and Fujinaga, 2006; McKay and Fujinaga, 2009) in Java, 'music21' (Ariza and Cuthbert, 2011) in Python, and MUSITECH (Weyde, 2005). A common feature of these frameworks is an 'internal' ('native') representation scheme for music information that is developed most closely to fit the implementations of related algorithms on each framework, accompanied with parsing functions that convert common music encodings to and from it.

D Computational implementations

This chapter describes the computational implementations of the various information processing methods found in this study. The source code for the examined methods is written in the Python[1] programming language and is organised in the following two libraries:

- A collection of utilities for low–level operations with symbolic music data (the `comutils` library, *see* Section D.1), and
- A system of interconnected classes implementing the architecture of the examined model (the Adaptive Expert System, `AES`, *see* Section D.2).

D.1 A collection of utilities for symbolic music data

The `comutils` library is a collection of utilities for low–level operations with symbolic music data, including classes and modules for the following:

- Conversion of digital scores (in MusicXML[2] encoding, Good, 2001) into 'operational' representations ('datapoints' and 'minimal segments');
- Task–oriented music–analytical functions for feature extraction, includ-

[1]https://www.python.org/
[2]https://www.musicxml.com/

ing:

- Filewise tonality identification;
- Outer–voices identification, and
- Identification of harmonic segments.

D.1.1 Operational representations

The MusicXML encoding offers notation–level information and is part of the so–called 'symbolic music data' formats (*see* Section C). The `comutils` library currently supports the conversion of MusicXML files into two kinds of algorithmic–friendly representations:

- Datapoints, and
- Stable–Pitch–Temporal–Segments (`SPTS`, also termed as 'simultaneities' and 'minimal segments').

To convert a MusicXML file into operational representations, the `comutils` library utilises the `music21`[3] framework (Ariza and Cuthbert, 2011). A MusicXML file is first converted into `music21.stream` structures, the core data container of the `music21` framework, from which custom scripts extract information utilising the `music21` ontology.

Datapoint representation

In the datapoint representation of a score, each note is a vector in the time–pitch plane containing a basic set of values concerning score–wise onset, duration, and pitch (*see* Appendix C.2.1). Utilising the note as the basic unit of information, the variety of musical contexts and entities of the music–

[3]https://web.mit.edu/music21/

Table D.1: The keys of a datapoint dictionary–entry with sample values.

Feature	Type	Example value
index	integer	0
beatStrength	float	1.0
ontime	float	0.0
beat	float	1.0
measureIncrementalNumber	integer	1
globalOntime	float	0.0
measureRealNumber	integer	1
measureDuration	float	4.0
duration	float	2.0
measureOntime	float	0.0
measureTimeSignature	time signature	4/4
pitchAbsolute	string	C4
midiPitch	integer	60

theoretic ontology of a score may be expressed as qualities stored in the feature–set of a datapoint. This study utilised representations for contextualised information (e.g., scale–degree, metric–strength, harmonic–context), and association with other datapoints (e.g., chord–group and melodic–movement–group). The listing of the basic values stored in a datapoint is shown in Table D.1.

Stable–Pitch–Temporal–Segments (`SPTS`, also 'minimal segments')

. The Stable–Pitch–Temporal–Segments (`SPTS`) representation is a method to handle polyphonic music as a sequence of non-overlapping temporal blocks that may also contain note–simultaneities. The `SPTS` representation simplifies the complexity of polyphonic textures and provides a low–level representation with information regarding the vertical properties on the music notation. The `comutils` library extracts this representation with the `music21.chordify()` function, and enriches it with processes similar to those applied in datapoint

Table D.2: The keys of a SPTS dictionary–entry with sample values.

Feature	Type	Example value
index	integer	0
beatStrength	float	1.0
ontime	float	0.0
beat	float	1.0
measureIncrementalNumber	integer	1
globalOntime	float	0.0
measureRealNumber	integer	1
measureDuration	float	4.0
datapoints	list(string)	[‘D5’, ‘D4’]
duration	float	2.0
measureOntime	float	0.0
measureTimeSignature	time signature	4/4

representations. The listing of the basic values stored in a `SPTS` element is shown in Table D.2.

With the conversion of the music notation of a score into datapoints and `SPTS` elements, the music information may be processed, also enabling the use of existing algorithms that utilise such structures.

D.1.2 Identification of basic music–theoretic elements

Utilising the ‘operational’ symbolic music representations of datapoints and `SPTS` elements (*see* Section D.1.1) the `comutils` library implements a set of music–theoretic analytical operations for the identification/extraction of musical contexts, entities and features. In addition to notated information available from the MusicXML encoding, information regarding rhythmic and tonality contexts is extracted utilising task–specific algorithms.

Rhythm

The rhythmic context of a score is explicitly defined through time–signatures and the consequent grouping of notes into measures. Within such a rhythmic context, each element of the operational representations is assigned a metric–strength quality, depending on the position within the measure in which these appear. Since metric–strength is extracted by considering the properties of each measure separately, changes in time–signatures (if any) are also included.

Tonality

The tonality of a score is represented by a single key–signature, and is calculated with a combination of encoded (tonality) information and probe–tone profiles (Krumhansl, 1990). The MusicXML file encoding provides a 'key' attribute that counts the accidentals of the scale as steps in the circle of fifths. However, there is no distinction regarding the mode. For example, the tonalities of C Major and A Minor have the same 'key' attribute with the value of zero (0). Similarly, a 'key' value of -3 represents an E–flat Major or a C Minor tonality, and a 'key' value of 5, the tonalities of B Major, or G–sharp Minor. To overcome this issue, the method of probe–tone profiles (Krumhansl, 1990) was utilised. This algorithm extracts the correlation between tone–profiles extracted from empirical studies with pitch–class vectors containing the total duration for each of the 12 pitch–classes within a score. This method yields the values of the 'root' of the tonality, representing the tonal of the scale, and the 'mode' of the tonality, concerning the two modes of 'major' and 'minor'. For example, the outcome of this method for the score example in Figure 2.1 is:

```
(0.9008148242382208,  'major', 0)
(0.6772131178041968,  'minor', 9)
(0.5732457972425041,  'major', 7)
(0.5295699269764087,  'major', 5)
(0.48711820754336976, 'minor', 0)
...
```

Extraction of outer voices

As an initial step towards schema–voice extraction, the `comutils` library provides a method that extracts the outer pitch contour of a score. The method selects the two datapoints of each `SPTS` element that have the greatest pitch–interval, and allocates them into two lists, each representing an outer voice. Voice continuity is achieved by considering the minimal pitch–interval between the datapoints in the lists and each new pair of outer datapoints. In the case where a single datapoint is present, it is added in both lists.

Harmonic segmentation

The `comutils` library extracts harmonic segments utilising the *HarmAn* algorithm (Pardo and Birmingham, 2002). The *HarmAn* algorithm uses a dictionary of harmonic qualities (e.g., E major) for pitch–profiles (e.g., '4, 8, 11'). The algorithm identifies harmonic segments by aggregating `SPTS` elements and comparing the pitch–properties of the merged segment with those from the predefined harmonic templates. The merging/segmentation of 'minimal segments' is controlled by a scoring mechanism and a threshold value.

D.2 The Adaptive Expert System

The Adaptive Expert System (`AES`, Katsiavalos et al., 2019) is a collection of classes that computationally implement the operations of the cognitive model described in Chapter 3. The first part of this section presents the architecture of the `AES` system, its classes, and their processes (*see* Section D.2.1). The second part outlines the processes involved on each of the high–level operations (*see* Section D.2.2).

D.2.1 Architecture

Simulating the examined cognitive model, the `AES` system comprises of three classes:

- The Music Data (`AES.MD`);
- The Galant Musical Schemata Classification (`AES.GMSC`), and
- The Schematic Analysis (`AES.SA`).

D.2.1.1 Managing music data and annotations

The Music Data class (`AES.MD`) initialises with datasets of musicXML files and annotations. The main functionality of this class is to programmatically control the music–data input scenarios on the high–level operations of search, classification and discovery of musical schemata. The `AES.MD` utilises the `comutils` library to provide operational representations of music notations to the model.

D.2.1.2 The Galant Musical Schemata Classification

The Galant Musical Schemata Classification class (`AES.GMSC`) simulates the long–term memory schemata of the examined model (*see* Section 3.4), implementing all of its structures and operations. The main components of the `AES.GMSC` class are:

- A collection of Galant family–class objects (`AES.GMSC`, *see* Section 3.2.1);
- The Training Integration–Routine (`GMSC.IR.T`),
- The Exemplar–Recognition–Routine (`ERR`), and
- The Discovery Integration–Routine (`GMSC.IR.D`).

As described in Section 3.4, an `AES.GMSC` object stores Galant family–class objects (`AES.GMSC.SF`). A Galant family–class object (`AES.GMSC.SF`) has at least one Galant family–class archetype object (`AES.GMSC.SF.AR`) with the following attributes:

- An exemplar–base;
- An archetype, and
- A class–similarity function.

Variants with structure–related differences are represented within the same Galant family–class (`AES.GMSC.SF`) but with their own archetype representation as above (e.g., `AES.GMSC.SF.V`). The *exemplar* is the basic unit of musical information in the `AES` system, and the composite representation of a Galant musical schema is based on it.

The training Integration–Routine

The Training Integration–Routine (`GMSC.IR.T`, *see* Algorithm 1) is utilised by the `XL` operation for the creation or update of a Galant family–class from annotated `SA`–segments. The `GMSC.IR.T` examines whether the annotated family–class exist in the `GMSC`, and performs one of the four learning modes (i.e., accretion, tuning, restructuring, and accommodation) to integrate the new information into the corresponding family–class. The `GMSC.IR.T` inputs an annotated `SA`–segment, and the class-similarity function of the annotated class (if any), and applies the Exemplar–Difference–Function (`EDF`) to calculate the Exemplar–Difference–Vectors (`EDV`) between them. The resulting Exemplar–Difference–Vectors (`EDV`) are categorised depending on their structure– and content–related differences, and a corresponding learning mode is applied. Therefore, the information of the training exemplar is integrated to the annotated family–class. The information steps of the `GMSC.IR.T` are described in the music–analytical thought pattern of the `XL` operation (*see* Section 3.4).

The Exemplar–Recognition–Routine

The Exemplar–Recognition–Routine (`ERR`, *see* Algorithm 2) is activated in the second stage of the `SD` operation, after the completion of the `SA.ISED` process. The `ERR` function inputs the repositories of the model (i.e., in the `GMSC` and `SA`), and performs the Exemplar–Discovery–Process (`EDP`), a thresholded Exemplar–Difference–Function (`EDF`) according to the Discovery–Matching–Threshold (`DMT`).

The Exemplar–Discovery–Process (`EDP`) is an extension of the Exemplar–Recognition–Process (`ERP`), approximating the similarity of two exemplars with the Discovery–Matching–Threshold (`DMT`). Whenever a pair of similar, according to the `EDP`, exemplars are found, the discovery Integration–Routine

Algorithm 1: The Training Integration–Routine

```
Input: A training excerpt (TE, in SA.SS form) with a family–class
       annotation (AFC)
Output: The creation/update of the annotated family–class (in the
        GMSC.SF)
Function GMSC.IR.T(TE, AFC):
    if AFC NOT in GMSC.SF then
        /* Perform accommodation and create a new
           family-class                                          */
        GMSC.SF.add(TE, AFC)
    else
        /* Call EDF for each exemplar in the exemplar-base of
           the AFC.                                              */
        EDVs = []
        for Exemplar in GMSC.SF.AFC.EB do
            EDVs.append(EDF(Exemplar, TE))
        /* Considering the minimal EDV from the EDVs list,
           examine the differences                               */
        if min(EDVs) has structure–related differences then
            /* Perform restructuring and create a new variant
               in the GMSC.SF                                    */
            GMSC.SF.AFC.add(TE)
        else if min(EDV) has content–related differences then
            /* Perform tuning by adding TE to the exemplar-base
               and update the class-similarity function          */
            GMSC.SF.AFC.EB.append(TE)
            GMSC.SF.CS.update()
        else
            /* Perform accretion and add TE to the
               GMSC.SF.AFC.exemplar-base                         */
            GMSC.SF.AFC.EB.append(TE)
        return ER
```

(GMSC.IR.D) is activated to integrate the discovered family–class.The pairs of repositories are considered only once (i.e., 10 pairs), as the EDP process calculates the minimum distance between two schema–events. When all exemplars among repositories are compared, the 'uncategorised' repository of the `SA` (`SA.UN`) is merged with the 'uncategorised' repository of the `GMSC` (`GMSC.UN`).

Algorithm 2: The Exemplar–Repository Recognition process (`ERR`)

Input: The repositories of the model (`R`), i.e., `GMSC.SF`, `GMSC.UN`, `SA.CLE` and `SA.UN`)

Output: An updated `GMSC`

```
for R1 in R do
    for R2 in R do
        for Ex1 in R1 do
            for Ex2 in R2 do
                if EDP(Ex1, Ex2) then
                    GMSC.IR.D(Ex1, Ex2)
```

The discovery Integration Routine

The discovery Integration–Routine (`GMSC.IR.D`, *see* Algorithm 3) is activated through the Exemplar–Repository–Recognition process (`ERR`), whenever an exemplar–pair is considered similar, according to the Exemplar–Discovery–Process (`EDP`) and the Discovery–Matching–Threshold (`DMT`). The `GMSC.IR.D` extends the training Integration Routine (`GMSC.IR.T` process, *see* Section 3.4), and instead of forcing the fitting of the information of a training exemplar into the annotated family–class, it attempts to integrate the discovered exemplar–pair into the most similar class in the `GMSC`. In an attempt to regulate the aggregation of discovered exemplar–pairs into the `GMSC`, the `GMSC.IR.D` involves the process of *equilibrium* (`GMSC.EQ`). The `GMSC.EQ` prohibits the integration of discovered exemplar–pairs that pass certain thresholds on the qualities of *variability* and *diversity* of the family–classes in the

GMSC.

Algorithm 3: The discovery Integration Routine of the Galant musical schemata classification

```
Input: A discovered pair of similar exemplars (DP)
Output: A learning mode to integrate the DP
GMSC.EQ threshold values   Function GMSC.IR.D(DP):
    /* Calculate the minimum EDVs between the DP and the
       family-classes in the GMSC                                   */
    mEDVs = []
    for SF in GMSC.SF do
        mEDVs.append(min(EDF(DP, SF.AR.EB)))
    /* Sort the mEDVs list and examine how the discovered
       exemplar-pair may be integrated into the GMSC                */
    for mEDV in mEDVs.sorted() do
        /* Check if the discovered exemplar-pair and the
           closest family-class are within the discovery
           similarity threshold                                     */
        if mEDV < Discovery–Matching–Threshold then
            /* Check the variability threshold                      */
            if GMSC.add(DP, mEDV).variability() < GMSC.EQ.variability
             then
                /* Check the diversity threshold                    */
                if GMSC.add(DP, mEDV).diversity() > GMSC.EQ.diversity
                 then
                    /* Allow adding the closest family-class in
                       the GMSC                                     */
                    GMSC.add(DP, mEDV)
        else
            /* Check the diversity threshold                        */
            if GMSC.add(DP, mEDV).diversity() > GMSC.EQ.diversity then
                /* Allow adding the closest family-class in the
                   GMSC                                             */
                GMSC.add(DP, mEDV)
            else
                Increase variability of the closest family–class to include
                 the DP.
```

D.2.1.3 The Schematic Analysis class

The `AES` implements the operation of schematic analysis with the `AES.SA` class. This class initiates with a configuration concerning the `SA` parameters and processes the notated music information of a score with the goal of extracting the schematic analysis form, a sequence of schema–events (*see* Section 3.3). The three main stages involved in the `SA` operation include (*see* Algorithm 4):

- Performing score analysis;
- Extracting schematic elements, and
- Extracting the schematic analysis form.

The discovery operation (`SD`) includes an additional processing layer with the Intra–Score Exemplar–Discovery process (`SA.ISED`, *see* Algorithm 5).

D.2.2 Performing high–level operations

D.2.2.1 Galant family–type search

The Galant musical schemata search task (*see* Chapter 4) identifies Galant family–classes in the schematic analysis form of a score. The Exemplar–Identification process (`EIP`, *see* Algorithm 6) compares every schema–event in the schematic analysis of a score with the schema–events of the search targets (Galant family–classes) in order to create an inverted index; each schema–event in the schematic analysis form is tagged with the targeted family–classes it appears (if any), and the position of the schema–event it matches with. After examining all of the schema–events in the schematic analysis form of a score, a temporal threshold is applied to bound those schema–events of the same family–class and correct order into possible complete instances.

Algorithm 4: Performing the schematic analysis operation

```
Input: A score (MS) and the SA configuration (SAC)
Output: The schematic analysis form of a score
Function ScoreAnalysis(MS, SAC):
    ScA.Tonality = SA.getTonality(MS, SAC)
    ScA.Metric = SA.getMetric(MS, SAC)
    ScA.OuterVoices = SA.getOuterVoices(MS ,SAC)
    ScA.Harmony = SA.getHarmony(MS, SAC)
    ScA.Features = SA.getFeatures(MS, SAC)
    return ScA

Function SchematicAnalysis(MS, SAC):
    SA.SchemaEvents = SA.getSchemaEvents(MS, ScA, SAC)
    SA.SchemaVoices = SA.getSchemaVoices(MS, ScA, SAC)
    SA.SchemaVoicePairs= SA.getSchemaVoicePairs(MS, ScA, SAC)
    SA.EventProgressions = SA.getEventProgressions(MS, ScA, SAC)
    SA.SchematicAnalysis = SA.getSchematicAnalysis(MS, ScA, SAC)
    return SA.SchematicAnalysis

Function SchematicAnalysisOperation(MS,):
    return SA(ScA(MS, SAC))
```

D.2.2.2 Galant family–type classification

The task of Galant archetype classification (*see* Chapter 5) examined the operation of example–based learning (*see* Section 3.4). The classification of Galant family–classes is created and updated through the training Integration–Routine (GMSC.IR.T). The trained Galant family–classes are identified in the schematic analysis of a score through the Exemplar–Recognition process (ERP), an extension of the exact matching of the Exemplar–Identification process (EIP). Training and testing of the classification may occur in the same session (*see* Algorithm 7).

Algorithm 5: The Intra–Score Exemplar–Discovery process (`SA.ISED`)

Input: Score–wise schematic analysis form (`SA.SS`), and the schematic analysis configuration (`SAC`)

Output: Score–wise Exemplar classification (`SA.CLE`)

```
Function ExemplarDiscoverySpace(SS.SA, SAC):
    EDS = SA.generateExemplars(SAC)
    return EDS
Function ExemplarRepetitions(EDS, SAC):
    ER = SA.identifyRepetitions(SAC)
    return ER
Function ExemplarRecognitions(EDS, SAC):
    EDC = SA.identifyDiscoveredClasses(SAC)
    return EDC
Function IntraScoreExemplarDiscovery(SA.SS, SAC):
    EDS = ExemplarDiscoverySpace(SA.SS,SAC)
    ER = ExemplarRepetitions(EDS,SAC)
    CLE = ExemplarClassification(SA.SS,SAC)
    return CLE
```

Algorithm 6: The Galant archetype exemplar identification process

Input: Schematic Analysis form (`SA.SS`)

Output: Labelled Exemplars

```
for SSE in SA.SS do
    for SF in GMSC.SF do
        for V in SF do
            for VE in V.Exemplar-base do
                for VESE in VE do
                    if (SSE.melody == VESE.melody) and
                    (SSE.melody == VESE.melody) and
                    (SSE.melody == VESE.melody) then
                        Tag SSE with the family–class and the
                         schema–event position.
```

Algorithm 7: The Galant family–type classification operation (`SD`)

Input: A dataset of annotated scores, `MD.[MS]`
Output: Creates/updates the `GMSC` classification and performs classification

```
for MS in MD do
    SA(MS)
    if annotations are found then
        GMSC.IR.T(annotated segment)
    else
        Perform ERP
```

D.2.2.3 Galant family–type discovery

The operation of Galant family–class discovery (*see* Section 3.5) is performed in two stages with the following processes:

- The Intra–Score Exemplar–Discovery process (`SA.ISED`), and
- The Exemplar–Repository–Recognition process (`ERR`).

The `SA.ISED` process is performed whenever a new score inputs the model. The outcome of the `SA.ISED` process (i.e., a segmentation of the score) is added to the `ERR` process. After the completion of the `ERR` process, the repositories merge and the next score element is examined (*see* Algorithm 8).

Algorithm 8: The top–function of the Galant family–class discovery operation (`SD`)

Input: A dataset of scores, `MD.[MS]`
Output: Creates/updates the `GMSC` classification

```
for MS in MD do
    ERR(SA.ISED(MS))
```

E Results of computational experiments in tabular format

This chapter presents tabular data for all of the results graphs in the computational experiments of this document (in Sections E.1 and E.2).

E.1 Search results

Table E.1: Search task computational experiments' results in tabular form.

	Harmonic							Schematic										
	Hnnn	HFFn	HFFS	HFFT	HFFE	HAAT	HAAE	SFFn	SFAn	SAFn	SAAn	SFFS	SFFT	SFFE	SAAS	SAAT	SAAE
Romanesca	0.985	0.956	0.947	0.960	0.958	0.951	0.938	0.937	0.934	0.931	0.921	0.896	0.934	0.933	0.884	0.938	0.914
Meyer	0.925	0.914	0.767	0.910	0.908	0.906	0.900	0.880	0.876	0.877	0.870	0.760	0.867	0.861	0.749	0.858	0.856
(Galant)	0.936	0.926	0.770	0.927	0.925	0.923	0.918	0.899	0.893	0.893	0.887	0.765	0.878	0.869	0.752	0.865	0.862
(Sonatas)	0.915	0.903	0.765	0.893	0.892	0.889	0.882	0.862	0.860	0.861	0.854	0.755	0.857	0.854	0.746	0.851	0.850
Do–Re–Mi	0.975	0.965	0.883	0.954	0.954	0.951	0.950	0.938	0.933	0.927	0.925	0.877	0.918	0.913	0.870	0.908	0.907
Fenaroli	0.946	0.934	0.924	0.924	0.923	0.894	0.890	0.884	0.886	0.883	0.877	0.862	0.910	0.916	0.914	0.910	0.903
Prinner	0.954	0.947	0.908	0.940	0.932	0.930	0.921	0.909	0.907	0.904	0.899	0.879	0.897	0.896	0.879	0.894	0.893
(Galant)	0.964	0.955	0.915	0.946	0.938	0.934	0.919	0.915	0.913	0.911	0.905	0.886	0.901	0.900	0.885	0.897	0.896
(Sonatas)	0.945	0.939	0.901	0.935	0.927	0.926	0.923	0.903	0.902	0.898	0.893	0.873	0.893	0.892	0.873	0.892	0.891
Quiescenza	0.794	0.789	0.754	0.779	0.774	0.768	0.763	0.675	0.668	0.661	0.658	0.628	0.675	0.669	0.637	0.666	0.659
Clausula	0.959	0.956	0.841	0.949	0.932	0.914	0.910	0.899	0.892	0.887	0.839	0.809	0.881	0.879	0.804	0.884	0.878
(Galant)	0.965	0.962	0.847	0.953	0.926	0.919	0.915	0.910	0.909	0.906	0.825	0.825	0.904	0.902	0.820	0.897	0.895
(Sonatas)	0.954	0.951	0.836	0.945	0.938	0.909	0.905	0.889	0.875	0.869	0.853	0.794	0.859	0.857	0.788	0.871	0.862

E.2 Classification results

Table E.2: Classification results for group `A` configurations.

		Meyer		Prinner		Clausula	
		P	R	P	R	P	R
`A1`	GS	0.882	0.903	0.926	0.935	0.873	0.943
	SN	0.872	0.893	0.901	0.911	0.852	0.924
`A2`	GS	0.831	0.926	0.893	0.956	0.804	0.968
	SN	0.803	0.913	0.883	0.932	0.791	0.936
`A3`	GS	0.894	0.890	0.931	0.919	0.888	0.939
	SN	0.876	0.885	0.913	0.905	0.872	0.918
`A4`	GS	0.841	0.912	0.898	0.933	0.813	0.958
	SN	0.823	0.899	0.891	0.917	0.805	0.931

Table E.3: Classification results for group `B` configurations.

		Meyer		Prinner		Clausula	
		P	R	P	R	P	R
`B1`	GS	0.854	0.893	0.906	0.904	0.836	0.939
	SN	0.848	0.879	0.879	0.901	0.817	0.903
`B2`	GS	0.798	0.904	0.869	0.937	0.744	0.951
	SN	0.779	0.893	0.854	0.910	0.725	0.928
`B3`	GS	0.861	0.884	0.916	0.899	0.847	0.927
	SN	0.852	0.866	0.895	0.896	0.828	0.897
`B4`	GS	0.806	0.899	0.885	0.924	0.767	0.945
	SN	0.787	0.887	0.868	0.904	0.748	0.925

Table E.4: Classification results for group `C` configurations.

		Meyer		Prinner		Clausula	
		P	R	P	R	P	R
`C1`	GS	0.822	0.868	0.884	0.886	0.822	0.938
	SN	0.816	0.860	0.863	0.865	0.808	0.903
`C2`	GS	0.787	0.899	0.845	0.913	0.736	0.949
	SN	0.766	0.882	0.844	0.894	0.718	0.930
`C3`	GS	0.852	0.861	0.901	0.863	0.840	0.935
	SN	0.835	0.852	0.874	0.854	0.827	0.895
`C4`	GS	0.796	0.895	0.885	0.905	0.753	0.941
	SN	0.780	0.880	0.860	0.902	0.731	0.921

Table E.5: Classification results for group `D` configurations.

		Meyer		Prinner		Clausula	
		P	R	P	R	P	R
`D1`	GS	0.781	0.875	0.863	0.862	0.805	0.925
	SN	0.773	0.854	0.844	0.843	0.796	0.894
`D2`	GS	0.766	0.894	0.834	0.912	0.732	0.943
	SN	0.752	0.883	0.823	0.902	0.715	0.923
`D3`	GS	0.813	0.869	0.897	0.867	0.834	0.925
	SN	0.805	0.846	0.864	0.844	0.824	0.890
`D4`	GS	0.781	0.890	0.884	0.907	0.754	0.933
	SN	0.775	0.879	0.854	0.900	0.738	0.919

List of acronyms

AES.SA.SA The Schematic Analysis stage of the Schematic Analysis class of the Adaptive Expert System. 110

AES.SA.ScA The Score Analysis stage of the Schematic Analysis class of the Adaptive Expert System. 110

AES.SA The Schematic Analysis class of the Adaptive Expert System. 110, 157

AES The Adaptive Expert System. The computational implementation of the proposed cognitive model. xi, 44, 46, 47, 107, 109–111, 113, 115, 117, 118, 131, 133, 155, 207, 213

DMT The Discovery–Matching–Threshold. The maximal difference between two exemplars for them to be considered of the same family–class. 97, 98, 100, 156–158, 164, 176, 215, 217, 227

EDF The Exemplar Difference–Function. A similarity function that inputs two exemplars and outputs their Exemplar–Difference–Vector (EDV). 90, 101, 215, 216

EDP The Exemplar–Discovery–Process. A function that decides whether two exemplars are of the same class, considering the Discovery–Matching–Threshold (DMT). 156, 158, 159, 215–217

EDV The Exemplar Difference–Vector. The structure– and content– related differences between two exemplars. 59, 90, 98–102, 158, 159, 164, 215, 227

EIP The Exemplar–Identification–Process. The exact matching of the pitch–related properties between two exemplars. 219, 220

ERP The Exemplar–Recognition–Process. The approximate matching of the pitch–related properties between two exemplars. 156, 215, 220, 222

ERR The Exemplar–Recognition–Routine. Inputs two repositories and returns a matrix with the Exemplar–Difference–Vectors (`edv`) of their elements. 95, 96, 100, 101, 103, 105, 155–159, 214, 215, 217, 222

GMSC.EQ The `GMSC` process of equilibrium, thresholding minimum diversity among discovered family–classes, and maximum variability among variants of the same family–class. xv, 100, 103, 105, 157, 159, 160, 164, 167, 169, 171, 217, 218

GMSC.IR.D The Discovery `GMSC` Integration–Routine, integrating discovered exemplar–pairs into family–classes in the `GMSC`. 100, 101, 105, 157, 159, 214, 216, 217

GMSC.IR.T The Training `GMSC` Integration–Routine, integrating annotated `SA`–segments into family–classes in the `GMSC`. 214, 215, 217, 220, 222

GMSC.SF.AR.CS The Class–Similarity Function of a `GMSC.SF.AR`. 53, 61

GMSC.SF.AR.EB The Exemplar–Base of a `GMSC.SF.AR`. 53

GMSC.SF.AR The Archetype of a `GMSC.SF`, an Exemplar. 53, 54

GMSC.SF.CS The Class–Similarity function of a Galant family–class in the `GMSC`. 216

GMSC.SF A Galant Musical Schemata Family–Class in the `GMSC`. 51, 53, 96

GMSC The Galant Musical Schemata Classification system that models the LTM. 94, 96, 100, 155–157, 215, 217, 218, 222

GMST The Galant Musical Schemata Theory. ix, 1–3, 5, 6, 11–13, 17, 18, 28, 30, 32, 33, 38, 40, 47, 49, 50, 173

SA.AP.GMSC The information–state of the GMSC. 62

SA.AP.SW The Score–Wise information–state of the model. 62

SA.AP.TW The Time–Window information–state. 62

SA.AP The Active Plane of the SA. 61, 66

SA.CLE The Classification of Exemplars, the output of the SA.ISED process. 95, 97

SA.ISED The SA Intra–Score Exemplar Discovery process. 95, 97–100, 104, 105, 155–157, 160, 171, 215, 219, 221, 222

SA.SS.INF The Inference method for the extraction of the schema–event progression. 68

SA.SS The output of the SA operation, the Sequence of Schema–events. 95, 97, 155

SA The operation of Schematic Analysis. 33, 39, 40, 50, 94, 95, 110, 164, 173, 175, 176, 215, 222

SD The operation of musical Schemata Discovery. xv, 39, 40, 94, 101, 103, 155, 159, 161–163, 170–173, 175, 176, 215, 219, 222

SPTS The Stable–Pitch Temporal–Segments. Also ‘minimal segments’ or ‘simultaneities’.. 208–210, 212

XL The operation of Galant archetype example–based learning. 37, 40, 94, 129, 173–175

comutils A library in Python for low–level operations with symbolic music

References

Ariza, C. and Cuthbert, M. S. (2011), The music21 stream: A new object model for representing, filtering, and transforming symbolic musical structures, *in* 'International Computer Music Conference (ICMC)', Vol. 2011, pp. 61–68.

Atkinson, R. C. and Shiffrin, R. M. (1968), 'Human memory: A proposed system and its control processes', *Psychology of learning and motivation* **2**, 89–195.

Atkinson, R. C. and Shiffrin, R. M. (1971), *The control processes of short-term memory*, Stanford University.

Bartlett, F. C. and Bartlett, F. C. (1932), *Remembering: A study in experimental and social psychology*, Cambridge University Press.

Bregman, A. S. (1994), *Auditory scene analysis: The perceptual organization of sound*, MIT press.

Byrd, D. and Crawford, T. (2002), 'Problems of music information retrieval in the real world', *Information processing & management* **38**(2), 249–272.

Cambouropoulos, E. (2008), 'Voice and stream: Perceptual and computational modeling of voice separation', *Music Perception* **26**(1), 75–94.

Cambouropoulos, E. (2010), 'The musical surface: Challenging basic assumptions', *Musicae Scientiae* **14**(2_suppl), 131–147.

Cambouropoulos, E. (2012), A directional interval class representation of chord transitions, *in* 'Proceedings of the 12th International Conference for Music Perception and Cognition & 8th Conference of the European Society for the Cognitive Sciences of Music', Thessaloniki, Greece, p. 77.

Cannam, C., Landone, C. and Sandler, M. (2010), Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files, *in* 'Proceedings of the ACM Multimedia 2010 International Conference', Firenze, Italy, pp. 1467–1468.

Collins, T., Arzt, A., Frostel, H. and Widmer, G. (2016), Using Geometric Symbolic Fingerprinting to Discover Distinctive Patterns in Polyphonic Music Corpora, *in* 'Computational Music Analysis', Springer, pp. 445–474.

Collins, T., Thurlow, J., Laney, R., Willis, A. and Garthwaite, P. H. (2010), A comparative evaluation of algorithms for discovering translational patterns in baroque keyboard works, *in* J. S. Downie and R. Veltkamp, eds, 'Proceedings of the International Symposium on Music Information Retrieval', International Society for Music Information Retrieval, Utrecht, The Netherlands, pp. 3–8.

Conklin, D. (2002), Representation and discovery of vertical patterns in music, *in* 'Music and Artificial Intelligence', Springer-Verlag, pp. 32–42.

Conklin, D. and Anagnostopoulou, C. (2001), Representation and discovery of multiple viewpoint patterns, *in* 'International Computer Music Association', Havana, Cuba, pp. 479–485.

Conklin, D. and Anagnostopoulou, C. (2006), 'Segmental pattern discovery in music', *INFORMS Journal on computing* **18**(3), 285–293.

Conklin, D. and Bergeron, M. (2008), 'Feature set patterns in music', *Computer Music Journal* **32**(1), 60–70.

Conklin, D. and Bergeron, M. (2010), Discovery of contrapuntal patterns, *in* 'Proceedings of the Eleventh International Society for Music Information Retrieval Conference (ISMIR)', Vol. 11, pp. 201–206.

Deliège, I., Mélen, M., Stammers, D. and Cross, I. (1996), 'Musical schemata in real-time listening to a piece of music', *Music Perception: An Interdisciplinary Journal* **14**(2), 117–159.

Deutsch, D. (1975), 'The organization of short-term memory for a single acoustic attribute', *Short-term memory* pp. 107–151.

Deutsch, D. (1999), Grouping mechanisms in music, *in* 'The psychology of music', Elsevier, pp. 299–348.

Deutsch, D. (2013), *Psychology of music*, Elsevier.

Evans, S. H. (1967), ‘A brief statement of schema theory’, *Psychonomic Science* **8**(2), 87–88.

Forte, A. and Gilbert, S. E. (1982), *Introduction to Schenkerian analysis*, Norton.

Foubert, K., Collins, T. and De Backer, J. (2017), ‘Impaired maintenance of interpersonal synchronization in musical improvisations of patients with borderline personality disorder’, *Frontiers in Psychology* **8**.

Friedman, J., Hastie, T., Tibshirani, R. et al. (2001), *The elements of statistical learning*, Vol. 1, Springer series in statistics New York.

Gjerdingen, R. (2007), *Music in the galant style*, Oxford University Press, USA.

Good, M. (2001), ‘Musicxml for notation and analysis’, *The virtual score: representation, retrieval, restoration* **12**(113-124), 160.

Haugeland, J. (1989), *Artificial intelligence: The very idea*, MIT press.

Hirata, K. and Matsuda, S. (2003), ‘Interactive music summarization based on generative theory of tonal music’, *Journal of New Music Research* **32**(2), 165–177.

Honing, H. (1993), ‘Issues on the representation of time and structure in music’, *Contemporary music review* **9**(1-2), 221–238.

Huron, D. (2001), ‘Tone and voice: A derivation of the rules of voice-leading from perceptual principles’, *Music Perception: An Interdisciplinary Journal* **19**(1), 1–64.

Huron, D. (2002), ‘Music information processing using the humdrum toolkit: Concepts, examples, and lessons’, *Computer Music Journal* **26**(2), 11–26.

Katsiavalos, A. and Cambouropoulos, E. (2012), An Interactive Computational System for the Exploration of Music Voice/Stream Segregation Processes, *in* ‘Proceedings of the 12th International Conference for Music Perception and Cognition & 8th Conference of the European Society for the Cognitive Sciences of Music’, Thessaloniki, Greece, p. 157.

Katsiavalos, A., Collins, T. and Battey, B. (2019), An initial computational model for musical schemata theory., *in* ‘Proceedings of the Twentieth International Conference on Music Information Retrieval (ISMIR)’, pp. 166–172.

Köhler, W. (1967), 'Gestalt psychology', *Psychologische Forschung* **31**(1), XVIII–XXX.

Krumhansl, C. (1997), Effects of perceptual organization and musical form on melodic expectancies, *in* 'Music, gestalt, and computing', Springer, pp. 294–320.

Krumhansl, C. L. (1990), *Cognitive foundations of musical pitch*, Vol. 17, Oxford University Press, New York.

Lartillot, O., Toiviainen, P. and Eerola, T. (2008), A matlab toolbox for music information retrieval, *in* 'Data analysis, machine learning and applications', Springer, pp. 261–268.

Lee, W. and Chen, A. L. (1999), Efficient multifeature index structures for music data retrieval, *in* 'Electronic Imaging', International Society for Optics and Photonics, pp. 177–188.

Lerdahl, F. and Jackendoff, R. (1985), *A generative theory of tonal music*, MIT press.

Lewin, D. (2007), *Generalized musical intervals and transformations*, Oxford University Press.

Marsden, A. (2010), 'Schenkerian analysis by computer: A proof of concept', *Journal of New Music Research* **39**(3), 269–289.

McKay, C. and Fujinaga, I. (2006), jsymbolic: A feature extractor for midi files, *in* 'Proceedings of the International Computer Music Conference', pp. 302–5.

McKay, C. and Fujinaga, I. (2009), *jMIR: Tools for automatic music classification*, Citeseer.

Meredith, D., Lemström, K. and Wiggins, G. A. (2002), 'Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music', *Journal of New Music Research* **31**(4), 321–345.

Mouton, R. and Pachet, F. (1995), The symbolic vs. numeric controversy in automatic analysis of music, *in* 'Artificial Intelligence and Music (IJCAI—95 Workshop Program Working Notes)', pp. 32–40.

Narmour, E. (1991), 'The top-down and bottom-up systems of musical implication: Building on meyer's theory of emotional syntax', *Music Perception: An Interdisciplinary Journal* **9**(1), 1–26.

Nattiez, J.-J. (1975), 'Fondements d'une sémiologie de la musique', *Union Generale d'Editions* .

Pardo, B. and Birmingham, W. P. (2002), 'Algorithms for chordal analysis', *Computer Music Journal* **26**(2), 27–49.

Pearce, M. T. (2002), 'Notes on charm - a specification for the representation of musical knowledge'.
URL: *http://webprojects.eecs.qmul.ac.uk/marcusp/index.php?page=3*

Piaget, J. and Cook, M. (1952), *The origins of intelligence in children*, Vol. 8, International Universities Press New York.

Pickens, J. (2001), A survey of feature selection techniques for music information retrieval, *in* 'Proceedings of the 2nd International Symposium on Music Information Retrieval (ISMIR)', Vol. 124.

Povel, D.-J. and Essens, P. (1985), 'Perception of temporal patterns', *Music Perception: An Interdisciplinary Journal* **2**(4), 411–440.

Purwins, H., Herrera, P., Grachten, M., Hazan, A., Marxer, R. and Serra, X. (2008), 'Computational models of music perception and cognition i: The perceptual and cognitive processing chain', *Physics of Life Reviews* **5**(3), 151–168.

Rumelhart, D. E. (1978), *Schemata: The building blocks of cognition*, Center for Human Information Processing, University of California, San Diego, CA, USA.

Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1985), Learning internal representations by error propagation, Technical report.

Rumelhart, D. E. and Norman, D. A. (1976), Accretion, tuning and restructuring: Three modes of learning, Technical report.

Ruwet, N. (1972), *Langage, musique, poésie*, editions du seuil edn, Paris.

Ruwet, N. and Everist, M. (1987), 'Methods of analysis in musicology', *Music Analysis* pp. 3–36.

Schedl, M., Gómez, E. and Urbano, J. (2014), 'Music information retrieval: Recent developments and applications', *Foundations and Trends in Information Retrieval* **8**(2-3), 127–261.
URL: *http://dx.doi.org/10.1561/1500000042*

Serra, X., Magas, M., Benetos, E., Chudy, M., Dixon, S., Flexer, A., Gómez, E., Gouyon, F., Herrera, P., Jorda, S., Paytuvi, O., Peeters, G., Schluter, J., Vinet, H. and Widmer, G. (2013), *Roadmap for Music Information ReSearch*, Creative Commons BY-NC-ND 3.0 license.

Shen, J. (2007), *Intelligent Music Information Systems: Tools and Methodologies: Tools and Methodologies*, IGI Global.

Sleator, D. and Temperley, D. (2001), 'The melisma music analyzer', *Available online atwww. link. cs. cmu. edu/music-analysis* .

Smaill, A. and Wiggins, G. (1990), *Hierarchical music representation for composition and analysis*, University of Edinburgh, Department of Artificial Intelligence.

Snyder, B. (2000), *Music and memory: An introduction*, MIT press.

Symons, J. (2012), Temporal regularity as a key to uncovering statistically significant schemas in an eighteenthcentury corpus, *in* 'annual meeting of the Society for Music Theory, New Orleans, LA, November'.

Weyde, T. (2005), Modelling cognitive and analytic musical structures in the musitech framework, *in* 'UCM 2005 5th Conference" Understanding and Creating Music", Caserta', pp. 27–30.

Wiggins, G. A. (2010), 'Cue abstraction, paradigmatic analysis and information dynamics: Towards music analysis by cognitive model', *Musicae Scientiae* **14**(2 suppl), 307–331.

Wissmann, J. (2012), Chord sequence patterns in owl, PhD thesis, City University London.

Witten, I. H. and Conklin, D. (1990), 'Modeling music: Systems, structure, and prediction', *Journal of New Music Research* **19**(1), 53–66.