

The Structure of Free Polyphony

Présentée le 13 février 2023

Collège des humanités
Digital and Cognitive Musicology Lab
Programme doctoral en humanités digitales

pour l'obtention du grade de Docteur ès Sciences

par

Christoph FINKENSIEP

Acceptée sur proposition du jury

Prof. R. West, président du jury
Prof. M. A. Rohrmeier, directeur de thèse
Prof. M. Pearce, rapporteur
Prof. J. Yust, rapporteur
Prof. F. Eisenbrand, rapporteur

Of course it is happening inside your head, Harry, but why on earth should that mean that it is not real?

—Albus Dumbledore

Acknowledgements

A project such as the one presented in this thesis is nothing that a single person can do without help and support from other people.

I would like to thank Martin Rohrmeier for his inspiration, encouragement, and advice, for the opportunity to work on one of the most interesting research topics in the world, and for creating this wonderful environment for scientific and personal growth. I also thank my friends and colleagues from and around the DCML, Daniel Harasim (who in particular contributed invaluable feedback to this thesis), Fabian Moss, Jessica Pidoux, Markus Neuwirth, Robert Lieck, Johannes Hentschel, Leona Wall, Gabriele Cecchetti, Ken Déguernel, Petter Ericson, Steffen Herff, Andrew McLeod, Shuxin Meng, Zeng Ren, Yannis Rammos, Aurélie Nicoulaz, Estelle Quinton, Richard Widdess, and Sebastian Klaßmann for teaching me countless things, for opening my eyes to the many perspectives you can take on music, science, and life, for the many little favours, for enduring bad puns, opinions on programming paradigms, and weeks of internal lectures, and for the simply amazing time I had in Lausanne in the last four years.

I would like to thank my family, Viktoria, Daniel, and Sebastian Finkensiep, for supporting all of my interests from early on and for all the music around me. I thank Steffen Schiel for introducing me to music theory and helping me develop my musical intuitions. I also thank Lotta Ottensmeyer and Arne Kramer-Sunderbrink for many inspiring discussions of music, art, science, philosophy, and life. Finally, I would like to thank Kirsten Heuermann for giving me the space to pursue this path, for enduring my fascination for weird things, for the constant supply of hot chocolates, for reminding me of the important things in life, and for all her patience, consolation, and encouragement.

Thank you all.

Lausanne, October 8, 2022

Christoph Finkensiep

Acknowledgements

Funding Statement

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 760081 – PMSB. The author also thanks the Volkswagen Foundation and Claude Latour for generously supporting this research.

Abstract

The human ability to perceive and understand music is remarkable. From an unstructured stream of acoustic input it creates a wide range of experiences, from psychoacoustic effects to emotional and aesthetic responses. One such set of phenomena is the experience of *structure*, the perception of notes standing in musically meaningful relationships to each other and to abstract entities such as chords, voices, schemata, formal segments, motives, or themes, which are not directly represented in the stream of notes and thus must be inferred.

This dissertation argues that the perception of musical structure from notes is an instance of the general principle of Bayesian perception, which states that perception is probabilistic inference to the latent causes that produce the sensory input. It first explores the fundamental relations between notes and latent entities in three case studies on modal melodies, recognition of voice-leading schemata, and harmonic ornamentation. Subsequently, it proposes a unified generative model of the note-level structure underlying Western tonal music and potentially other styles. This model is based on the elaboration of simple latent note configurations into the musical surface, maintaining vertical, horizontal, and hierarchical relations in the process.

On the music-theoretical side, this model provides a language to formally express analytical intuitions and a foundation for precise definitions of traditional concepts and clarification of their relation to the musical surface. On the computational side, the model demonstrates how complex musical structures can be inferred and how the structural properties of a style can be learned using parsing and probabilistic inference. On the cognitive side, the model shows that the perception of tonal structure can be linked to general Bayesian perception through a generative process. This thesis therefore constitutes a bridge between different perspectives and disciplines, and thus contributes to a unified understanding of the human capacity for music.

Zusammenfassung

Die menschliche Fähigkeit, Musik wahrzunehmen und zu verstehen, ist bemerkenswert. Aus einem ungeordneten Strom akustischer Signale generiert sie eine breite Palette an Erlebnissen, von psycho-akustischen Effekten bis hin zu emotionalen und ästhetischen Erfahrungen. Eines dieser Phänomene ist die Wahrnehmung von *Struktur*, das Hören von Tönen in einer musikalischen Beziehung zu einander und zu abstrakten Entitäten wie Akkorde, Stimmen, Satzmodellen, formalen Abschnitten, Motiven, oder Themen, welche nicht direkt aus dem Strom der Töne ablesbar sind sondern aus ihm inferiert werden müssen.

Diese Dissertation vertritt und erläutert den Standpunkt, dass die Wahrnehmung musikalischer Struktur aus einem Strom von Tönen eine Instanz des allgemeinen Prinzips der bayesschen Wahrnehmung ist, nach dem alle Wahrnehmung das Schließen von sensorischer Information auf die zugrunde liegenden Ursachen nach den Regeln der subjektiven Wahrscheinlichkeit ist. Sie nähert sich zunächst den grundlegenden Beziehungen zwischen Tönen und latenten Entitäten in drei Fallstudien zu modalen Melodien, Erkennung von Satzmodellen und zur Ornamentierung von Harmonien. Im Anschluss wird ein vereinheitlichtes generatives Modell der Tonstruktur, die westlicher tonaler Musik (und möglicherweise auch anderen Stilen) zugrunde liegt, vorgestellt. Dieses Modell beschreibt einen Prozess der Elaboration einfacher, latenter Tonkonfigurationen zur komplexen Oberflächenstruktur von Musikstücken, in dessen Verlauf die vertikalen, horizontalen, und hierarchischen Beziehungen zwischen Tönen hergestellt werden.

Aus musiktheoretischer Sicht bietet dieses Modell eine Sprache, in der analytische Intuitionen formell ausgedrückt werden können, sowie eine Grundlage zur Definition und Schärfung von traditionellen theoretischen Begriffen und deren exakter Beziehung zu den Tönen, aus denen ein Musikstück besteht. Aus mathematischer Sicht wird gezeigt, wie komplexe musikalische Strukturen algorithmisch inferiert und die Struktureigenschaften eines Stils gelernt werden können. Im Bezug auf Kognition zeigt das Modell durch seinen zugrunde liegenden generativen Prozess, wie die Wahrnehmung von tonaler Struktur als Spezialfall von allgemeiner bayesscher Wahrnehmung verstanden werden kann. Damit bildet diese Dissertation eine Brücke zwischen verschiedenen Perspektiven und Disziplinen und trägt somit zu einem ganzheitlichen Verständnis der menschlichen Fähigkeit zur Musik bei.

Contents

Acknowledgements	v
Funding Statement	vi
Abstract (English/Deutsch)	vii
List of Figures	xvii
List of Tables	xxi
List of Algorithms	xxiii
List of Listings	xxv
List of Publications	xxvii
I Foundations	1
1 Introduction	3
1.1 Music and Structure	3
1.1.1 Basic Structural Relations	3
1.1.2 Previous Models of Tonal Structure	5
1.1.3 The Voice Problem	15
1.2 Interpretation	18
1.2.1 Interpretations as Explanations	18
1.2.2 Interpretations in Music Psychology	22
1.2.3 Interpretation and Expectation	24
1.3 Outlook	27
1.3.1 Goals and Methods	27
1.3.2 Thesis Outline	28
2 Model-Based Music Theory	31
2.1 Introduction: Music Theory and Corpus Research	31
2.2 Model-Driven Music Research	32

Contents

2.3	Modelling Decisions in Musical Corpus Studies	35
2.3.1	Entities and Relations	35
2.3.2	Analysis	38
2.3.3	Modelling Decisions as Assumptions	39
2.4	Probabilistic Modelling and Bayesian Inference	39
2.4.1	Expressing Models as Probability Distributions	40
2.4.2	Types of Inference in Bayesian Models	44
2.4.3	Inference Methods	48
2.4.4	Probabilistic Programming	51
2.4.5	Relevance to Music Research	54
2.5	A Recipe for Model-based Corpus Research	56
2.6	Acknowledgments	59
 II Preliminary Studies		61
Interlude 1		63
3 Finding Voice-Leading Schemata		65
3.1	Introduction	65
3.2	Related Work	67
3.3	Dataset	68
3.3.1	Schema Formalization and Lexicon	68
3.3.2	Data Production	69
3.4	Features and schema classification	70
3.4.1	Musical Features	70
3.4.2	Classification and Evaluation Method	73
3.5	Results and Discussion	74
3.5.1	Classification Performance	74
3.5.2	Feature Evaluation	77
3.6	Conclusion	79
3.7	Acknowledgements	79
 4 Chord Types and Ornamentation		81
4.1	Introduction	82
4.2	Method	85
4.2.1	Model	85
4.2.2	Inference	90
4.2.3	Datasets and Preprocessing	92
4.3	Results and Discussion	96
4.3.1	Chord Profiles	97

4.3.2	Degree of Ornamentation	102
4.4	Conclusion	104
5	A Graph Grammar for North Indian Melodies	107
5.1	Introduction	108
5.2	Melodic Operations	109
5.3	Modes and Generalized Neighbors	111
5.4	A Formal Grammar of Rāga Melodies	113
5.4.1	Representing Melodies as Graphs	113
5.4.2	Formal Definition of the Grammar	115
5.5	Discussion	118
5.6	Conclusion	121
5.7	Acknowledgements	121
III	The Protovoice Model	123
	Interlude 2	125
6	The Protovoice Model	129
6.1	Introduction	129
6.2	The Protovoice Model	132
6.2.1	Constructing Protovoices	132
6.2.2	Temporal Organization	133
6.3	A Parsing Algorithm for Protovoices	136
6.3.1	Representing Derivations	136
6.3.2	Parsing	138
6.4	Discussion and Conclusion	140
6.5	Acknowledgements	143
7	Protovoice Theory	145
7.1	Introduction	145
7.2	Motivation	146
7.3	The Protovoice Model	150
7.4	Related Theoretical and Formal Frameworks	155
7.4.1	Theoretical Frameworks	155
7.4.2	Computational Models	157
7.5	Latent Polyphony	160
7.6	Free Polyphony	172
7.7	Latent Entities	175
7.7.1	Harmonies	175

Contents

7.7.2	Voice-Leading Schemata	180
7.8	Harmonic Syntax	186
7.9	Conclusion	190
8	An Annotation Tool for Protovoice Analyses	193
8.1	Introduction	193
8.2	Software Components	196
8.2.1	The Annotation Tool	196
8.2.2	The Viewer Widget	199
8.2.3	The Internal Library	200
8.3	Data Formats	200
8.4	Conclusion	201
9	Bayesian Modeling of Protovoice Structure	205
9.1	Introduction	205
9.2	Generative Models and Probabilistic Programs	206
9.3	A Proof-of-Concept Model	210
9.4	Evaluating the Proof-of-Concept Model	216
9.4.1	Parameter Inference	216
9.4.2	Baseline Parsing	217
9.4.3	Results and Discussion	219
9.5	Conclusion and Future Work	222
10	Conclusion	225
10.1	Looking Back	225
10.2	General Insights	228
10.3	Looking Onwards	230
IV	Appendix	233
A	Chord Types and Ornamentation	235
B	Examples of the Annotation Format	239
B.1	Example of an Input Piece	239
B.2	The JSON Format of an Analysis File	240
B.3	Example of an Analysis File	242
C	The Probabilistic Model	249
C.1	Model Parameters	249
C.2	Pseudocode of the Model	250
C.3	Additional Data	258

Bibliography	263
Curriculum Vitae	287

List of Figures

1.1	French Suite No. 2 in C minor (BWV 813), II. Courante by J. S. Bach, mm. 54-57.	4
1.2	Chorale “Wer hat dich so geschlagen” (BWV 244.37) by J. S. Bach, mm. 11-12.	16
1.3	The schematic relation between prediction and interpretation.	25
2.1	Latent entities and relations in different models of harmonic sequences.	36
2.2	Examples of a PDF (continuous X) and a PMF (discrete X).	40
2.3	Graphical notation of the two chord models.	44
3.1	An example of a Fonte and a candidate for a Fonte.	66
3.2	Distribution of model prediction and feature values over instances and non-instances.	76
3.3	An ambiguous Fonte match (K333-iii).	77
3.4	The parameters $\vec{\beta}$ of the model trained on the full upsampled dataset.	78
4.1	Examples of ornaments in chords.	83
4.2	A schematic example of a chord type as represented in the model.	86
4.3	The core process that generates a single note in a chord.	87
4.4	A factor graph of the the chord model.	91
4.5	The posterior distributions of the chordtones ϕ^{ct} and ornaments ϕ^{or} of the chord types that are common to the ABC+ and EWLD corpora.	98
4.6	The posterior distributions of θ_h for common chord types.	101
4.7	The clustered- θ models with the highest probability for the ABC+ and EWLD datasets.	104
5.1	Rāga Multānī with pitches in an approximate Western notation.	110
5.2	A short Multānī phrase and its derivation.	111
5.3	The upper and lower neighbors (dark) of $b3$ (black) in the Multānī rāga.	112
5.4	Conventional formal analyses of the phrase in Figure 5.2.	114
5.5	Three possible derivations of $5 \sharp 4 7 \flat 6 5$	114
5.6	An analysis of the phrase in Figure 5.2 using the rāga grammar.	114
5.7	The spine modeling the deep structure of the octave expansion in an ālāp.	118

List of Figures

5.8	Selected phrases, in order of performance, excerpted from the ascending part of an ālāp in rāga Multānī, recorded by the sitarist Dharambir Singh.	119
5.9	The first two phrases from <i>Nun komm der Heiden Heiland</i> and <i>Moanin'</i> .	119
5.10	The A part of <i>Take the A-Train</i> and a summary of its underlying lines.	120
6.1	An example of free polyphony in J. S. Bach's Allemande BWV 812 I.	130
6.2	A protovoice derivation of the notes in Figure 6.1.	134
6.3	The three operations on outer structure.	135
6.4	An example derivation of a short cadential phrase.	137
7.1	The beginning (mm. 1 and 2.1) of Invention No. 13 in A minor, (J. S. Bach, BWV 784).	146
7.2	The Representation of a piece during a protovoice derivation.	151
7.3	The protovoice derivation of the pattern in Figure 7.1c.	154
7.4	Derivation of the surface in Figure 7.1a.	161
7.5	Mm. 1-4 of the Cello Suite in G major, I. Prelude (J. S. Bach, BWV 1007).	164
7.6	Mm. 2-4 of the Partita in A minor for Solo Flute, II. Corrente (J. S. Bach, BWV 1013).	165
7.7	The A part (mm. 1 to 8) of <i>Fly Me to the Moon</i> (Bart Howard, 1954).	166
7.8	The A part (mm. 1-8) of <i>Con Alma</i> (Dizzy Gillespie, 1954).	168
7.9	Two examples of melodies with static latent structure.	169
7.10	Two basic melodic phenomena arising from protovoices.	170
7.11	The first phrase (mm. 1-4) of <i>Hinunter ist der Sonne Schein</i> in four parts.	171
7.12	Mm. 1-5 of <i>Träumerei</i> (Robert Schumann, Op. 15 No. 7) with minor simplifications.	173
7.13	Mm. 1-3 of the French Suite in D minor, I. Allemande (J. S. Bach, BWV 812).	174
7.14	Mm. 1-2 of the French Suite in D minor, I. Allemande (J. S. Bach, BWV 812).	177
7.15	Gjerdingen's Example of a Romanesca from Handel's exercises for Princess Anne.	181
7.16	A Fonte at the beginning of Piano Sonata No. 3 in B♭ major K. 281, III. (W. A. Mozart).	182
7.17	Measures 2 and 3 of the French Suite in D minor, BWV 812, I. Allemande.	183
7.18	The functional structure of the harmonic progression from the Cello Suite (Figure 7.5).	186
7.19	The G ⁷ chord serves as both a harmonic preparation to C as well as a voice-leading connection between G and C.	187
7.20	Contrapuntal (a) and harmonic (b) progressions generated through elaboration.	188
8.1	Representations and operations in the protovoice model.	194
8.2	Overview of the annotation interface.	196

8.3	An example of a reduction workflow in the annotation tool.	197
8.4	The viewer component can also display the inner protovoice graph at the current step.	199
9.1	An overview of the structure of the model program.	212
9.2	Four possible ways to reduce the current surface (a) at the point between frozen and open transitions (*).	218
9.3	Baseline vs. test derivation logppn for each cross-validation split.	222
A.1	The posterior distributions of the chordtones $\phi^{(ct)}$ and ornaments $\phi^{(or)}$ of the chord types that are specific to either the ABC+ or the EWLD corpus.	235
A.2	Posterior distributions of the chord type probabilities χ	237
A.3	Posterior distributions of the note rate λ	237
C.1	Prelude in C major by J. S. Bach (BWV 939).	258
C.2	Analysis of Bach's prelude in C major (BWV 939).	259
C.3	Prelude in D minor by J. S. Bach (BWV 940).	260
C.4	Analysis of Bach's prelude in D minor (BWV 940).	261

List of Tables

3.1	List of schemata with their variants and number of occurrences in the Mozart Piano Sonata dataset.	69
3.2	Performance of the model in different conditions.	75
4.1	Representation of a chord instance as a data point.	88
4.2	The note-type heuristic used to infer if an encoded pitch represents a chord-tone or an ornament, if known.	93
4.3	Subcorpora of the ABC+ corpus.	95
5.1	A formal description of the rāga Multānī, showing the direction and hierarchical level of each scale degree.	111
7.1	Note-generating operations.	152
9.1	The set of examples used for evaluating the model.	219
9.2	Log-probability per note (higher is better) and perplexity per note (lower is better).	221
C.1	An overview over all global parameters and their prior distributions. . .	249

List of Algorithms

6.1	The steps of the parsing algorithm.	140
9.1	Markov chains, HMMs, and PCFGs expressed as probabilistic programs.	210
9.2	Generating global parameters and outer structure.	213
9.3	Generating inner structure.	215
9.4	A baseline parsing algorithm.	217
C.1	Top-level structure of the model: sampling parameters and derivations.	250
C.2	Sampling outer structure (splits, spreads, freezes).	251
C.3	Sampling freeze and spread.	252
C.4	Sampling a split.	253
C.5	Elaborating regular edges.	254
C.6	Elaborating passing edges.	254
C.7	Elaborating single notes.	256

List of Listings

2.1	A probabilistic program written in Python using the Pyro framework. . .	52
8.1	The format of a <code>.piece.json</code> file expressed as a PureScript type.	201
8.2	Parts of the format of a <code>.analysis.json</code> file expressed as PureScript types.	202
B.1	An example of a <code>.piece.json</code> file.	239
B.2	The full format of a <code>.analysis.json</code> file expressed as PureScript types.	241
B.3	An example of a <code>.analysis.json</code> file.	247

List of Publications

C. Finkensiep, M. Neuwirth, and M. Rohrmeier (2018). “Generalized Skipgrams for Pattern Discovery in Polyphonic Streams”. In: *Proceedings of the 19th International Society for Music Information Retrieval Conference*. ISMIR. Paris, France, pp. 547–553. DOI: 10.5281/zenodo.1492473

C. Finkensiep, R. Widdess, and M. Rohrmeier (2019). “Modelling the Syntax of North Indian Melodies with a Generalized Graph Grammar”. In: *Proceedings of the 20th International Society for Music Information Retrieval Conference*. ISMIR (Delft, The Netherlands). Delft, The Netherlands, pp. 462–469. DOI: 10.5281/zenodo.3527844
(included as **Chapter 5**)

G. Cecchetti, S. A. Herff, C. Finkensiep, and M. Rohrmeier (2020). “The Experience of Musical Structure as Computation : What Can We Learn?” In: *Rivista di analisi e teoria musicale* XXVI, 2, 2020, pp. 91–127. DOI: 10.53152/1032

C. Finkensiep, K. Déguernel, M. Neuwirth, and M. Rohrmeier (2020). “Voice-Leading Schema Recognition Using Rhythm and Pitch Features”. In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. ISMIR. Montreal, Canada (online), pp. 520–526. DOI: 10.5281/zenodo.4245482
(included as **Chapter 3**)

D. Harasim, C. Finkensiep, P. Ericson, T. J. O’Donnell, and M. Rohrmeier (2020). “The Jazz Harmony Treebank”. In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. ISMIR. Montreal, Canada (online), pp. 207–215. DOI: 10.5281/zenodo.4245406

C. Finkensiep and M. Rohrmeier (2021). “Modeling and Inferring Proto-Voice Structure in Free Polyphony”. In: *Proceedings of the 22nd International Society for Music Information Retrieval Conference*. ISMIR. Online, pp. 189–196. DOI: 10.5281/zenodo.5624431
(included as **Chapter 6**)

D. Harasim, C. Finkensiep, L. Bigo, M. Giraud, F. Levé, D. R. W. Sears, D. Shanahan, and M. Rohrmeier (2021). “Music Cognition: The Complexity of Musical Structure”.

List of Publications

In: *Proceedings of the Annual Meeting of the Cognitive Science Society*. CogSci. Vol. 43. Online

S. A. Herff, D. Harasim, G. Cecchetti, C. Finkensiep, and M. Rohrmeier (2021). “Hierarchical Syntactic Structure Predicts Listeners’ Sequence Completion in Music”. In: *Proceedings of the Annual Meeting of the Cognitive Science Society* 43.43

G. Cecchetti, S. A. Herff, C. Finkensiep, D. Harasim, and M. Rohrmeier (in press). “Hearing Functional Harmony in Jazz: A Perceptual Study on Music-Theoretical Accounts of Extended Tonality.” In: *Musicae Scientiae*

C. Finkensiep, M. Neuwirth, and M. Rohrmeier (submitted). “Music Theory and Model-Driven Corpus Research”. In: *Oxford Handbook of Corpus Studies in Music*. Oxford: Oxford University Press
(included as **Chapter 2**)

C. Finkensiep, P. Ericson, S. Klassmann, and M. Rohrmeier (in preparation). “Chord Types and Ornamentation – A Bayesian Model of Extended Chord Profiles”. In: *Open Research Europe*
(included as **Chapter 4**)

Foundations

Part I

1 Introduction

1.1 Music and Structure

1.1.1 Basic Structural Relations

Music is a cognitive phenomenon. While music is commonly associated with sound that has certain characteristics, “music” is not the same as “sound”, even sound of a certain quality. Instead, music corresponds to a set of mental experiences that arise when humans are confronted with such a sound, with other representations of music (such as written scores), when imagining, remembering, or inventing, or when singing or playing an instrument. The unifying aspect of all these activities is not a physical sound (the vibration of air), which does not even occur in all cases, but rather the mental representations and experiences that humans have when partaking in these activities (Longuet-Higgins 1976; Bernstein 1976; Jackendoff 1977; Pearce and Rohrmeier 2012).

Musical experiences come in a large variety, ranging from acoustic and psycho-acoustic phenomena (such as perception of individual tones, or consonance and dissonance), to recognition of familiar patterns (such motives, themes, chord types, or voice leading schemata), to anticipation and resolution, to metaphor, symbolism, and association. One particular mode of experiencing music is trying to *make sense* of the notes that are heard or read. Notes are not perceived in isolation, they get their musical meaning through relations to other notes. However, which relations are important for understanding the function of a note in its context is generally not obvious but must be inferred and is subject to *interpretation*. Everyone who has performed an unknown piece of music from a score will be familiar with the problem, that certain constellations of notes seem to be irritating and strange at first, but begin to make sense and sound “right” once they are looked at from a certain angle and integrated with their context.

An example shall help to illustrate this point. Figure 1.1 shows the last four measures of the Courante from J. S. Bach’s French Suite No. 2. Already from the number of accidentals and the prevalence of leaps in the melody one can tell that making sense of the tonal relations in this excerpt is not trivial. Indeed, looking just at the penultimate measure (m.

The image shows a musical score for measures 54-57 of the French Suite No. 2 in C minor, II. Courante by J. S. Bach. The score is in 3/4 time and C minor. The treble clef part starts with a D4 on beat 1, moves to C5 on beat 2, then to B4 on beat 3. In measure 55, it continues with B4 on beat 1, then F4 on beat 2, and B4 on beat 3. In measure 56, it starts with B4 on beat 1, then moves to A♭4 on beat 2, and finally to G4 on beat 3. In measure 57, it starts with G4 on beat 1, then moves to F4 on beat 2, and finally to E♭4 on beat 3. The bass clef part starts with a D3 on beat 1, moves to C3 on beat 2, and then to B2 on beat 3. In measure 55, it continues with B2 on beat 1, then F2 on beat 2, and B2 on beat 3. In measure 56, it starts with B2 on beat 1, then moves to A♭2 on beat 2, and finally to G2 on beat 3. In measure 57, it starts with G2 on beat 1, then moves to F2 on beat 2, and finally to E♭2 on beat 3. Chords are labeled above the staff: Dø (D4, F4, A♭4) on beat 1 of m. 56, G7b9(G78) (G4, B♭4, D5, F5) on beat 2 of m. 56, and Cm (C3, E♭3, G3) on beat 1 of m. 57. Red dashed lines connect notes across measures, indicating non-vertical relationships: D4 to G4, B4 to A♭4, and F4 to E♭4.

Figure 1.1 – French Suite No. 2 in C minor (BWV 813), II. Courante by J. S. Bach, mm. 54-57.

56), the direct relations between the notes seem strange: On beat 56.2, the G3 in the left hand is juxtaposed with an A♭4 in the right hand, forming the rather dissonant interval of a minor 9th. Similarly, the melody in the right hand mainly consists of dissonant intervals, first leaping up a minor 7th from D4 to C5 and later alternating between F and B, using the interval of a tritone twice in a row. Thus, when only considering directly occurring vertical and horizontal note relations, it looks like Bach is using a rather unstable and unconventional configuration of notes in m. 56.

Bach's composition makes more sense when we start looking at the less direct note relations (indicated by red lines in Figure 1.1): F4 and B4, while directly adjacent on the surface, can be understood as resolving *independently* into E♭4 and C5, respectively. The B is thus not primarily understood as coming from F (a tritone up) and going back to F (a tritone down), but rather as a coming from the first C5 in m. 56 and going back to C5 in m. 57. Similarly, the A♭ in m. 56 is logically neither a minor 9th above the simultaneous G, nor third from its melodic predecessor and successor, but rather another neighbor to a later G, a temporary dissonance that is resolved once the reference tone G is reached.¹ F, B, and A♭, can thus all be considered as neighbor notes to tones of the final chord. At the same time, the musical texture gives rise to a number of harmonic entities, starting with a D half-diminished chord on b. 56.1, continuing to a G⁷ chord (with a b9 on b. 56.2 resolves either on b. 56.3 or on b. 57.1) and ending on a C minor chord, which is a typical cadential progression in the minor key. In this context, even the D4 on b. 56.1 is not an isolated tone but can be seen as sustained through m. 56 before resolving into a C in m. 57 (either an implied C4 or one of the other Cs in the chord). Except for the final chord, none of these chords occurs literally as a group of simultaneous notes. Instead, the harmonic events in m. 56 (and the preceding measures) must be inferred by grouping notes that are not vertical in the original score, just as the sequential connections described above are assumed between notes that are not directly adjacent

¹There are two potential interpretations about the resolution of the A♭: Either, the A♭ is assumed to be active throughout the remainder of m. 56 and only resolve on beat 57.1 with the G entering in the same octave; or the absence of the A♭ on beat 56.3 is taken as a resolution of the A♭ to an implied G4.

in the score. However, together these inferred relations provide an *explanation* for the observed notes.

The goal of this thesis is to develop a formal model of the three basic types of tonal relations that we have seen in the above example – *sequentiality*, *simultaneity*, and functional *dependency* – as well as the latent entities that they give rise to (such as harmonies). Dependency relations refer to the purpose or “function” of a note relative to a reference note. For example, the function of the F in m. 56 is that of a neighbor note to the following Eb. Similarly, the B is a neighbor to the two Cs that precede and follow it. These dependency relations can be *recursive*: a note that depends on a reference note can itself become the reference for another note. All three types of relations are generally *latent*, i.e., they cannot be directly derived from the musical *surface* (the literal note configuration that is written or played) but must be inferred. In particular, it is unclear, which notes stand in a direct, musically meaningful relation at all. For example, the second F4 in m. 56 is not directly related to the C5 in m. 57, although they are directly adjacent in time. Instead, they are indirectly related, for example, via the Eb4 that is simultaneous to the C and at the same time the resolution tone of the F. In addition, inferred interpretations are usually *ambiguous*, as there can be several different interpretations for the same surface, although they might not be equally plausible.

The two main problems that need to be addressed are the following: How exactly are the three types of relations characterized and what is their musical interpretation? And how do the three independent dimensions they represent – horizontal (sequentiality), vertical (simultaneity), and recursive-hierarchical (dependency) – interact with each other? So far, no formal model exists that addresses all of these aspects jointly.

1.1.2 Previous Models of Tonal Structure

Models of Harmony and Voice Leading

There are a number of models that address the problem of harmonic analysis (see e.g., Aldwell and Cadwallader 2018), i.e., the extraction of a sequence of underlying harmonies (expressed as chord labels) from the surface notes (Temperley 1997; Raphael and Stoddard 2004; Sapp 2007; Rhodes et al. 2009; Temperley 2009; Mearns 2013; Ju et al. 2017; Koops et al. 2020; McLeod and Rohrmeier 2021). Harmonic analysis addresses to some extent the problem of latent simultaneity and the corresponding latent entities: The surface notes are grouped into segments each of which is supposed to represent a chord. Since not all notes in the segment have to directly represent the chord, the problem of non-chord tone identification is sometimes addressed explicitly to differentiate between

chord tones and ornaments (Chuan and Chew 2011; Ju et al. 2017; T. Hu and Arthur 2021). To this extent, harmonic labeling can create at least implicitly a form of dependency relation between chord tones and non-chord tones. However, relations between notes across chords (sequential and functional) are not modeled explicitly.

The analogous problem to harmonic labeling for horizontal organization is voice separation, i.e., the partitioning of the surface notes into a set of monophonic streams (Kilian and Hoos 2002; Chew and Wu 2005; Kirilin and Utgoff 2005; Cambouropoulos 2006; Cambouropoulos 2008; Temperley 2009; Guiomard-Kagan et al. 2015; Makris et al. 2016; McLeod and Steedman 2016; de Valk and Weyde 2018). Voice separation is generally based on the idea of auditory streaming (McAdams and Bregman 1979; Bregman 1990), which postulates that humans group tones horizontally by assigning them to common sound sources based on principles such as pitch proximity. While the equation of auditory streams and musical voices is problematic on its own (Cambouropoulos 2008, also see Section 1.1.3 below), voice separation alone says nothing about the vertical and functional relations between notes discussed above.

Another approach at treating voice leading is to model theoretical rules of counterpoint and part-writing (e.g., Aldwell and Cadwallader 2018; Huron 2016) in constraint- and optimization-based systems that aim to create voices that observe these rules as closely as possible (Ebcioğlu 1979; Herremans and Sörensen 2013; Harrison and Pearce 2020a). Similar systems often include knowledge about chords and harmonic progressions to solve a family of tasks around chorale generation and harmonization (Ebcioğlu 1988; Farbood and Schoner 2001; Allan and Williams 2004; Phon-Amnuaisuk et al. 2006; Chuan and Chew 2011; Hadjeres et al. 2016; Whorley and Conklin 2016; Colombo and Gerstner 2018; Wilk and Sagayama 2019; Wassermann and Glickman 2020). These approaches integrate harmonic and voice-leading structure to varying extents, and can also contain knowledge about ornamental notes, usually in the form of local passing notes, neighbor notes, and suspensions as they are used in chorale-style part writing.

While some of these approaches have produced impressive results, the general methodology is of limited use as a model of musical understanding for several reasons (which apply to various degrees to the different solutions, but to some extent to each solution): First, they usually aim at reproducing a musical style and its specific composition techniques rather than modeling a general understanding of tonal relations. As a consequence, they either implement explicit rules and heuristics (based on theoretical writings about composition techniques, such as Fuxian counterpoint, or on the authors own insights and intuitions), or the train implicit black-box models on dataset (e.g., through deep learning) that are difficult to interpret. Second, they are usually only concerned with generating a piece, not with obtaining an interpretation of its structure.

Notable exceptions are the systems by Ebcioğlu (1988), which includes a hand-crafted heuristic parser of Schenker-like analysis for the generated melody and bass (that does not seem to contribute to the composition process, however), and the unified model by Temperley (2009), that presents a generative probabilistic system for tonal music that can be inverted to provide analyses (similar to the model proposed in this thesis, in this respect). Third, most of these systems make relatively strong assumptions about the musical texture (chorale style with a fixed number of voices, including bass, melody, and inner voices), and none of them explicitly models latent relations between notes that are not directly adjacent vertically or horizontally (at least up to the insertion of simple, non-recursive ornamental notes). Thus, phenomena such as latent polyphony and latent simultaneity as we have seen in Figure 1.1 are not addressed in the general case and thus general polyphonic structure remains an unsolved problem.

Two notable theoretical accounts of voice leading from the constraint-based perspective have been given by Huron (2016) and Tymoczko (2011). Both accounts build on a set of constraining principles or compositional goals that shape music in the Western tradition. Huron starts from a set of canonic voice-leading rules and explores the underlying perceptual effects (such as pitch perception and auditory streaming) that explain why these rules have become standard. Tymoczko, on the other hand, begins with a set of five “features” that in his view constitute the foundation of tonality. Instead of investigating their perceptual underpinnings, he then demonstrates how the combination of these features give rise to the musical patterns observed in (extended) common practice tonality, in particular a syntax of chord progressions.

Like the other approaches on harmony and voice leading mentioned above, both Huron and Tymoczko integrate horizontal and vertical relations but do not take into account the functional-hierarchical aspect. While Huron discusses ornamental notes in the context of embellishing homophonic part writing (his Chapter 9), he does not consider a recursive form of elaboration. The “hierarchical streams” he mentions (his Chapter 13) refer to a hierarchical understanding of what constitutes a stream, not to functional dependencies. Tymoczko considers functional relations between chords, but uses them in a non-hierarchical state-transition model (his Chapter 7, see also the section on sequential models below). He even explicitly discusses the (non-)relation of his model to the hierarchical structures in Schenkerian analysis (his Section 7.6, see also the section on Schenkerian analysis below), advocating for a “pluralist” perspective that asserts independence between the harmonic and Schenkerian view on a piece. The primary reason for this non-relatedness is that Tymoczko intends his model as a characterization of which harmonic progressions are licensed or typical, a solution to the constraints given by his five principles of tonality, rather than a means of uncovering the latent structure of a given piece. Even if a non-hierarchical model is sufficient to describe typical harmonic

progressions (although that is debated, see Rohrmeier 2011; Rohrmeier 2020a), it does not necessarily capture all functional dependencies between chords, in particular if those are nested.²

Sequential Models

A general model architecture that is used by many of the above approaches (but also for other problems) are probabilistic, generative models that produce a piece from left to right (or vice versa), namely Markov and hidden Markov models (for a review, see Pearce and Wiggins 2012; Rohrmeier and Graepel 2012). It is interesting to note the difference between the two model types: Markov models (and similar non-probabilistic models) generate new tokens based solely on previously generated tokens and features that can be deterministically derived from previous tokens (e.g., Ebcioğlu 1988; Conklin and Witten 1995; Farbood and Schoner 2001; Pearce 2005; Chuan and Chew 2011; Moss, Neuwirth, Harasim, et al. 2019). 1st-order Markov chains consider only the directly preceding token, while higher-order models consider longer contexts. For Markov models of any order, however, all information that is needed to understand the generative process of a sequence is entirely represented in the sequence, so there is no uncertainty about the decisions the model made when the sequence is fully observed. Markov models thus provide no interpretation of a piece and its latent structure, they just observe the sequence of tokens. However, Markov models learn latent structure that is shared across several sequences, which is represented in their transition probabilities. In particular, viewpoint models (Conklin and Witten 1995; Pearce 2005; Pearce and Rohrmeier 2012; Whorley and Conklin 2016) represent this shared knowledge in a structured way by using *viewpoints*, different representations (or *features* in modern machine learning terminology) of the surface tokens that allow parameters (and thus predictive information) to be shared between different surface configurations with the same viewpoint values. Viewpoint models, such as IDyOM (Pearce 2005), also extend basic Markov models with a number of additional mechanisms: A backoff mechanisms allows the model to look at contexts of different lengths, which addresses the problem that longer contexts are more informative but also much rarer to encounter several times. A combination of long-term (offline) and short-term (online) learning can capture both stylistic regularities in a corpus as well as recurring patterns within a single piece, and thus be seen as a form of inference to latent information (namely patterns) within a piece. Finally, the set of viewpoints (including derived viewpoints) can be selected according to their predictive power, which constitutes a form of learning beyond the bare transition probabilities.

²In linguistic terms, a non-hierarchical and a hierarchical model can be *weakly equivalent* (in particular cases) without being *strongly equivalent* (Chomsky 1963).

Hidden Markov models, in contrast to regular Markov models, make use of a hidden state (e.g., Allan and Williams 2004; Mavromatis 2009; McLeod and Steedman 2016; C. W. White and Quinn 2018; Wilk and Sagayama 2019; Duane 2019; Wassermann and Glickman 2020; also Tymoczko 2011, as discussed above). At each step, the current token in the sequence as well as the next hidden state are determined based on the current state. The latent states, however, are generally not observed, only the sequence of emitted tokens. Hidden Markov models thus represent *latent* information that is not directly represented in the observed surface. Similar to the latent relations and entities we identified in the Bach example in Figure 1.1, the latent states provide explanations for the surface tokens. A token might be unlikely under one state but have a high probability under another state, and an unusual progression of surface tokens might be explained by a change in the latent state. Hidden Markov models are therefore often used in the context of harmonic analysis: Latent states encode harmonies, while state transition model harmonic progression and emission probabilities model the realization of harmonies by surface notes.

Hierarchical Models

A limitation of sequential models such as (hidden) Markov models is their limited ability to capture recursive dependency structure since generated elements can only depend on their predecessors in the sequence. More complex, embedded forms of dependency structure can be modeled by grammatical models such as context-free grammars (CFG, Chomsky 1956). In a CFG, a sequence of symbols is recursively elaborated through the application of production rules that replace a symbol by a string of symbols. In this way, grammars can establish relations both between nested, non-adjacent elements in a sequence, and between surface entities and latent entities. Grammars (both CFGs and other grammar formalisms) have been used to model recursive structure in melodies (Bod 2001; Gilbert and Conklin 2007; Groves 2016; Abdallah, N. E. Gold, and Marsden 2016; Nakamura et al. 2016), chord progressions (Rohrmeier 2011; Rohrmeier 2020a; Harasim, Rohrmeier, et al. 2018; Granroth-Wilding and Steedman 2014; Melkonian 2019), and rhythm (Melkonian 2019; Harasim, O'Donnell, et al. 2019; Foscarin et al. 2019; Rohrmeier 2020b).

The main limitation of standard grammar formalisms is that they produce sequences of tokens (i.e., chords or melody notes) but are generally not able to integrate several dimensions of relations. As a result, two families of grammar models have developed: Object-based grammars (e.g., Hamanaka et al. 2016; Rohrmeier 2020a; Harasim, Rohrmeier, et al. 2018) operate on musical objects such as notes or chords, and can thus express modification of existing objects (e.g., harmonic substitution) or the derivation of objects

from abstract categories that need not correspond to surface entities. Transition-based grammars (e.g., Mavromatis and Brown 2004; Yust 2006; Gilbert and Conklin 2007), on the other hand, elaborate the transitions between two sequential objects, such as the interval between two notes. This allows the generated objects to depend on two parents (which is essential for passing notes, for examples), but prevents already inserted objects from being modified or used as latent entities. One of the main technical contributions of the present thesis is to introduce a class of models that unify the elaboration of objects and transitions, which allows both latent simultaneities (such as chords) to be horizontalized and latent note transitions to be ornamented.

One reason that using grammars to model musical structure has been popular for a long time is that it has been argued that the hierarchical properties of music and language are similar and even that their underlying biological mechanisms might be tightly related (e.g., Forte 1967; Winograd 1968; Bernstein 1976; Katz and Pesetsky 2011; Fitch and Martins 2014; but see also Jackendoff 2009, for a more skeptical view). The first major attempt at a unified model of tonal structure using hierarchical tools inspired by linguistics was the Generative Theory of Tonal Music (GTTM) by Lerdahl and Jackendoff (1983). Despite its name, the GTTM is not a *generative* theory in the sense that it describes a generative process that produces the observed notes (as noted by others, e.g., Longuet-Higgins 1983; Rohrmeier 2007). Instead it defines a space of possible interpretations of a piece by means of well-formedness rules, together with a set of preference rules for obtaining analyses in a bottom up fashion.

The GTTM defines four types of hierarchies: a metrical grid of stronger and weaker time points, a grouping structure that expresses a piece as a hierarchy of nested segments, a time-span reduction that adds a *head* (the most salient event) to groups and a prolongational structure, which forms a separate hierarchy of superordinate and subordinate events. While time-span reduction is based on the grouping structure (and thus a form of nested constituency), prolongational structure rather expresses a form of dependency among events based on a notion of relative stability and movement away and towards stable points (tension and relaxation), which may not agree with the time-span hierarchy. Prolongational structure thus comes closest to the type of structures that are of interest in this thesis, although a different set of functional relations is assumed here. A somewhat odd property of prolongational structure in GTTM is that it is introduced as a *dynamic* hierarchical phenomenon: the progression from a stable note to a neighbor note, for example, increases tension while the progression from the neighbor back to the reference constitutes a relaxation (Lerdahl and Jackendoff 1983, their Figure 8.1, p. 180). Longer progressions can exhibit both internal and overarching tension or relaxation (their Figures 8.2 and 8.3). Despite this dynamic character, the GTTM encodes prolongational structure using binary trees over events (their Figures 8.4

ff.), which necessarily ignores one of the two connections between an event and the two adjacent events. This shortcoming is particularly evident in the examples that Lerdahl and Jackendoff themselves use to introduce prolongational structure, (complete) neighbor and passing motions. Not only does the middle note (the neighbor or passing note, respectively) logically depend on both its predecessor and its successor as reference points, a tree analysis of these configurations cannot express both the tension leading to the ornament and the relaxation going away from it at the same time. This problem has been noted and addressed in subsequent models of tonal dependency structure, which describe hierarchies of *transitions* rather than events (see above).

Since the GTTM analyses are derived from the surface of a piece, they constitute passive interpretations rather than explanations of a piece (see also Section 1.2.2), although later computational implementations have aimed to extend the theory to generative, grammar-like models (e.g., Nakamura et al. 2016). Besides that, the GTTM represents the surface as a string of events, either single notes (for melodies) or vertical groups of notes (for polyphonic textures), but only describes relations between events, not between the individual notes within a complex event. It thus shares a lack of voice-like note-to-note connections with most grammar-based models and like them is unable to capture polyphonic structure.

Schenkerian Analysis

The arguably most elaborate theoretical framework for describing tonal relations is Schenkerian analysis (Schenker 1979; see Cadwallader and Gagné 2011, for an introduction). Schenkerian analysis involves all three types of relations between notes: sequential, vertical, and functional, including latent relations of each type. It also assumes a hierarchy of latent configurations, reductions of the piece's surface that are organized in a series of levels, which outline the evolution of a piece from a shared background structure (the *Ursatz*) to its specific surface. The levels are connected by a number of transformations (*prolongations* or *diminutions*) that elaborate the structures of the higher levels and introduce the latent relations between latent and surface entities. Being a music-theoretical framework, Schenkerian analysis is usually described informally, and relies on the intuition of the analyst to apply these transformations correctly and select the most plausible and musical interpretation of a piece. For this reason, it has proven notoriously difficult to formalize, and its logical consistency and adequacy as a psychological (pre-)theory have been subject of debate (e.g., Narmour 1977; Temperley 2011).

While Schenkerian analysis seems to achieve the integration of the three dimensions,

all previous attempts at formalizing and implementing it so far had to abandon this integration in some way. For example, a family of models based on transition elaboration (Marsden 2001; Yust 2006; Kirlin and Utgoff 2008; Kirlin and Jensen 2011) only describe relations within melodies. These models have later been extended to integrate several voices at the cost of assigning one primary voice that carries the relations (Yust 2015b; Yust 2018) and to homophonic note-groups similar to the GTTM representation (Kirlin and Thomas 2015; Mavromatis and Brown 2004), which in principle allows for horizontal relations between individual notes but cannot express the horizontalization of latent verticalities, as discussed above. Other approaches have prioritized event-based elaboration (e.g., Marsden 2010), which can capture horizontalization but not consistent sequential relations and elaboration of note transitions (e.g., for passing notes).

Besides the practical difficulties with expressing Schenkerian analysis in a consistent formal language, there are some underlying theoretical issues with using it as a model of general perception and understanding: First, many of the diminutions are not general and fundamental operations on structure but rather specific and high-level transformations. Some of them can only be applied in specific conditions or in a specific way depending on context, such as many of the transformations between the *Ursatz* and the first level (e.g., the initial ascent (*Anstieg*), a linear progression up to the head of the *Urlinie*, Schenker 1979, §§ 120 ff.). Some diminutions can be decomposed into a set of simpler operations, such as the voice exchange, which consists of note repetition combined with octave transposition (§§ 236, 237). Finally, most of the structures and operations have some very specific connotations beyond the notes that they produce and beyond the primitive operations in which they might be decomposed, such *Züge* (linear progressions, §§ 113 ff., 203 ff.) which have a sense of directionality (the “will” to move forward, § 116) and of vertical connection between their endpoints (§ 115), or the voice exchange which establishes a relation between two voices (§ 236). All of this taken together makes clear that Schenkerian theory does not operate on the most general and fundamental level that this thesis is looking for. This situation can be compared to the relation between physics and chemistry, where the former provides the principles while the latter describes specific structures and process (i.e., substances and reactions) based on these principles in a contextual, conditional, and more macroscopic way. The difference is that in the case of Schenkerian theory it is not precisely known how it maps to the fundamental principles of tonal structure or how these principles look like.

The second difficulty lies in the lack of orthogonality in Schenkerian principles. Since Schenker’s point of reference is the piece, in which all constraints and conditions are satisfied, there is little reason for him to postulate principles that are *independent* from each other. On the other hand, some degree of independence is required for a limited set of principles to produce the high variety of pieces. As a result, Schenker generally claims

that diminutions can be freely chosen (e.g., § 47), but rarely gives a characterization of a specific transformation or configuration that does not integrate several principles at once. For example, the earlier transformations of the *Ursatz* are presented almost like a lexicon of possible forms (e.g., the possible extensions of the fundamental bass in Schenker 1979, Fig. 14 - 19) rather than general formulations of these transformations, with variants that are impossible due to a different principle already excluded. This conflict between an analytical or combinatorial and a holistic perspective, as well as the aforementioned specificity of Schenkerian concepts can be seen as a consequence of Schenker describing compositional practice (and its perception) rather than the general perception and interpretation of structure in music (or even in Western tonal music). In particular, there is no real distinction between what is understandable in principle – the *language* – and what is considered adequate, idiomatic, or licensed – the style.

There are two ways to resolve the above issues: One option is to take Schenkerian analysis as a starting point and try to disentangle, sharpen, and generalize its concepts. This is the route that most of the formalization attempts mentioned above have taken, although those have usually adopted the more generic features of Schenkerian theory (e.g., its recursive structure) rather than its more specific concepts. The other approach, which is adopted here, is to construct a new theory based on principles that have an independent justification and are orthogonal, general, and primitive by design. Such a theory need not abandon the insights provided by Schenkerian theory, and the outcome might in fact look very similar to Schenkerian theory or some of its formalization attempts. However, its principles would be supported by an independent motivation instead of an argument to Schenkerian authority, and there would be neither guarantee nor obligation to be compatible with Schenkerian theory in all respects.

Expectation and Schemata

A different line of criticism against Schenkerian analysis argues that it does not cover all aspects of musical organization that are relevant to both the experience of music in time and to historical composition (or improvisation) practice. The first aspect refers to the experience of expectation and anticipation when listening to a piece of music, which Schenkerian analysis is not really concerned about and only addresses implicitly. The implication-realization model (Narmour 1977; Narmour 1990; Narmour 1992) proposes a more direct, less rationalistic mode of musical experience than analytical approaches: By directly observing melodic patterns that immediately precede the current point in time, expectations about the continuation of these patterns are formed. The formation of these implications and their realization (or denial) then give rise to a musical and emotional experience. This perspective has sparked a separate line of psychological

and theoretical research (e.g., Huron 2006; McAdams 2004; Pearce and Wiggins 2006; Pearce and Wiggins 2012; Vuust and Witek 2014; Arthur 2017; Sears, Pearce, et al. 2018; Sears, Spitzer, et al. 2018; B. P. Gold et al. 2019; Koelsch et al. 2019; Morgan et al. 2019; Quiroga-Martinez et al. 2021; Vuust, Heggli, et al. 2022) and serves as a motivation and justification for some of the left-to-right models already mentioned above (e.g., Conklin and Witten 1995; Pearce 2005; Pearce and Wiggins 2006). Models of expectation in music are usually sequential models as described above: they make use of both vertical and horizontal note relations for predicting the next event, but they usually do not consider hierarchical relations between notes. Instead, the dependency of interest is the one between the upcoming event and its predecessors, as well as higher-order quantities such as the uncertainty about the next event. Similarly, prediction models are often (but not necessarily) less concerned with latent entities and rather try to predict the next event directly based on its predecessors. It should be noted that the set of musical experiences (and the corresponding computational problems) captured by the expectation perspective are different from the ones that motivate this thesis (i.e., understanding and interpretation). The differences and intersections between the two perspectives will be discussed in greater detail in Section 1.2.3.

The other major theoretical perspective focuses on historical composition practice based on compositional patterns, such as voice-leading schemata and cadences (Forte 1979; R. O. Gjerdingen 1988; R. Gjerdingen 2007; Jan 2013; Rice 2015; Rabinovitch 2018; IJzerman 2019). Schema theory asserts that historical composition and improvisation have been based on the reproduction and concatenation of prototypical patterns, which can in turn be recognized by an informed listener. It is thus somewhat complementary to Schenkerian theory: instead of a shared set of relations and operations that can encode a large variety of configurations (which are proprietary to each piece), schema theory provides a vocabulary of fixed configurations, which themselves are shared across pieces. Instead of general but abstract principles, the composer is believed to apply and combine ready-made templates, which is why schema theory has been compared to similar construction-based approaches in linguistics (R. Gjerdingen and Bourne 2015). Since schema prototypes are usually not reproduced literally, but ornamented and elaborated, they constitute another class of latent structural entities, similar to harmonies (but larger, more complex, and more specific). Despite an increased theoretical interest in voice-leading schemata, there have been relatively few computational models that address them (e.g., Symons 2017; Finkensiep, Neuwirth, et al. 2018; Katsiavalos et al. 2019). In the context of this thesis, the focus is less on the characterization of schema prototypes, their contents, functions, and implications, but rather on how schemata (as latent entities) can serve as explanations of the surface. Thus, the main question is how schema prototypes are connected with the notes encountered on the musical surface, which will be discussed in Chapters 3 and 7.

1.1.3 The Voice Problem

One of the three basic types of relations discussed in Section 1.1.1 is *sequentiality*: notes are usually considered to form horizontal progressions with other notes, they have predecessors and successors. What exactly characterizes these sequential progressions? The traditional answer to this question is that notes form *voices*, contiguous streams of notes in which notes do not overlap and in which every note has exactly one predecessor and one successor, except for the first and the last note of the voice.³ In many cases, these voices are known explicitly either because the piece is just a single melody (which forms a single voice) or because it is written in several concurrent parts, for example for several singers or instrumentalists. Likewise, theoretical accounts of voice leading are usually based on the combination of a fixed set of voices, for example a cantus firmus and a counterpoint, or a set of parts including soprano, bass, and middle voices (e.g., Aldwell and Cadwallader 2018).

As the example in Figure 1.1 shows, the explicitly written parts of a piece do not necessarily coincide with the musically meaningful sequential connections between notes. This problem becomes even more obvious when considering free polyphony, where notes are not organized in strict parts in the first place. This has two consequences: First, horizontal relations between notes are not always directly observable and must be inferred, just like other forms of latent structure. Second, if sequentiality between notes is not given, we need a criterion for determining which notes are sequentially related and which are not. One such criterion is derived from *auditory streaming* (McAdams and Bregman 1979; Bregman 1990), which assumes that a listener interprets acoustic events as originating from a set of sources. When two events are assigned to the same source, they are considered to be part of the same stream. This idea is applied to implicit voices too: If a listener assigns two notes to the same stream (based on Gestalt principles such as pitch proximity or good continuation), they are considered to be in a sequential relationship. In the excerpt from Figure 1.1, for example, one could assign the notes C5, B4, and C5 to one stream and the notes Ab4 and G4 to a different stream, each of which is an implied voice.

In the common understanding of streams – as, for example, used by voice-separation algorithms (see Section 1.1.2) – streams are distinct entities that keep their identity over time. A note is never part of two voices at the same time since this would imply that it was produced by two different sources at once. However, Figure 1.2 shows that the same is not generally true for sequential relations in tonal music. The shown four-part

³The condition of monophony is sometimes dropped for streams of chords (that do not give away independent voices) or voices that are so strongly coupled that they seem to fuse into one voice with a complex timbre (Cambouropoulos 2008; Huron 2016)

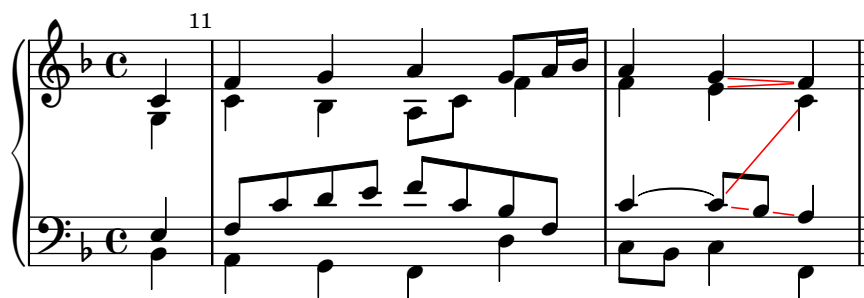


Figure 1.2 – Chorale “Wer hat dich so geschlagen” (BWV 244.37) by J. S. Bach, mm. 11-12.

chorale ends on a perfect authentic cadence. The alto part moves down from the leading tone E4 to a C4 in order to complete the tonic triad. The tenor part also moves down from C4 to A3 through a passing B \flat 3. When disregarding the written parts, however, there is a second interpretation of the voice leading in this configuration: The E as the leading tone wants to resolve into the root of the tonic chord F, not only because this is the closest possible resolution, but also because its function as a leading tone (a lower neighbor of the tonic) is defined relative to the tonic F. However, the soprano’s preferred resolution direction is also down to the tonic, and its 2-1 progression is important for expressing cadential closure, so the two progressions (7-1 and 2-1) meet on a single tone. Conversely, the C4 in the tenor is involved in two kinds of progressions, one going down to A3 via B \flat 3, the other reflecting a repetition of the C, the only common tone between the two chords, which again reflects parsimonious voice leading and another functional relation: repetition.

The underlying problem that causes this apparent conflict is that there are at least three criteria for defining voice-like relationships.⁴ One criterion is simply what is written. This criterion only really applies to explicitly written parts, as there is not one single systematic way parts are constructed (parts in fugues work very differently from parts in chorales). The second criterion is voice assignment based on streaming. From a streaming perspective, both the E4-F4 and the C4-C4 connection are locally plausible (because of pitch proximity), but they are excluded contextually: the most parsimonious voice assignment must assign the F4 to the soprano and the A3 to the tenor, leaving the C to the alto voice. The example shows, that notes can be involved in several sequential progressions at the same time, they can have several predecessors or successors. Only if the assumption of mutually exclusive streams is dropped, configurations such as two voices meeting on the same note (F) or departing from the same note (C) can be expressed, allowing the third criterion to be applied: functional relatedness. From this

⁴The ambiguity of the concept of a “voice” has been observed by Cambouropoulos (2008), but in the context of a different problem (voice separation) and correspondingly with slightly different conclusions.

perspective, the C4 in the tenor is both the starting point of a passing motion and a repetition of the final C4 in the alto. Similarly, the final F4 is the goal of both a downwards passing motion and a lower neighbor motion, and thus is the successor of both G4 and E4.

I argue that for voice-leading structure in free polyphony, only the criterion of functional relatedness is generally meaningful. The first criterion does not have an intrinsic definition or interpretation, it simply reproduces the explicit parts as notated by the composer and thus does not apply to implicit voice leading. Moreover, the example in Figure 1.1 has shown that notated voices do generally not indicate the musically meaningful relations between notes, especially in the presence of latent polyphony. Streaming refers to an interpretation of a sequence of tones as produced by the same sound source. While looking for underlying sound sources is meaningful for the actual purpose of analyzing auditory scenes (i.e., identifying physical sources of acoustic events), its application to musical structure requires the additional assumption that composers intentionally try to evoke the impression of independent interacting sound sources or actors. While this could be argued for in the context of explicit polyphony (Huron 2016), it is rather unclear whether it also applies in the general case. In particular, the example in Figure 1.2 suggests that distinct streams (from distinct sources) cannot capture all relevant sequential relations at the same time. Functional relations, on the other hand, lie at the heart of tonality since they express how notes are understood relative to other notes. In fact, they do not only apply to free polyphony (where the other two criteria break down), but also to explicit polyphony, as shown in the example above.⁵

For these reasons, this thesis proposes a new, network-based model of functional voice-leading relations in which pairwise connected notes form a directed acyclic graph of horizontal progressions. Since these progressions are only pairwise and do not unambiguously imply larger “voice-like” entities, they are called *protovoices*. Earlier approaches have already abandoned the explicitly written parts in favor of more abstract forms of counterpoint (e.g., Benjamin 1981) or atemporal voice-leading spaces between chords and scales (Tymoczko 2006; Cohn 2012; Yust 2015a; Harasim, Schmidt, et al. 2016; Harasim, Noll, et al. 2019; Harrison and Pearce 2020a). Separately, it has been observed that voices need not be exclusive but can split and merge on shared notes (Marsden 2005). However, an integrated and fully formal model of free functional voice-

⁵It could be argued that, in certain situations, one might be explicitly interested in streaming effects or written parts as opposed to functional voice-leading relations. My point here is that (1) these different notions of voice leading are *not identical* and that (2) the functional notion is the implied one when referring to “voice leading” in the context of general tonal structure, in particular free polyphony. This is independent of whether voice leading is studied from the perspective of perception, composers’ intentions, or analysis, although these different perspectives might be interested in the other notions as well (e.g., streaming effects in perception).

leading structure has not been proposed so far. That being said, the idea that strings of notes form network-like structures is clearly present in Schenkerian theory which (like protovoices) aims to capture voice-leading structure without reliance on explicit voices. In particular, *Züge* are connections between the notes of different voices and thus introduce a subordinate structure in which the initial note and the target note both have a double function. As discussed above, however, the operations that generate and manipulate linear progressions in Schenkerian theory are rather specific and high-level, whereas the protovoice model aims to describe the most fundamental and general level of horizontal progressions.

1.2 Interpretation

1.2.1 Interpretations as Explanations

As argued in Section 1.1, structural relations in music are generally latent interpretations. Interpretations in the most general sense are abstract representations that are in some way derived from the surface observations and may be ambiguous, as different interpretations of the same observations are possible. There is, however, a stronger notion of interpretation under which an interpretation *explains* an observation: The latent entities and relations that are assumed in the interpretation provide a justification for why the observation looks the way it does. For example, interpreting a group of notes in a piece as an instance of a chord could mean that they to some extent resemble a chord, or that the impression of a chord is evoked by the notes (*weak interpretation*). On the other hand, it could also mean that assuming an underlying chord explains why these particular notes occur in the piece (*strong interpretation*): the (actual or hypothetical) composer decided (consciously or intuitively) to use that chord and then chose the notes accordingly. A weak interpretation is an interpretation that passively results from the observation, and this connection from observation to interpretation is problem specific (e.g., which chords are evoked by which notes). A strong interpretation, in contrast, constitutes a cause or origin of the observation, something that presumably produced or gave rise to the observation. For a given observation, strong interpretations are obtained by reasoning to the best explanation. The generative or causal connection is again problem-specific (e.g., which chords produce which notes), but the inference process is generic: it selects the most plausible explanation.

What does it mean that an interpretation “explains” an observation? Explanation can refer to causal relations. For example, a particular pattern of light that falls on the retina of a human eye can be explained by a combination of physical objects and light sources

that cause this particular pattern. Assuming a set of underlying objects and light sources is thus a strong interpretation (in the causal sense) of the retinal input. The relation between latent causes and observations need not be as direct as in the visual case. When trying to understand a linguistic utterance (e.g., a sentence or a question), for example, a strong interpretation refers to the speaker's intentions. Similarly, paintings or drawings of objects do not necessarily resemble the physical objects they refer to. Still, humans are able to recognize that a stick figure stands for a person, not because the visual system is deceived into believing that the visual impression is caused by an actual human body instead of a drawing, but because we are able to infer the underlying intention of the drawer, and because we are aware of the social convention of using stick figures to represent persons in an abstract way. In both cases, the intention of the speaker or the drawer together with the observer's knowledge about how intentions cause the observed entities explain that a certain observation was made.

An observer can never be completely certain that their interpretation of an observation reflects the *true* causes (e.g., the objects that actually produced a visual sensation) since these are generally not directly accessible (and thus *latent*) and because there are usually several possible explanations for the same observations. This divide between latent causes and observations has two consequences: First, when inferring possible explanations of an observed event, the observer must be able to deal with and reason about *uncertainty*. Second, an interpretation need not always correspond to the true underlying causes. This is particularly true when interpretations refer to abstract concepts such as intentions. Speaker and listener might not even share the exact same concepts, which would make it impossible for the listener to recover the exact original intention. In the case of music (or art in general), it becomes even less important to recover the original intention of the composer or artist. From the perspective of a listener, any interpretation that provides a good explanation of the observed piece and connects it to the concepts, norms, and conventions familiar to them, is a satisfactory strong interpretation.⁶ Thus, strong interpretations are subjective explanations in the sense that they merely explain the observations to the observer and do not necessarily reflect the true world.⁷

⁶The same argument can even be made in the case of physical objects: Mental representations of objects usually do not refer to the "true" physical properties of an object (e.g., its molecular structure) but to abstract entities. The light that excites the cells of the retina is not reflected by a "chair" entity but by a bunch of atoms; the entity of the chair "as such" only exists in the mind of the viewer. Explaining a visual impression by referring to the concept of a chair with all its connotations does thus not exactly reproduce the physical cause of the impression, i.e., a particular configuration of interacting particles and waves.

⁷Interpretations being subjective does not imply that they are arbitrary or that the interpretations of two different observers are completely different or even unrelated to each other. Generally, observers with similar backgrounds and in similar contexts share concepts and can thus come to similar conclusions about the plausibility of an interpretation.

In the case of musical structure, interpretations are inferred on several levels. Latent relations between the observed notes in a piece explain the function of a note and its relation to its context. In addition, latent entities (such as chords, voice-leading schemata, motives, or themes) as well as relations between the latent and observed entities can explain how larger configurations of notes come about. Finally, stylistic regularities (such as chord and schema prototypes, modes, or form templates) can be shared across many pieces, generalizing explanations of individual pieces. While these latent concepts are connected to compositional intentions – in the sense that a composer might “use” a scale, a chord type, or a schema, possibly unconsciously, to decide how the notes of the piece are arranged – structural interpretation does not necessarily recover the exact intention of the composer. For one, as mentioned above, a plausible explanation from the listener’s perspective need not correspond to the composer’s goals or even use the same concepts.⁸ In addition, the composer might not just generate a piece based on naive intentions, but play with “higher-order” intentions, for example by making the intended interpretation very easy to discover (composing in a way that is difficult to misinterpret) or very ambiguous, thus playing with the communication process between composer and listener (Temperley 2007a).

Structural interpretation of a piece of music in the cognitive sense is closely related to music-theoretical analysis, which can be seen as aiming for an understanding and discussion of a piece on all levels, up to high-level aesthetic experience. In practice, however, interpretations do not need to reach this level of completeness. For a casual listener, a superficial recognition of a piece’s style as familiar might be sufficient, and even a musician might have a good understanding of a piece’s structural properties without being aware of all of its deep implications.

That humans try to make sense of a sensory input by inferring its underlying causes is an old and well-known idea in cognitive science, especially in the context of Bayesian cognition and predictive processing (Helmholtz 1860; Chater, Oaksford, et al. 2010; Clark 2013). According to the Bayesian framework, cognitive tasks that involve some form of uncertainty (such as decision making, perception, or reasoning) are solved using (approximately) probabilistic inference. For example, the plausibility of an interpretation I given an observation O is expressed as a conditional probability distribution over possible I :

$$P(I | O). \tag{1.1}$$

⁸For example, a harmonic interpretation can be plausible from the perspective of a modern-day listener, even when anachronistically applied to music that was primarily composed contrapuntally. That being said, subjective explanations that are somewhat close to the “true” underlying process of an observation usually have the best explanatory power and will thus be (on average) preferred, even from a subjective perspective.

According to probability calculus, this conditional probability can be rewritten in terms of the joint probability $P(O, I)$ and the marginal probability of the observation alone $P(O)$:

$$P(I | O) = \frac{P(O, I)}{P(O)}. \quad (1.2)$$

The joint distribution can be understood as the mental model of the observer that defines how the plausibility of different combinations of observation and interpretation are rated. In the case of strong interpretations, this joint model can be further decomposed into a *prior* $P(I)$ (the plausibility of the interpretation alone, independent from the observation) and a *likelihood* $P(O | I)$, the probability that the interpretation (if true) would give rise to the observation. Thus, the observer can infer the *posterior* distribution over possible interpretations from the prior and the likelihood:

$$P(I | O) = \frac{P(O | I) \cdot P(I)}{P(O)}. \quad (1.3)$$

When the observers model is expressed as a product of prior and likelihood, it corresponds to a *generative* model, i.e., a hypothetical process by which the observation is generated from the interpretation. The application of generative probabilistic models to music theory will be discussed in detail in Chapter 2. Bayesian models of perception have been suggested for domains such as visual perception (Knill et al. 1996; Weiss 1996; T. S. Lee and Mumford 2003; Kersten et al. 2004; Kulkarni et al. 2015; Felip et al. 2019) and language processing (Chater, Crocker, et al. 1998; Narayanan and Jurafsky 2001; Chater and Manning 2006), but also for the perception of musical structure (Temperley 2007a; Abdallah and N. E. Gold 2014; Abdallah, N. E. Gold, and Marsden 2016; Harasim, O'Donnell, et al. 2019; Harasim 2020).

Bayesian models first and foremost characterize a problem and its optimal solution. When applied to cognitive tasks, they therefore describe first of all the *computational* level of explanation, that is, they are concerned with which problem is (approximately) solved by a cognitive system, and why (Marr 1982). However, since Bayesian inference is a mathematically well-defined problem, algorithms for computing conditional distributions serve as hypotheses for the *algorithmic* level of explanation that describes how the solution to a problem is computed. Finally, the *implementation* level (how inference algorithms are realized in the brain) has been addressed to some extent by neuroscientific research (see Chater, Oaksford, et al. 2010; Clark 2013, for a review). This thesis is mainly concerned with characterizing the relationship between musical surfaces and their possible interpretations on the computational level, as well as possible algorithms that implement this inference.

1.2.2 Interpretations in Music Psychology

In music psychology (especially concerning the perception of music), interpretations of the sensory input are widely used. The focus, however, is usually on the “bottom-up” direction, that is, which abstract representations (i.e., weak interpretations) occur in music processing, and how they are obtained from the sensory input. The “top-down” direction (i.e., the explanatory link between interpretation and observation) is usually not considered. Top-down effects are, however, often acknowledged in a different sense: the choice of interpretation can be influenced by other aspects than the direct sensory input, such as prior beliefs, expectations, or integration with the context. In probabilistic terms, this would refer to the prior of the latent interpretation $P(I | \text{context})$, rather than the likelihood of the observation given interpretation $P(O | I)$.

Instead of explanatory inference, structural relations between the observed entities are usually believed to arise through heuristics such as *Gestalt principles* (McAdams and Bregman 1979; Bregman 1990; Huron 2016; Lerdahl and Jackendoff 1983; Terhardt 1987; Deutsch 1999, e.g.), similar to the ones proposed in visual perception in the first half of the 20th century (Wertheimer 1923). Strong interpretations, in contrast, describe the other direction, where the observation is “generated” from (or explained by) the interpretation based on generative principles, while the interpretation is obtained through inference to the best explanation. A wide range of different types of interpretations have been investigated in music perception both empirically and theoretically. They are usually more low-level (and thus easier to operationalize) than their music-theoretical counterparts, and include pitch and chroma (Terhardt 1974; Longuet-Higgins 1976; Krumhansl 1979; Shepard 1982; Krumhansl 1990a), melodic contours (B. W. White 1960; Dowling 1978; Jones 1987), voices or streams (McAdams and Bregman 1979; Bregman 1990; Huron 2016), key and mode (Longuet-Higgins and Steedman 1971; Krumhansl 1990a; Temperley 1999b), groups (Deliege 1987; Bregman 1990; Deutsch 1999), harmony (Parncutt 1989; Yeary 2011), and hierarchical structure (Deutsch and Feroe 1981; Lerdahl and Jackendoff 1983).

Generative and bottom-up approaches emphasize different aspects of the same type of phenomenon. Bottom-up approaches address the “what” and the “how”: which types of interpretations occur at all, and how they are derived from the observed sensory input. Generative approaches focus on the “why”: A certain type of interpretation is used because it is a useful explanation for a large set of observations. The precise “how” generally remains implicit at this level, it corresponds to whatever mechanism performs the explanatory inference. Accordingly, the biggest weakness of the bottom-up approach is the strength of the generative approach: answering the question, why humans have learned or evolved to use the abstract representations that they use, and

not others. From the generative perspective, interpretations are useful because they allow for *prediction* and *generalization*. If using one type of interpretation leads to better predictions and generalizes better to new situations, it is preferable over a different type of interpretation. The ability to make *predictions* follows naturally from the definition of strong interpretations: if latent interpretations gives rise to observations, then knowing (or assuming) a certain latent entity changes the expectations about the observations to be made. For example, knowing that a piece is written in a minor key (from listening to the beginning of the piece or from reading its title) changes the listeners expectations about the note configurations that will be encountered. *Generalization* arises naturally, when the same explanations are applied to many observations. An explanation that makes many pieces likely (e.g., that the minor mode uses certain notes more frequently than others) is probably going to be useful for a new, unknown piece as well. Just applying the “best explanation” principle to several observations (together with the predictive and explanatory power of strong interpretations) thus provide a good motivation for choosing certain types of interpretations (e.g., major and minor modes, Harasim, Moss, et al. 2021) over others.

Another difference between the bottom-up and the generative view is the role of ambiguity. From a pure bottom-up perspective, confronting an observer with a stimulus generates a percept. A stimulus may evoke different percepts when presented to different observers (or to the same observer in different conditions), but when observer and conditions are fixed, then the path from stimulus to interpretation is fixed too. From an explanatory perspective, there might be several competing explanations for the same observation, even from the perspective of a single observer. The observer may prefer one explanation as more plausible than the others (and this assessment may differ between different observers or conditions), but there is still an inherent ambiguity in the possible interpretations of a stimulus that is not directly visible from the bottom-up perspective.

It is important to note, however, that bottom-up and top-down approaches are not mutually exclusive, but rather complement each other. Bottom-up studies in music psychology often establish or confirm the existence of certain abstract representations in the first place. Moreover, since the Bayesian perspective defines inference only on the computational level, its concrete implementation for a specific inference task is subject to empirical investigation. Finally, heuristic bottom-up preference rules (such as Gestalt principles) can often be reinterpreted as generative principles (Chater 1996; Feldman 2009).⁹ For example, the principle of good continuation (elements are grouped

⁹There is some debate about whether Gestalt and Bayesian approaches are really equivalent, but this debate focuses on the question of whether the (probabilistic) *likelihood principle* and the (information theoretic) *simplicity principle* are really equivalent (Feldman 2009; van der Helm 2017). This debate does not challenge that Gestalt principles often have a generative interpretation.

if they form contiguous lines) can be understood as appealing to an underlying generative process: the next element is added relative to the previous element, keeping the previous directions of insertion with high probability or changing direction with lower probability. An example of the connection between bottom-up and generative principles can be found in auditory scene analysis (Bregman 1990). The idea of auditory streams is motivated by underlying sound sources. Acoustic events are grouped together based on the assumption that they were *generated* by the same sound source. However, the mechanisms that give rise to stream interpretations were suggested to be based on bottom-up Gestalt principles. Conversely, the Generative Theory of Tonal Music (Lerdahl and Jackendoff 1983) is (despite its name) a bottom-up theory rather than a generative model, with preference rules describing how interpretations are obtained from the observed surface (see Section 1.1.2). Only later, researchers have adapted this bottom-up model into a generative form (Nakamura et al. 2016).

1.2.3 Interpretation and Expectation

The second major framework for perception of musical structure is expectation (Meyer 1956; Narmour 1990; Huron 2006; Pearce and Wiggins 2012). The expectation framework describes the experience of a music in time rather than from the abstract, atemporal point of view of interpretation. At a given point of time, a listener has already perceived past musical events but not yet future events. Based on the previous events and long-term musical knowledge, the new musical events might be expected or surprising. Similarly, the current musical context can give rise to very strong expectations (e.g., right before the end of a cadence), or the listener might be rather uncertain about the continuation of the piece (e.g., after a cadence, see Sears, Pearce, et al. 2018). All of these phenomena contribute to the listener's experience, and all of them rely on *predicting* (or forming *expectations* about) future events.

Expectation and interpretation focus on different aspects of musical experience. Interpretation is generally applied retro- instead of prospectively and does not need to happen in the moment, but can change the understanding of a piece over time and through reflection. Interpretation is more concerned with “making sense” of a piece while expectation describes the (immediate and instantaneous) dynamics of experience. Nevertheless, these two types of experience are related and often interact, as shown in Figure 1.3 (see also Rohrmeier and Koelsch 2012). For one, both deal with uncertainty, although this uncertainty has different sources: In the case of interpretation, the latent structure that explains the piece is unknown because it is generally unobservable. In the case of prediction, the future events are unobserved because they are not accessible *yet*. Furthermore, interpreting the past events yields useful information for future events. For

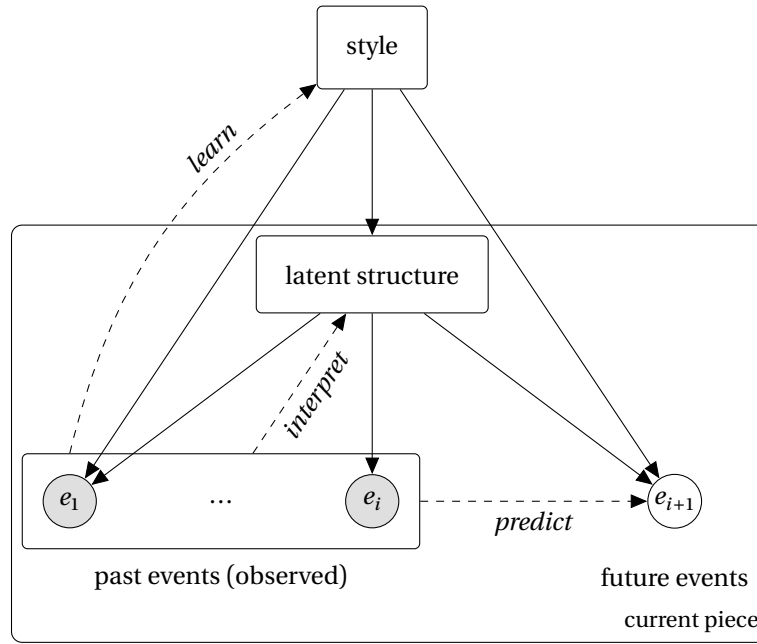


Figure 1.3 – The schematic relation between prediction and interpretation.

example, when a listener infers from the past events that the piece is approaching the end of a cadence, their expectations will change accordingly. Conversely, experiences related to expectation can play a role in the formation of interpretative concepts such as deceptive cadences. In addition, both perspectives share the problem of generalizing over different pieces by learning and applying style knowledge.

The relation between interpretation and expectation can be expressed in probabilistic terms. From a purely predictive perspective, expectation can be expressed as a probability distribution of the next event given all previous (observed) events:

$$P(e_{i+1} | e_1, \dots, e_i). \quad (1.4)$$

When combined with the interpretation perspective, the same distribution can be expressed as a marginalization over latent interpretations I :¹⁰

$$P(e_{i+1} | e_1, \dots, e_i) = \sum_I P(e_{i+1} | I) \cdot P(I | e_1, \dots, e_i). \quad (1.5)$$

In general cognitive science, this integration of prediction and interpretation is the foundation of *predictive processing* (Clark 2013), which is strongly linked to Bayesian

¹⁰In Equation 1.5, no direct dependencies between the surface symbols are assumed, i.e., e_{i+1} depends *only* on I . However, by choosing an appropriate structure for I (which, for example, remembers the relevant information about past events) these event-to-event dependencies can be included in the distribution $P(e_{i+1} | I)$. Generally, however, the interpretation serves as a *compression* of the relevant information in the past events, so that prediction becomes simpler and generalizes over similar (but not identical) contexts.

inference: Abstract beliefs about the state of the world (e.g., an object moving from left to right in front of observer) generate predictions about the next sensory input (observing the object further to the right) while mismatch between prediction and actual input (observing the object further left than expected) lead to updated beliefs (the object stopped or slowed down). Thus, instead of predicting the next event directly from the past events (as in Equation 1.4), an abstract, compressed representation of the world state is used (as in Equation 1.5).

In music perception, this integrated view is applied less frequently, at least explicitly. For example, Herff et al. (2021) have found that using a grammatical model of latent harmonic structure is useful for modeling listeners' expectations about the number of remaining chords in an incomplete harmonic progression. The more common approach, which goes back to Narmour's implication-realization model (Narmour 1990; Narmour 1992), models expectations directly on the basis of past events, as in a Markov model. In particular, the multiple-viewpoint model and its later successor IDyOM (Conklin and Witten 1995; Pearce 2005; Pearce and Wiggins 2012) describe expectations that are formed based on a combination of features of past events within the current piece as well as general regularities learned over many pieces. While the features that are used to predict the next event are deterministic, the set of features that is used is chosen based on the training data. Thus, the model does not *interpret* the already observed events in the full, probabilistic sense (although complex features could be seen as heuristic and deterministic approximations of interpretations) but it *learns* to focus on stylistically relevant features across different pieces, just like interpretative models can infer global properties of a style. The used set of features is selected based on optimal cross entropy of the observed sequences, which is equivalent to the likelihood principle.¹¹

Implicitly, expectation and interpretation are combined in some psychological paradigms. For example, probe tone experiments (Krumhansl 1990a) reveals an abstract interpretation of a presented context (the context's key) through its effect on subsequently presented tones. While this does not explicitly refer to prediction, it shows how the perception of future events is influenced by an interpretation of past events. The combination of expectation and interpretation can also arise implicitly in computational models that do not directly aim at modeling listeners' expectations but still have a sequential form, such as hidden Markov models (HMMs) and recurrent neural networks (RNNs, see Section 1.1.2). HMMs directly incorporate an interpretation aspect in the model form since they assume a sequence of latent states that are probabilistically con-

¹¹In Bayesian inference, learning of global properties θ is not based on the likelihood $P(\vec{O} | \theta)$, but on the posterior $P(\theta | \vec{O})$, which is proportional to the product of likelihood and prior ($P(\theta)$). However, when a flat prior is assumed that assigns equal probability to all possible θ , then the maximum-likelihood and maximum-a-posteriori estimate are equivalent.

nected to the surface events. In particular, they assume no direct dependency between the surface events, so prediction in HMMs is entirely based on inference over the latent states. In RNNs, a hidden state is used as well, but it is computed deterministically from the current observation and the previous state (Goodfellow et al. 2016). Both HMMs and RNNs can be trained on a set of sequences, which can be understood as inference to the stylistic properties of the dataset.

1.3 Outlook

1.3.1 Goals and Methods

The goal of this thesis is to develop a model of basic tonal structures as perceived by an abstract, idealized listener. Such a model abstracts from human understanding of music in two major ways: First, interpretations are conceptualized as complete and consistent derivations of a piece. While humans can generally not be expected to have such a clear and unified understanding of a piece, complete interpretations can be considered the ideal case that a listener would converge to given enough time and experience, and thus constitute an important foundation. In this sense, the goal is to model human *competence* rather than *performance* (Chomsky 1965). Second, the model focuses on a specific set of structural aspects (simultaneity, sequentiality, functional dependency) while other aspects such as rhythm and meter, form, or motivic material, as well as other modes of musical experience (e.g., acoustic, emotion, association, metaphor and symbolism) are ignored.

The methodological approach of this project is based on computational modeling. Broadly speaking, models are abstract descriptions of a certain domain in terms of entities and their relations and interactions. They are foremost descriptive tools: While models can be used to describe reality (or hypotheses about reality), they are, on their own, neutral descriptions until applied for a certain purpose, which can be encoding theories or hypotheses, but also communication and understanding, or simulation and prediction. Because of this versatility, models can serve as a useful bridge between music cognition and music theory, two fields that do not necessarily share the same goals, but might very well share concepts.

One aim of this thesis is to derive and justify a model in a systematic way. The modeling decisions taken here are thus based on a number of design principles: First, the model is *generative*, i.e., it describes a process that produces the observable objects of interest (here: the surface notes of a piece) and reveals the latent relations and entities

of interest. Second, it is *combinatoric*, i.e., it uses a small set of orthogonal primitives. Complexity and variety in the produced surface forms arises from combination of these primitives. Both the generative and the combinatoric principle are shared with grammatical approaches in computational linguistics (Chomsky 1965). As a third criterion, the operations and relations used by the model should have a musical interpretation. Structure is not just assumed for the sake of structure but to carry meaning and provide an “understanding” of a piece (Marsden 2005; Rohrmeier 2020a; Rohrmeier 2020b). Finally, this model aims at covering structure in the *general* case instead of the typical case. This is important since music theory is often concerned with simple or typical cases. When a complex or rare case is not covered by a theory, it is either regarded as an exception or it is left to the theorist’s intuition to extend the narrow explicit rules to the general case. While exceptions can be acceptable when caused by a phenomenon or principle that lies outside the scope of the model or theoretical framework, all cases that are believed to be captured by the principles of interest should be covered by the model. This is particularly challenging when applying the principles of voice-leading in the context of implied and free polyphony, as explained in Section 1.1.3.¹² Shifting the burden of generalization from the analyst to the model can require the reframing and reorganization of music-theoretical concepts that have traditionally been formulated in restricted settings, such as voice-leading rules in species counterpoint or part writing (Aldwell and Cadwallader 2018; Huron 2016).

1.3.2 Thesis Outline

This thesis is structured in three parts. Part I introduces the problem, the research goal, and the approach of the overall project. Chapter 2 gives an overview of computational modeling for music theory and corpus research, in particular in the form of Bayesian generative models.

Part II approaches the problem of tonal structure through three case studies, each of which addresses a separate structural aspect in isolation: In Chapter 3, local features and heuristics are used to recognize voice-leading schemata (as latent entities) from a score in a “bottom-up” fashion, by selecting and examining the notes that instantiate the schema in the score. Chapter 4, in turn, explores the relation between harmonic types and the ornaments they typically generate, i.e., the notes that do *not* instantiate them on the surface (non-chord tones). In Chapter 5, finally, functional relations between surface notes are modeled using a formal grammar on the example of melodies in North Indian classical music (and potentially other styles with similar modal properties).

¹²This problem, extending the principles of counterpoint from strict polyphony to free composition, was one of the motivations for Schenker to develop his general theory (Schenker 1979).

Part III then turns to an integrated model of tonal structure called “protovoice model” that addresses the problems laid out in Section 1.1: coordinating basic tonal relations of simultaneity, sequentiality, and functional dependency, and proposing a principled solution to the voice problem for free polyphony. The model characterizes strong interpretations of the tonal structure of the musical surface and its relation to latent entities. Chapter 6 provides a formal definition of the model and presents a parsing algorithm for obtaining protovoice interpretations. Chapter 7 demonstrates on a number of examples how protovoice derivations can express analytical intuitions and capture a wide range of theoretical phenomena related to implied and free polyphony as well as the surface realization of latent entities. Chapter 8 briefly describes a family of tools for creating and exploring derivations, as well as file formats for storage and processing. In Chapter 9, a probabilistic version of the protovoice model is presented, closing the circle to Bayesian modeling, learning, and inference, and thus to computational models of cognition.

2 Model-Based Music Theory¹

2.1 Introduction: Music Theory and Corpus Research

One of the core aims of music theory is to characterize musical structures as they occur across cultures, historical periods, and styles, and to reveal the general principles underlying these structures. More specifically, music theory provides models of such structural domains as harmony, counterpoint, rhythm, meter, form, timbre, etc. and the interplay between these domains. A branch of music theory explores musical structures in their ecological environment, which involves considering the cultural, social, institutional etc. contexts in which music is produced, performed, and perceived.

For centuries, this endeavour involved either scientific speculation or the study of musical “data” in a broad sense (e.g., Rameau 1722; Koch 1782, 1787, 1793). Such early “empirical” studies date back to the eighteenth century: treatises of composition, although prescriptive in nature, often invoke notions of frequency and typicality, which are loosely based on observations of the repertoire of the time.² Empirically conceived in this informal sense are also what one might call “repertoire studies,” that engage with specific corpora (e.g., of Palestrina’s vocal music, Bach chorales, Haydn’s Symphonies, Radiohead songs, Turkish folk music, etc.³); the same applies to music theory textbooks that draw on “intuitive statistics” when describing compositional systems, practices, and styles (e.g., Piston 1987; Aldwell and Cadwallader 2018; Gauldin 1997; Laitz 2016).⁴

However, both repertoire studies and theory textbooks typically suffer from a lack of

¹This chapter has been submitted as:

C. Finkensiep, M. Neuwirth, and M. Rohrmeier (submitted). “Music Theory and Model-Driven Corpus Research”. In: *Oxford Handbook of Corpus Studies in Music*. Oxford: Oxford University Press

CF is the main author of sections 2.3 to 2.5. Section 2.1 was written by MN and MR. Section 2.2 was written by all three authors.

²For instance, Heinrich Christoph Koch in the third volume of his *Versuch* (1793, 307ff) differentiates between formal options for the structure of the first “Hauptperiode” of a sonata-form movement’s second part (what later came to be known as the “development”) based on their frequency of occurrence in compositional practice.

³E.g., Jeppesen 1960; McHose 1947; Haimo 1995; Osborn 2017

⁴See Neuwirth and Rohrmeier 2016.

transparency in terms of the samples considered and the way they arrive at general observations and insights (e.g., Neuwirth and Rohrmeier 2016). These two shortcomings are addressed by the more recent musical corpus studies, whose core goal is to examine musical data available in machine-readable formats, adopting a wide range of quantitative methods (including sampling strategies).⁵ Corpus studies benefit previous music theoretical research and repertoire studies by providing an empirical underpinning of arm-chair hypotheses and claims. On the other hand, corpus studies require music theoretical foundation in order to provide insights with musically meaningful and interpretable statistical results. In both ways, model-driven statistical corpus research constitutes a central avenue for novel empirical music research, as it offers models establishing an explicit mapping between theoretical concepts and observed or computed quantities.

2.2 Model-Driven Music Research

Model-driven music research may be characterized as the analytic activity by which abstract concepts are inferred based on observations of musical objects (e.g., Temperley 2007a; Temperley 2009; Abdallah, N. E. Gold, and Marsden 2016). Claims about the relation between objects on the “surface” (i.e., the level of observation)⁶ and abstract concepts as well as, for instance, about their frequencies of occurrence can be validated using the methods available in corpus research. For corpus research and music theory (or analysis) to efficiently interact, theories need to be sufficiently precise and formalizable; in other words, they require a model. In the most general sense, a model is a description of a particular segment of the world in terms of *entities* and their *relationships*.⁷ Since the “true” structure of the world is generally not known, models typically express a set assumptions about a segment of the world. In a formal model, entities and relations are expressed in a precise way using the language of mathematics.

Models inevitably involve abstraction and simplification: A model specifies which aspects of the world are described, and which are not. Aspects deemed irrelevant for the structures of interest (i.e., the modeled world) are necessarily ignored. This can happen either by *abstraction* or by *approximation*. Abstraction restricts the model to refer

⁵One of the earliest instances of corpus study in the pre-computer age is Budge 1943.

⁶We use the term “surface” to describe the musical objects that are explicitly described in a corpus and therefore can be considered to be “directly observed” in the context of corpus studies. In music cognition, the term is also used to denote a level of a listener’s musical perception, which is a more complex issue (Cambouropoulos 2010).

⁷For further literature, see for instance Tenenbaum et al. 2011; Norvig 2017; Page 2018; Flanders and Jannidis 2018; Winn 2020.

only to some aspects of the real world while ignoring others, but without introducing contradictions between the world and the model.⁸ Approximation, in contrast, trades a simpler model for slight inconsistencies with the real world: while the model predictions are technically wrong,⁹ the error is deemed sufficiently small.¹⁰ Most models make use of both abstraction and approximation. By enriching the level of detail contained in a model, it gradually approximates (some of) the richness and fuzziness of the world. Abstraction does not only restrict the scope of a model, it is also an important internal property of models: In the form of generalization, it allows one to formulate general insights that transcend observations specific to individual works or corpora. In this sense, music theories represent a form of *compression* (e.g., Abdallah, N. E. Gold, and Marsden 2016): abstract elements and relations are assumed to describe musical structures that are common to a well-defined set of otherwise different passages or pieces. Instead of making the same statement about every passage or piece individually, a theory makes one general statement that applies to all of them.

Models are often (but not necessarily) generative (see e.g., Chomsky 1965; MacKay 2003; Winn 2020). Generative models express the relationship between observed and latent entities as a process that “generates” the observations based on the latent entities. Such a model architecture is common, because it captures how latent entities give rise to surface observations in an intuitive way (a process is easy to imagine). Note that while the idea of a “generative process” has a causal connotation, the relationship it expresses need not be causal. For example, a set of surface objects being connected to each other through a latent entity (e.g., a set of pitches that form a chord) can often be expressed as the latent entity generating the surface objects. However, this does not mean that the surface objects are *actually* generated by the latent entity (a composer need not think of the chord before deciding to write the notes), it is just a convenient way of expressing the relation. Moreover, generative models are attractive, because they can provide *explanations* of the observed entities rather than mere descriptions (e.g., notes occurring together *because* they form a chord). Regarding their modelling philosophies, one may attest a strong affinity between music theory and the framework of generative theories, as abstract music-theoretical concepts are often used to “explain” phenomena observed on the musical surface or in the perception of the listener. Generally, analysis

⁸An example of abstraction is a mathematical description of a computer, as opposed to a detailed physical description of a computer. While the physical description would include the exact atoms the computer is made of, the mathematical description only specifies that the computer has certain functional capabilities. While the two descriptions focus on very different aspects, there is no contradiction between them.

⁹This view is nicely captured in Box’s “All models are wrong but some are useful” (Box 1976).

¹⁰An example of approximation is Newtonian mechanics. While it is known to be inaccurate in general (in particular with velocities that approach the speed of light), it is a very useful model for mechanics at lower speeds.

can be understood as reasoning about how latent structures or entities give rise to the musical surface. Latent structures may range from rather concrete (e.g., chords) to very abstract concepts (such as ideas or intentions). The field of generative modelling is formally and scientifically directly linked with Bayesian modelling, as will be discussed in Section 2.4.

Some examples shall illustrate how ideas from music theory relate to these considerations about modelling. For instance, theories of tonal harmony are based on functional categories (e.g., dominant, predominant, tonic), which abstract over classes of concrete tones (here over classes of chords, which in turn are classes over pitch class distributions, which in turn form classes of different single tones), in order to characterize harmonic progressions by means of relations between categories (e.g., predominant \rightarrow dominant). Statements about sequences of functional categories generalize over the chord or note level.

Schenkerian theory employs abstract concepts such as Züge, couplings, neighbor notes, horizontalization, unfolding, etc. and explains their relations (for example neighbor notes within a Zug) by means of hierarchical-recursive constructions (e.g., Cadwallader and Gagné 2011; Yust 2015b). These constructions are assumed to be general, being applicable to a wide range of pieces as well as different levels of reduction within the same piece. Attempts at a precise formal model are rare, however (e.g., Frankel et al. 1978; Marsden 2005; Kirilin and Utgoff 2008; Kirilin and Jensen 2011).

Similarly, musical schema theory takes various voice-leading patterns as elements, modelling the relationships of structurally relevant and irrelevant (ornamental) tones as well as the sequence of various schemata (R. Gjerdingen 2007). Like chords, schemata generalize over different note configurations on the musical surface that can be considered instances of a schema. In that sense, a sequence of schemata is an abstraction from the note level, just like a chord progression (e.g., Sears and Widmer 2020; Finkensiep, Déguernel, et al. 2020).

The above theories about structural properties of music can all be turned into formal models by explicitly specifying the assumed objects and relations in a formal (i.e., mathematical) way. This has a series of advantages: Exact formalization enables deeper insights into as well as uncovering problems (such as internal inconsistencies, and conceptual or logical gaps) with these theories.¹¹ At the same time, the statements made by these

¹¹For example, only the attempt by Hamanaka et al. to formalize the Generative Theory of Tonal Music (GTTM) by Lerdahl and Jackendoff (1983) exactly revealed how many aspects of the GTTM remain vague and under-specified (Hamanaka et al. 2016). The same can be observed with attempts at precisely formalizing Heinrich Schenker's theory (e.g., Marsden 2010; Kirilin and Yust 2016), and schema theory (e.g., Sears, Arzt, et al. 2017; Sears and Widmer 2020; Finkensiep, Déguernel, et al. 2020).

theories about structural relationships can be examined and validated empirically.

2.3 Modelling Decisions in Musical Corpus Studies

In the previous section, models were introduced as consisting of entities and relationships that reflect some aspect of interest of the world. To make this idea more specific, we will now look at examples of musical corpus studies from the perspective of *modelling decisions*, i.e., the questions and options that arise when defining a model. We will focus on models related to harmonic syntax, i.e., the description of typical chord progressions in a style, though the general principles outlined here apply to any domain.

2.3.1 Entities and Relations

One primary question in corpus research concerns which objects are represented in the corpus, and what are the directly observable relations between them. In the example of harmonic syntax, this is usually a set of chord progressions, i.e., sequences of chords in some representation. Some entities and relations follow directly from this structure: There are two types of objects, namely chords and progressions of chords (that can stand for phrases or whole pieces). Progressions are related to chords in that progressions combine chords into sequences. The chords involved in a single progression are related to each other through their sequential organization: they have, for example, a temporal order, direct neighbors, a set of predecessors, and a set of successors. Other relations are less obvious and depend on the exact representation of chords. For example, chords that are represented as pairs of a root and a chord type can be compared through the intervallic distance between their roots (which is another entity). If, however, chords are represented in terms of figured bass (e.g., “voice-leading types” in Sears, Arzt, et al. 2017), no explicit root is assigned, so the aforementioned root relation cannot be expressed. Instead, these chords can be related through their bass notes, for example. Even when a root is given, different relations are implied by a notation that uses absolute roots than by a notation in terms of scale degrees (Roman numerals), which implies relations that are independent of the local key. Finally, different representations of a chord *type* (e.g., according to Kostka-Payne, ABC, or Roman Text; see Temperley 2011; Moss, Neuwirth, Harasim, et al. 2019; Gotham et al. 2019; Harrison and Pearce 2020b) both contain different information and imply different relations, for example, which chords are considered to belong to the same type. Generally speaking, the representation of the information contained in the corpus determines the relations that can be expressed and used in the model.

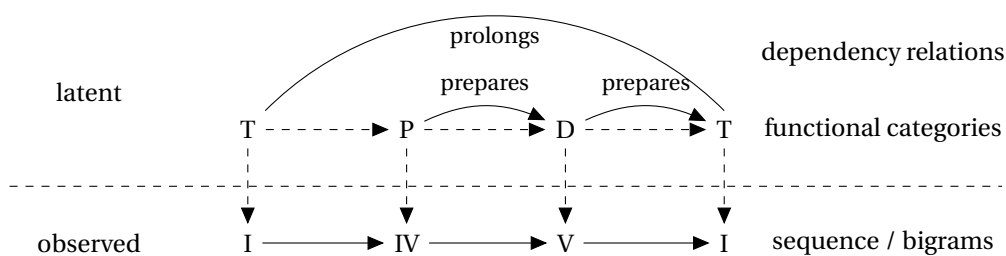


Figure 2.1 – Latent entities and relations in different models of harmonic sequences. Bigrams are directly represented in the observed data, while functional categories and dependency relations must be inferred.

Reflecting on the way musical information is represented is especially important when a project involves the creation of corpora by, e.g., transcription or annotation. However, even using a given corpus requires decisions about how to use the encoded information, whether to ignore some of it, or whether to transform it into a different representation. In making these decisions, it can be helpful to consider what would be the *ideal* representation for the goals of the project. Even if this ideal representation is not attainable, it can also help identify limitations of the given representation, where information that is necessary to answer a given question is not given the corpus (e.g., the roots of chords given in figured bass notation). The choice of representation necessarily involves abstraction and approximation. For example, representing chords as labels is an abstraction over the actual note configurations that they stand for. Similarly, assuming that the harmonies in a piece form a single continuous sequence is an approximation (there might be nonharmonic or polyharmonic sections). Which approximations and abstractions are appropriate depends on the research goals and should be explicitly discussed.

While the basic entities and their relations are usually given by the data in the corpus, the objects of interest are usually *latent*, i.e., they are not directly contained in the data and must be inferred in some way. In the case of harmonic syntax, the objects of interest are generally the relations between chords that are relevant for a certain musical style as well as their quantitative properties. A simple model of the harmonic syntax of a style may, for instance, look at the direct transitions from one chord to the next. While the fundamental relation (the *chord transition*) is given directly in the corpus (and thus is observed, not latent), the *typicality* of different transitions (or types of transitions) in the style must be inferred. Such a model that looks at pairs of chords is known as a *bigram model* (or, more generally, *n-gram model*).¹² One simple way to assess the typicality of transitions

¹²Bigram models are very common in harmonic and melodic modelling, see for example de Clercq and Temperley 2011; Moss, Neuwirth, Harasim, et al. 2019.

is to look at the number of occurrences of each bigram. Alternatively, the transitions between chords may be understood probabilistically, and typicality can be explored by comparing the probabilities of different transitions from the same chord.¹³ In the latter model, which is known as a *first-order Markov model* (MM), the probabilities are latent entities (the “true” probabilities of the style are unknown) and need to be inferred from the data.

More complex models of harmonic syntax rely more extensively on latent relations and entities. In functional analysis, for example, chords are interpreted as representatives of a *functional category* (such as tonic, dominant, and predominant). The functional category of a chord is something that is generally not known with certainty, and must be inferred. In a model proposed by C. W. White and Quinn (2018), harmonic function is represented as a latent variable for each chord that is connected to the observed chords through a probabilistic generative process (Figure 2.1): The (latent) functions are generated first according to a Markov model (i.e., each function probabilistically depends on its direct predecessor). Then the surface chords are chosen probabilistically based on the function. Note how this architecture, which is known as a *hidden Markov model* (HMM), combines observed entities (chords) with latent entities (functions) and specifies the relations between them through conditional probability distributions.

However, relations in harmonic sequences need not be restricted to direct neighbors. Instead, *dependency relations* (such as preparation and prolongation) can also exist between non-adjacent chords over large spans of time (Rohrmeier 2011; Rohrmeier 2020a; Rohrmeier and Neuwirth 2015). Such non-local relationships can be expressed with grammar models such as context-free grammars (CFG). Similar to HMMs, CFGs define a (potentially probabilistic) generative process that derives a harmonic sequence from a set of production rules (Abdallah, N. E. Gold, and Marsden 2016; de Haas et al. 2009; Harasim, Rohrmeier, et al. 2018; Harasim, O’Donnell, et al. 2019) that reflect dependency relations. In this case, the dependency relations are the latent objects of interest, and they are related to the observed entities via the concept of derivation.

In all three cases, the objects of interest determine the structure of the model. Dependency relations have different properties than functional categories and chord transitions, and the three models reflect these differences. While some model architectures are more convenient than others (bigrams are arguably easier to compute than derivations from a grammar), they also come with different limitations on the interpretability of their relations (direct transitions cannot capture long-distance relations). Crucially, the choice of model architecture is informed by the objects of interest, not the other way

¹³As will be shown in Section 2.4, the two approaches are formally related and counting the absolute occurrences has a probabilistic interpretation as well.

around.

2.3.2 Analysis

While a model can be a contribution on its own (e.g., by formalizing intuitive theories), it is often used as a means of answering an empirical question, usually by performing some kind of statistical analysis. In relation to the model, two types of analysis can be distinguished that might be called *external* and *internal* analysis. External analysis uses observations obtained from applying the model to the corpus and performs a separate statistical analysis. For example, a corpus of harmonic sequences can be dissected into bigrams, while statistics on these bigrams are used to analyze typical patterns (e.g., de Clercq and Temperley 2011; Hedges and Rohrmeier 2011; Moss, Souza, et al. 2020) or properties such as “directedness” (e.g., Rohrmeier and Cross 2008; Moss, Neuwirth, Harasim, et al. 2019). In order to answer a specific question, statistics can be combined with a hypothesis test. It is important to note that both computing statistics and performing hypothesis tests requires assuming a relation between the statistical quantity that is computed and the model observations. Thus, statistics and tests can be thought of as defining implicit models that extend the explicit model on which a study is based. An example of such an implicit model is using relative frequencies to characterize quantitative properties of a bigram model: For a given antecedent chord it seems to be useful to look at the relative frequencies of the consequent chord. However, knowing how to compute this statistic (by counting and normalizing) does not tell us why the statistic is meaningful. Instead, it is implicitly assumed that the consequent chord *depends* on the antecedent chord, that this dependency is captured by a conditional probability distribution $p(c | a)$, and that the relative frequencies of consequent chords are good estimates of this distribution.

Internal analysis relies on the relation between statistics and observations being *intrinsic parts* of the model. In a Markov model, for example, the transition probabilities (which were implicitly assumed in the external analysis of the bigram model) are represented as explicit entities. In such a case, analysis can be seen as a form of inference: what can we conclude about the latent parameters of the model from the observations given in the corpus? The model expresses the assumed relation between the latent parameters and other (latent or observed) entities. Often, internal and external analysis approaches lead to similar or even the same models. For example, relative frequencies can be seen as a *maximum-likelihood estimate* of the true transition probabilities (see Section 2.4.5). The difference between the two approaches is rather one of perspective and of making statistical relations explicit and interpretable. The central framework for systematically approaching internal analysis is *Bayesian inference*, which is discussed in Section 2.4.

2.3.3 Modelling Decisions as Assumptions

Modelling decisions are sometimes characterized as *assumptions*. “Assumption” here does not refer to assuming that some individual fact is the case (e.g., that a certain harmonic annotation is correct and reflects the “true” chord), but rather that the perspective that is taken in the model is appropriate for a particular question. For instance, a piece of music is not completely characterized by a sequence of chords. However, if the harmonic properties of the piece (or a set of pieces) is of interest, then the perspective on a piece as a sequence of chords is a sensible abstraction. Making such a model assumption (i.e. “a piece can be represented by a sequence of chords”) does not mean believing that pieces are, in fact, sequences of chords, or even that this is generally the best perspective. In addition, the set of assumptions related to a model go beyond the internal structure of the model. Since corpus studies are often motivated by abstract concepts such as “style,” adopting a certain model structure (e.g., bigrams) or a certain type of entity (e.g., chord sequences) means assuming that these are related to the high-level concept in the first place.

2.4 Probabilistic Modelling and Bayesian Inference

How can a model-based research strategy be implemented practically? This section aims to give an overview of Bayesian modelling, a framework that is well-suited for model-based quantitative research for three reasons: a) it requires an explicit formulation of the model as a probability distribution, which facilitates a clear and explicit expression of the modelling assumptions; b) it constitutes a systematic way of reasoning under uncertainty, which is a common property of models that involve latent entities; and c) it comes with a generic scheme for inference within and between models, which provides a principled way of deriving statistical analyses from the model (i.e., internal analysis). All three of these aspects apply to corpus research, which involves inferring latent properties (e.g., statistical quantities) from a set of observations, the corpus (for similar arguments for using Bayesian modelling in music research, see Temperley 2007a; Abdallah, N. E. Gold, and Marsden 2016). The following subsections discuss how models can be formulated in the Bayesian framework (Section 2.4.1), how probabilistic inference provides a generic mechanism for analysis (Section 2.4.2), provide an overview over inference methods (Section 2.4.3) and probabilistic programming (Section 2.4.4), and discuss the implications of the Bayesian approach for music research (Section 2.4.5). For more comprehensive introductions to Bayesian modelling, see Winn 2020, Murphy 2012, MacKay 2003, and Jaynes 2003. More detailed discussions of probabilistic programming can be found in N. D. Goodman, Tenenbaum, et al. 2016; and van de Meent et al. 2018.

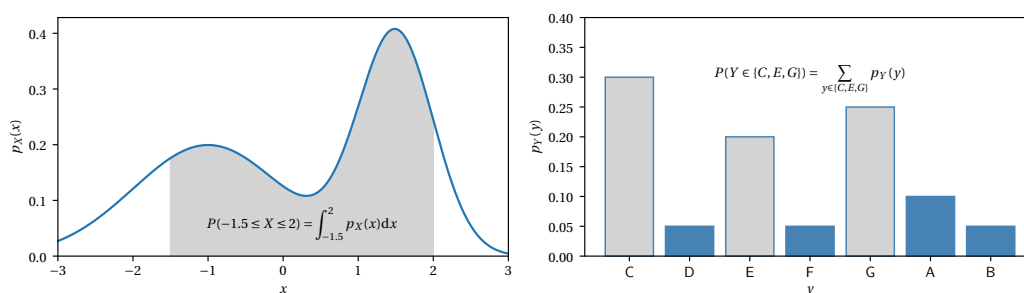


Figure 2.2 – Left: A PDF of a continuous random variable X . The probability of X being within a certain range is given by the integral of the PDF over that range. The integral over the full domain of X is 1. Right: A PMF of a discrete random variable Y . The probability of Y being in some set is the sum of the PMF over the elements of that set. The sum over all possible values of Y is 1.

2.4.1 Expressing Models as Probability Distributions

The central question of Bayesian inference is: What can be learned from entities that are observed about entities that are not (or cannot be) directly observed? For instance, given a corpus of chord sequences (observed, because given by the data), what can we learn about the (unobserved) rules and regularities that give rise to these sequences? Such an inference is only possible if the relationship between observed and unobserved entities is specified. This relationship is typically not deterministic: for a given set of observations, several values are possible for unobserved entities. This uncertainty is expressed through a *probability distribution* over observed and unobserved *random variables* (which in turn represent observed and unobserved *entities* of the model, respectively).¹⁴ Let X denote the observed and Z the unobserved variables. Then the model is expressed as the *joint distribution* over X and Z . Intuitively, it describes the probability (i.e., the plausibility) of every possible instantiation of X and Z .

A probability distribution over some *continuous* random variable X is represented by *probability density function* (PDF) $p_X(x)$ over the different values that X can take (Figure 2.2). The probability of X taking a value in a certain range is given by the integral of the PDF over that range:

$$P(a \leq X \leq b) = \int_a^b p_X(x) dx. \quad (2.1)$$

Note that the probability of X taking a specific value is 0, since the corresponding integral

¹⁴Mathematical probabilities can be interpreted in different ways, for example as a quantification of chance, but also as a quantification of plausibility or degree of belief in a statement. The latter interpretation is called the Bayesian interpretation of probability. It can be argued that probability theory is a formalization of rational inference under uncertainty (Cox 1946).

is empty. Yet, it can still be useful to compare the value of the PDF at different individual points. If X is a *discrete* random variable, the integral is replaced by a sum, so the probability of X being in some set S becomes

$$P(X \in S) = \sum_{x \in S} p_X(x). \quad (2.2)$$

Since a sum over a single value is not empty, $P(X = x) = p_X(x)$ in the discrete case, and $p_X(x)$ is called a *probability mass function* (PMF).¹⁵ When several random variables are involved, the *joint distribution* is given by a function ranging over all random variables. In this case, a probability is obtained by integrating over all continuous and summing over all discrete variables.

To understand how the joint distribution encodes the complete model, it is instructive to look at an example.¹⁶ Consider the problem of inferring a chord label based on a set of observed notes. In this case, the chord label is the unobserved variable Z while the notes are the observation X . The joint distribution encodes the assumptions that we make about the relationship between chord labels and notes by specifying the plausibility of a set of notes co-occurring with a chord label, e.g. $p_{X,Z}([C, C, E, G, D], F\sharp^{7\sharp 11})$. In this example, one may assign a low plausibility to the co-occurrence of the notes and the chord label.

Expressing a model directly as a joint distribution is not always convenient. For example, instead of expressing the plausibility of a co-occurrence of a set of notes and a label (relative to other co-occurrences), it may be more straightforward to express the plausibility of observing a set of notes in relation to a *given* chord label. For example, we could say that it is more likely to observe the notes $[F\sharp, A\sharp, B\sharp, E]$ under an $F\sharp^{7\sharp 11}$ chord than the notes $[C, C, E, G, D]$. In probabilistic terms, this relationship is expressed as the conditional probability $P(X | Z)$, and it corresponds to a generative process, by which the notes are “generated” from the given chord label. The joint distribution is related to the conditional probability distribution via the equation $p(a, b) = p(b | a) \cdot p(a)$, so we can obtain the joint distribution via $p(x, z) = p(x | z) \cdot p(z)$. Since this way of *factorizing* the joint distribution is very common, the two factors have names: $p(x | z)$ is called *likelihood* and $p(z)$ is called *prior distribution*, since it encodes prior knowledge about the unobserved variable Z . Importantly, this factorization may be understood as corresponding to a generative process: in a first step, a chord label z is chosen with probability $p(z)$, independently of the notes. In a second step, the collection of notes x is chosen with probability $p(x | z)$, i.e., based on the knowledge that the chord label is z .

¹⁵When it is clear from the context which random variables the density refers to, they are omitted. For example, instead of $p_{X,Z}(x, z)$, we write $p(x, z)$.

¹⁶The two models of chords used as examples might be compared to the models of key given by Temperley (2002; 2009) and to the chord model of Rhodes et al. (2009).

The assumptions of the model are encoded in the specific prior and likelihood distributions. In our example, the prior is a distribution over a finite set of values (all chord labels), so the most generic choice for the prior is a *categorical distribution*, in which every possible value (i.e., every possible chord label) is directly assigned a probability. The likelihood generates a number of notes from the chosen chord, so we might choose a *multinomial distribution* for the likelihood, which corresponds to repeatedly drawing notes from a categorical distribution and collecting them irrespective of the order in which they were sampled.¹⁷ The parameters of this multinomial distribution depend on the chord label that is sampled first, so that different chords generate different notes. We can summarize the model in a notation that reflects the generative process behind the distribution:

$$\begin{aligned} Z &\sim \text{Categorical}(\lambda) \\ X | Z &\sim \text{Multinomial}(v_z), \end{aligned} \tag{2.3}$$

where λ is the parameter vector for the prior distribution (i.e., the probabilities of the chord labels) and v_z is the parameter vector for the likelihood that corresponds to the chord label z .

A problem with the above model is that we might not want to specify the parameters λ and v manually. Instead, as empirical corpus researchers, we could be interested in assuming a certain *type* of relationship while inferring the specific parameters from a corpus of annotated music, i.e., several pairs of notes and chord labels. We thus need to extend our model to include the parameters λ and v as random variables, so we write the joint distribution as $p(N, L, \lambda, v)$, where N stands for a list of note collections and L is the corresponding list of chord labels, one label for each note collection. Note that now N and L are observed variables, whereas λ and v are unobserved. We can adapt the previous model to our new assumption by making two changes: First, instead of sampling a single chord label and set of notes, we sample several label-notes pairs, which we assume to be independent and identically distributed. This assumption allows us to write the likelihood $p(N, L | \lambda, v)$ as

$$p(N, L | \lambda, v) = \prod_{i=1}^{|N|} p(N_i, L_i | \lambda, v), \tag{2.4}$$

where N_i and L_i correspond to a single notes-label pair and are distributed in the same way as in the previous model, i.e. $p(N_i, L_i | \lambda, v) = p(N_i | L_i, v)p(L_i | \lambda)$. As a second change, we need to provide a prior distribution $p(\lambda, v)$. Assuming that λ and

¹⁷Note that choosing a multinomial distribution corresponds to making very specific assumptions about how the notes depend on the chord label, and different model assumptions can be expressed through different distributions. Generally, for more information on the probability distributions involved and their properties, see MacKay 2003, ch. 23; or Bishop 2006, ch. 2.

v are independent and that all chords are independent, we can rewrite the prior as $p(\lambda, v) = p(\lambda) \cdot p(v) = p(\lambda) \cdot \prod_l p(v_l)$ and provide separate priors for λ and each v_l .¹⁸ In this case we will choose a *Dirichlet* distribution for both $p(\lambda)$ and each $p(v_l)$. The Dirichlet distribution (see, e.g., MacKay 2003, ch. 23) is defined over probability vectors (i.e., vectors that sum to 1), so drawing a sample from a Dirichlet distribution generates a probability vector such as λ or v_l , which is exactly what we need. Moreover, the Dirichlet distribution is the *conjugate prior* of both the categorical and the multinomial distribution, which means that the combination of a categorical or multinomial likelihood and a Dirichlet prior has good mathematical properties.¹⁹ Finally, we set the parameters of the Dirichlet distributions to 1, which amounts to a uniform distribution over all possible λ and v_l . We can express the full model more compactly in the following notation:

$$\begin{aligned}
 & \lambda \sim \text{Dirichlet}(1, \dots, 1) && \text{prior of the chord labels} \\
 \forall l: & \quad v_l \sim \text{Dirichlet}(1, \dots, 1) && \text{prior of the notes, one per chord label} \\
 \forall i: & \quad L_i | \lambda \sim \text{Categorical}(\lambda) && \text{chord label of each data point} \\
 & N_i | L_i, v \sim \text{Multinomial}(v_{L_i}) && \text{notes of each data point.}
 \end{aligned} \tag{2.5}$$

It is a common scheme for models of datasets to assume that the data points are independent and identically distributed (i.i.d.). In that case the model is of the form

$$p(D, \theta) = \prod_{i=1}^{|D|} p(D_i | \theta) p(\theta), \tag{2.6}$$

where D is the dataset and θ is the set of parameters that apply to all points in the dataset. In fact, the above example is of that form too, with $D = (L, N)$ and $\theta = (\lambda, v)$.²⁰

Sometimes, Bayesian models are depicted using a graphical notation (Figure 2.3). Random variables are represented by circular nodes while edges indicate the dependencies among the random variables. Nodes with no incoming edges are sampled from their

¹⁸That each chord label has an independent set of parameters is a specific modelling choice that we do not have to make. We might instead want to express that chords of the same type (but different roots) have the same parameters, but transposed to the respective root. This alternative, more advanced model can also be expressed through a corresponding probability distribution or generative process.

¹⁹To be precise, combining a likelihood with a conjugate prior results in a posterior distribution (the distribution over unobserved variables *after* taking into account the observed variables) of the same family as the prior distribution. In this example, the posterior distributions of the model parameters will also be Dirichlet distributions. This has the advantage that the posterior distribution is easy to compute and that the parameters of the prior and the posterior are often directly interpretable. Therefore, conjugate priors are an important tool for choosing priors, and it is often a good strategy to start with a likelihood that models the problem well and then selecting a prior that is conjugate to the likelihood.

²⁰However, it must be ensured that the data points can actually be assumed to be i.i.d., otherwise a different model should be chosen. For example, if we are not interested in the order of the chords, then it is reasonable to assume them to be independent. If the order matters, then subsequent chords will likely be related to each other, so they are not independent and the model should reflect that.

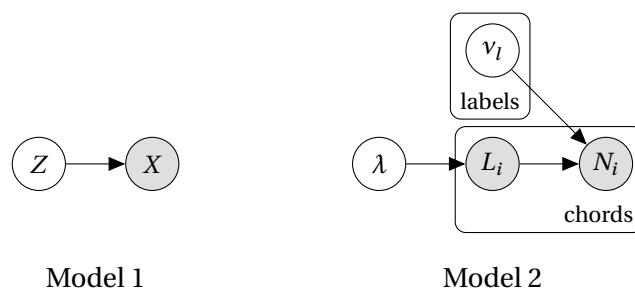


Figure 2.3 – Graphical notation of the two chord models. Nodes indicate variables, edges indicate a direct dependency, and rectangles (plates) indicate several instances of i.i.d. variables. Observed variables are shown in grey.

respective priors. Nodes with incoming edges are sampled conditioned on their “ancestors.” For example, in the graphical notation of our first model, the edge between Z and X indicates that X is sampled from the conditional distribution $p(x | z)$ while Z is sampled directly from its prior $p(z)$. Whenever a set of variables is i.i.d., we can use *plates* to write them just once. For example, our second model has one parameter vector v_l for each chord type. Since all of them are drawn from the same prior $p(v_l)$, we display v_l *once* inside a plate instead of using individual nodes for v_1, v_2 , etc. Similarly, the observed chords (L_i, N_i) are i.i.d. and can be notated using a plate as well. Note how every plate and its variables correspond to a product over the same variables in the model’s equation: The “labels” plate corresponds to the term $\prod_l p(v_l)$, i.e., the priors of each v_l . The “chords” plate corresponds to the term $\prod_i p(L_i, N_i | \lambda, v)$. The dependency of this term on λ and v is reflected by the arrows going into the “chords” plate.

This graphical notation can be useful to get an overview of the dependencies (and independencies) between a model’s random variables. However, it does not specify the exact probabilistic relations between the variables and it can be hard to read for larger models. It should therefore be accompanied by a full description of the generative process, e.g., as a sequence of sampling statements as shown above, or as a probabilistic program as explained in Section 2.4.4.

2.4.2 Types of Inference in Bayesian Models

In order to understand the different types of inference in Bayesian models, it is helpful to revisit the two fundamental operations on probability distributions: *marginalization* and *conditioning*. A marginal distribution isolates a subset of the joint’s random variables and describes their distribution irrespective of the values the other random variables take. The marginal distribution is obtained by considering all possible values for the

eliminated variables and summing (for discrete variables) or integrating (for continuous variables) over the corresponding probabilities, i.e.:

$$p_A(a) = \sum_d \int_c p_{A,D,C}(a, d, c), \quad (2.7)$$

where d denotes all possible values of the discrete variable(s) D , and c denotes all possible values of the continuous variable(s) C .

A conditional distribution also isolates a subset of the random variables, but it does so by fixing the value of the remaining variables instead of summing or integrating them out, which is notated as $A | B$ (read “ A given B ”), or $p_{A|B}(a | b)$ for the density. As we have seen above, the conditional distribution is related to the joint distribution via the marginal distribution of the variable(s) that we condition on:

$$p(a, b) = p(a | b) \cdot p(b) = p(b | a) \cdot p(a). \quad (2.8)$$

The motivating inference question in Bayesian models is what can be learned about the unobserved (or hidden, latent) variables Z from the observed variables X . This relationship can be expressed using a conditional distribution $p_{Z|X}(z | x)$, the distribution of the unobserved variables given that the observed variables take a certain value x , i.e., the observed data. As above, this conditional can be derived from the joint distribution using $p(x, z) = p(z | x) \cdot p(x)$, so

$$p(z | x) = \frac{p(x, z)}{p(x)}. \quad (2.9)$$

The marginal probability of the data $p(x)$ is called *evidence* and can (in principle) be derived from the joint distribution via $p(x) = \int_z p(x, z)$. The distribution $p(z | x)$ is called the *posterior distribution*, since it encodes the knowledge about the unobserved variables *after* considering the values of the observed variables. Its relation to the prior can be seen when splitting up the joint into prior and likelihood:

$$p(z | x) = \frac{p(x | z)p(z)}{p(x)}, \quad (2.10)$$

which is known as *Bayes' theorem*.²¹

What does the posterior express in practice? When looking at our initial example, we can see that the posterior $p(z | x)$ is the distribution over the chord label Z given that the set of notes X takes a certain value x , e.g., $p_{Z|X}(z | [C, C, E, G, D])$. This distribution

²¹Note how the semantics of the prior and the posterior are reflected in their mathematical form: the prior is the knowledge about the unobserved variables *irrespective* of the observed variables, so it is expressed as a *marginal distribution*. The posterior is the knowledge *given* the values of the observed variables, so it is expressed as a *conditional distribution*.

encodes the relative *plausibility* of each possible chord label z for the given set of notes. It is important to note that we have a *distribution of plausibility* over the possible chord labels instead of a single chord label that the model deems to be the most plausible one, as this allows the model to express its *degree of uncertainty*. We can still obtain a single chord label from the posterior, e.g., by taking the most plausible chord label, the so-called *maximum a posteriori* (MAP) estimate. Similarly, in the second example the posterior $p(\lambda, \nu | L, N)$ expresses the belief about λ and ν after observing a set of example chords with labels L and note collections N . Since λ and ν encode properties of chords (the probability of each label and the probabilities of notes under each label, respectively), the posterior reflects the belief about these properties after considering the corpus of chord instances. Again, instead of a single estimated value for the parameters, we get a distribution that reflects the relative plausibility of different parameter values. Finally, it is worth noting that a given joint distribution can have different posteriors depending on which variables are observed and which are unobserved. For example, instead of observing chords as pairs of labels and notes, we might try to learn chords in an *unsupervised* fashion, i.e., just from sets of notes without knowing the chord labels. We can express this problem using the posterior $p(\lambda, \nu, L | N)$ and it can be derived from the joint distribution just like any other conditional:

$$p(\lambda, \nu, L | N) = \frac{p(\lambda, \nu, L, N)}{p(N)}. \quad (2.11)$$

The posterior then expresses the plausibility of different combinations of label assignments for each chord and global parameter estimates.²²

The posterior is not the only inferred distribution one could be interested in. After learning the posterior distribution from a set of observations, we might want to make predictions over new observations. For example, after learning about the properties of different chords from a set of examples, we can use this knowledge to infer the labels of a new chord, for which we only know its notes. We can express this situation as $p(l' | n', L, N)$, where l' stands for the label of the new chord and n' stands for its notes. The prediction problem can be reduced to the posterior by marginalizing out (i.e., integrating) the parameters λ and ν :

$$p(l' | n', L, N) = \int_{\lambda} \int_{\nu} p(l' | n', \lambda, \nu) \cdot p(\lambda, \nu | N, L). \quad (2.12)$$

The term $p(l' | n', \lambda, \nu)$ expresses the distribution over the label to predict for a given set of notes and given parameters λ and ν , as if they were sampled from the posterior. If we ignore λ and ν for a moment, the term $p(l' | n')$ can be interpreted as a posterior itself,

²²Since the space of these combinations can be quite large, it may not be trivial to make use of it, for example to find the MAP assignment of chord labels and parameters.

namely the probability distribution over a chord label given an observed set of notes. Using Bayes' formula, it can be reduced to the corresponding likelihood and prior:

$$p(l' | n', \lambda, \nu) = \frac{p(n' | l', \lambda, \nu) \cdot p(l' | \lambda, \nu)}{p(n' | \lambda, \nu)}, \quad (2.13)$$

which we already know from our model: The prior $p(l' | \lambda, \nu)$ is the prior probability of the chord labels, a categorical with parameters ν . The likelihood is the probability of the notes given the chord label, a multinomial with parameters $\lambda_{l'}$. Note how this corresponds exactly to the posterior in our first model (Equation 2.10, with $l' = z$ and $n' = x$), except that every term is conditioned on λ and ν .

Finally, the Bayesian approach can be used to compare the plausibility of different models, e.g., as an alternative to classic hypothesis testing. Similar to how we can integrate parameters into the model to make inference about them, we can also integrate the choice of the model as a random variable in a "meta model." Let's say, we have a model m_1 over observed variables X and unobserved variables Z , i.e., the joint distribution is $p_{m_1}(x, z)$. We also have a second model m_2 over the same observed variables, which assumes a different set of unobserved variables U and is given by $p_{m_2}(x, u)$. By introducing a random variable M for the model choice, both models can be integrated into a single joint distribution $p(x, z, u, m)$. The original model distributions can be expressed as $p_{m_1}(x, z) = p(x, z | m_1)$ and $p_{m_2}(x, u) = p(x, u | m_2)$.²³ The plausibility of a model under the given observations can now be expressed as $p(m | x)$, i.e., by marginalizing over the unobserved variables. For example, the *relative* plausibility of m_1 compared to m_2 is given by

$$\frac{p(m_1 | x)}{p(m_2 | x)} = \frac{\frac{p(x|m_1) \cdot p(m_1)}{p(x)}}{\frac{p(x|m_2) \cdot p(m_2)}{p(x)}} = \frac{p(x | m_1) \cdot p(m_1)}{p(x | m_2) \cdot p(m_2)} = \frac{\int_z p(x, z | m_1) \cdot p(m_1)}{\int_u p(x, u | m_2) \cdot p(m_2)}. \quad (2.14)$$

This ratio requires a prior distribution over the models $p_M(m)$, which encodes the a priori probability of each of the models. It can be used to express, for example, that m_1 is the standard model that fits within an established theoretical framework (higher prior), while m_2 is more exotic and contradicts established theories (lower prior). There is no general rule for assigning prior probabilities, although in some cases it may be possible to justify a specific prior on quantitative grounds. In case it is not desirable to make specific prior assumptions about the models, one option is to pick a flat prior that assigns equal prior probability to both models. Another option is to directly compare

²³Note that the distribution for each model ignores the unobserved variables of the respective other model. That is because assuming one model as the true generator of the data means assuming that the other model is just fiction, so its unobserved variables should be independent from the data and the true model.

the likelihoods of the data given each model. The ratio of these likelihoods is called the *Bayes factor*:

$$K = \frac{p(x | m_1)}{p(x | m_2)}. \quad (2.15)$$

The Bayes factor can be used to express the relative evidence that the data provides for one model over the other, independent of prior beliefs in the model. It is thus (on an abstract level) comparable to a classic hypothesis test, where m_1 and m_2 encode the null and the alternative hypothesis, respectively. At the same time, the strong relationship between the Bayes factor and the posterior ratio via the prior makes the Bayes factor an indication of how the observed data would change different prior beliefs about models.²⁴ For a more detailed discussion of Bayes factors, see Kass and Raftery 1995; MacKay 2003; Dienes 2008; and Rougier 2019.

2.4.3 Inference Methods

Bayesian modelling provides an elegant framework for formulating models and inference questions. However, it is not always straightforward to obtain the results for a specific inference problem, such as calculating the posterior distribution. There are three main approaches to inference in Bayesian models, namely exact inference, sampling, and variational inference, and we will briefly discuss each of these approaches.

Exact inference aims at analytically deriving the shape of the posterior (or any other inferred distribution or value). This requires marginalizing out the probability of the observed variables $p_X(x)$ (called *evidence*), which is a normalizing constant that turns the product of likelihood and prior into a properly normalized probability distribution. Since this marginal distribution cannot always be derived analytically, exact inference is not possible for every model. A special case in which exact inference *is* always possible is when the prior and the likelihood are conjugate, i.e. the distributions for the prior and the likelihood are chosen such that the posterior is of the same family as the prior, but with different parameters. Examples of this relationship include Bernoulli (or binomial) likelihood with a beta prior, a categorical (or multinomial) likelihood with a Dirichlet prior, or a Gaussian likelihood with a Gaussian prior for the mean. For example, our second example above uses a Dirichlet distribution for the prior of the chord label probabilities λ , together with categorical distributions for each actual chord label $p(L_i | \lambda)$ (the likelihood of the label given the probabilities). This choice ensures that the

²⁴While a posterior ratio with a flat prior and the Bayes factor amount to the same value, the interpretation is different: the posterior ratio assumes the prior to be fixed; the Bayes factor leaves the prior *unspecified* and could be combined with different priors.

posterior of the chord-label probabilities $p(\lambda | L)$ is again a Dirichlet distribution.²⁵ The same goes for the note distribution of each chord with a Dirichlet prior $p(v_l)$, a multinomial likelihood $p(N_i | v_{L_i}, L_i)$ for each chord, and again a Dirichlet posterior $p(v_l | N)$.

Exact inference is not always possible, usually because some normalizing constant (e.g., the evidence term) requires computing an integral that does not have a closed form. As an alternative, the posterior can be approximated using *sampling*. The idea is to draw samples from the posterior distribution in a way that does not require to specify the posterior as a closed formula, and then use the sample to inspect the shape of the distribution (e.g., as a histogram) or to estimate quantities such as mean, variance, or entropy. The simplest form of sampling is called *rejection sampling*, which works for any distribution that allows drawing samples of the joint distribution (which is usually the case for generative models). When conditioning on the observed variables, all samples that do not have the correct assignment for the observed variables are “rejected.” The remaining samples are distributed according to the posterior distribution. While rejection sampling works without any knowledge about the joint distribution (except for being able to sample from it), it becomes very inefficient when samples with the correct values for observed variables are rare. This is particularly problematic when the observed variables describe a whole dataset, which is usually extremely unlikely to be exactly drawn from the joint distribution by chance, or when the observed variables are continuous.

A different set of sampling algorithms is the family of *Markov chain Monte Carlo* (MCMC) methods, which produce a set of samples from the posterior by creating an auxiliary Markov chain that has the same stationary distribution as the posterior.²⁶ Therefore, samples generated by the Markov chain are approximately distributed according to the posterior. The *Metropolis-Hastings algorithm* (Hastings 1970) works by drawing samples of the unobserved variables from an arbitrarily chosen proposal distribution q , conditioned on the previous sample: $q(z' | z^{(i-1)})$. The new sample is compared to the previous example using the true posterior distribution. By taking the ratio between the posterior probabilities of the two samples, the normalization constant is canceled and

²⁵The change of parameters from the Dirichlet-distributed prior to the Dirichlet-distributed posterior even gives rise to an interpretation of the parameters as “pseudo-counts” since they correspond to how often each chord label has been observed in the data. The parameters of the initial distribution then can be interpreted as how often we pretend to have seen each chord label before, where 1 simply encodes that the label is possible.

²⁶The auxiliary Markov chain has nothing to do with the model being a Markov model, this approach works for different model architectures. The Markov chain here is only a way of obtaining samples from the posterior.

we only need to evaluate the joint probabilities of the samples and the data:

$$r = \frac{p(z' | x)}{p(z^{(i-1)} | x)} = \frac{p(z', x) \cdot p(x)}{p(z^{(i-1)}, x) \cdot p(x)} = \frac{p(z', x)}{p(z^{(i-1)}, x)}. \quad (2.16)$$

In the case that $r \geq 1$, z' is accepted as the new sample $z^{(i)}$. If $r < 1$, then z' is accepted with probability r , otherwise $z^{(i)} = z^{(i-1)}$. Thus, the efficiency of Metropolis-Hastings sampling depends on a proposal distribution q that is not too different from the actual posterior.²⁷ An alternative sampling strategy is *Gibbs sampling* (S. Geman and D. Geman 1984), which is useful if many unobserved variables are involved. For each new sample, each variable Z_i is resampled iteratively, conditioned on the current values of all other variables X and Z_{-i} : $p_{Z_i|X,Z_{-i}}(z_i | x, z_{-i})$. Since all values except one are fixed, it is often possible to obtain this conditional distribution by renormalizing the joint distribution over the non-fixed random variable:

$$p(z_i | x, z_{-i}) = \frac{p(x, z_i, z_{-i})}{p(x, z_{-i})} = \frac{p(x, z_i, z_{-i})}{\int_{z'} p(x, z', z_{-i})}. \quad (2.17)$$

The resulting Markov chain again has a stationary distribution that corresponds to the posterior distribution $p(z | x)$.

When many unobserved variables are involved, sampling might still be an inefficient method since the space that should be mapped is simply too large and too many samples would be needed to represent it accurately. An alternative to sampling is *variational inference* (for an overview, see Blei et al. 2017), which is based on the idea of approximating the true posterior $p(z | x)$ with a simpler distribution $q(z)$, the *variational distribution*. The variational distribution is obtained by specifying a family of distributions $q_\theta(z)$ and choosing its parameters θ to minimize the difference between $p(z | x)$ and $q_\theta(z)$, measured by the Kullback-Leibler divergence $D_{\text{KL}}(q_\theta(z) \parallel p(z | x))$. By replacing the KL-divergence with an equivalent loss function, the *evidence lower bound* (ELBO), a series of optimization algorithms can be used to find the optimal parameters θ . A common choice for the variational family q_θ is a *mean-field family*, which assumes that all unobserved variables are independent, and uses an appropriate family for each individual variable:

$$q_\theta(z) = \prod_i q_{\theta_i}(z_i). \quad (2.18)$$

Mean-field families are useful when marginal posteriors of the individual variables are of interest, however they cannot capture any correlation between variables that are assumed to be independent. Recently, there has been interest in so-called *black-box variational inference* (Ranganath et al. 2014), i.e., the derivation of suitable variational

²⁷Equation 2.16 assumes that the proposal distribution q is symmetric, i.e., that $q(z | z') = q(z' | z)$. If that is not the case, r needs to be corrected by $\frac{q(z^{(i-1)} | z')}{q(z' | z^{(i-1)})}$.

families and inference procedures from any type of probabilistic model, in particular those expressed by probabilistic programs. This has led to the development of *automatic differentiation variational inference* (Kucukelbir et al. 2017), which makes use of automatic differentiation (as used, e.g., for training artificial neural networks) to derive a stochastic gradient for the ELBO that can then be used with generic optimization algorithms designed for stochastic gradient descent. As a result, variational inference can be used without the need for deriving model specific algorithms or using specialized software, making variational inference more flexible and accessible.

2.4.4 Probabilistic Programming

Deriving and implementing concrete inference algorithms for Bayesian models is tedious and time-consuming. *Probabilistic programming* is a method for expressing Bayesian models (i.e., probability distributions) as generative computer programs (e.g., N. D. Goodman, Tenenbaum, et al. 2016; van de Meent et al. 2018). A probabilistic program is a function or procedure that contains statements about sequentially drawing random variables from a local distribution. As such, the program defines a joint distribution over all sampled variables, factorized into a local conditional distribution. In fact any probability distribution can be represented as a probabilistic program, which can be seen by the fact that every distribution can be factorized into a product of conditional distributions, e.g. $p_{A,B,C}(a, b, c) = p(a) \cdot p(b | a) \cdot p(c | a, b)$.²⁸ The corresponding probabilistic program will then first sample A , then B (based on the value of A), and finally C (based on the values of A and B). Between the sampling statements, the program can perform arbitrary computations, which makes it possible to express very complex generative processes.

An example of a probabilistic program is shown in 2.1. Our two models are implemented as regular Python functions. The random variables are sampled using the `sample` function from the Pyro framework (Bingham et al. 2019). Both models return the observed variables while using the unobserved variables only internally, so running either function will return samples drawn from the joint distribution. Within each model function, `sample` statements can be freely mixed with other Python code, e.g., the `print` statements that are used to print the values of unobserved variables when a model is executed. Note, in particular, how the model functions reflect the structure of the notation used in Section 2.4.1. In particular, i.i.d. variables (or plates in the graphical notation) are

²⁸Note that expressing a given distribution as a sequence of conditionals is not always straightforward or even possible. However, if the model is generative, it already has a factorized form and is generally easy to implement as a probabilistic program, which also follows a generative idea.

```
import pyro
import pyro.distributions as dist
import torch

# definition of model 1
def model1(p_labels, p_notes, n_notes):
    # sample the chord label from the prior  $p(Z)$ 
    label = pyro.sample("Z", dist.Categorical(p_labels))
    print(f"Z = {label}")
    # sample the notes  $X$  from  $p(X|Z)$ 
    notes = pyro.sample("X", dist.Multinomial(n_notes, p_notes[label]))
    return notes # return  $X$ 

# definition of model 2
def model2(n_types, n_chords, n_notes):
    # sample the unobserved variables ("parameters"):
    # sample  $\lambda$  from its prior  $p(\lambda)$ 
    p_labels = pyro.sample("lambda", dist.Dirichlet(torch.ones(n_types)))
    print(f"lambda = {p_labels}")
    # sample  $v$  from its prior  $p(v)$ 
    p_notes = [pyro.sample(f"nu_{l}", dist.Dirichlet(torch.ones(12)))
                for l in range(n_types)]
    print(f"nu = {p_notes}")
    # sample the observed variables ("data"/chords)
    chords = [] # used to collect the sampled values
    for i in range(n_chords):
        # sample the chord label  $L_i$  from  $p(L_i|\lambda)$ 
        label = pyro.sample(f"L_{i}", dist.Categorical(p_labels))
        # sample the notes  $N_i$  from  $p(N_i|v, L_i)$ 
        notes = pyro.sample(f"N_{i}",
                            dist.Multinomial(n_notes, p_notes[label]))
        # collect the values
        chords.append({"label": label, "notes": notes.int()})
    return chords
```

Listing 2.1 – A probabilistic program written in Python using the Pyro framework. It encodes the two models discussed in the text as generative functions.

implemented using ordinary Python for-loops and list comprehension.²⁹

Frameworks for probabilistic programming come in a wide variety of types. A practical distinction is whether the language used to express probabilistic programs is dedicated or embedded. Frameworks with a dedicated language usually have full introspection into the probabilistic program, which helps with deriving efficient inference algorithms for a particular program since its operations (e.g., the distributions it uses) and program structure are known by the system. A disadvantage of dedicated systems is that users are restricted to the operations and programming constructs offered by the modelling language, which cannot be easily integrated with other code. Examples include *Stan*³⁰ (Carpenter et al. 2017), *Church* (N. D. Goodman, Mansinghka, et al. 2008), or *WebPPL*³¹ (N. D. Goodman and Stuhlmüller 2014).

Embedded probabilistic programming frameworks represent probabilistic programs as functions in an existing programming language, i.e., they provide additional operations (such as sampling or observing random variables) as library functions that can be mixed with ordinary program code. While these systems are very flexible, they usually provide less assistance with deriving inference mechanisms, which is why the application of black-box methods in this context is a very active area of research. Examples include *Anglican*³² (Clojure; Wood et al. 2014), *edward2*³³ (Python; Tran et al. 2018), *Gen*³⁴ (Julia; Cusumano-Towner et al. 2019), *monad-bayes*³⁵ (Haskell; Ścibior et al. 2015), *Pyro*³⁶ (Python; Bingham et al. 2019), or *Turing.jl*³⁷ (Julia; Ge et al. 2018). Embedded probabilistic programming frameworks are particularly attractive for corpus research since they provide a simple framework to start at a high level, good integration with existing libraries for data processing, and a high flexibility with respect to the models they can express.

²⁹While both functions work (using Pyro 1.5.0 and PyTorch 1.6.0) and produce samples from the correct distributions, they are not completely idiomatic in Pyro. In particular, the for loops would use the `plate` construct to indicate that the variables in different loop iterations are i.i.d.. Another shortcoming is that we use a fixed number of notes for all chords in the second model. A more sophisticated implementation would either take this information from the observed chords, or model the number of notes in each chord as another random variable (which would keep the model fully generative).

³⁰<https://mc-stan.org/>

³¹<http://webppl.org/>

³²<https://probprog.github.io/anglican/index.html>

³³<https://github.com/google/edward2>

³⁴<https://github.com/probcomp/Gen.jl>

³⁵<https://github.com/tweag/monad-bayes>

³⁶<https://pyro.ai/>

³⁷<https://github.com/TuringLang/Turing.jl>

2.4.5 Relevance to Music Research

Bayesian modelling, as has been just briefly outlined, is a natural fit for corpus studies on music: Corpora naturally constitute a set of “observations” from which a corpus researcher wants to draw conclusions about properties that cannot be directly observed. These conclusions are also naturally *uncertain* since the unobserved properties are generally not unambiguously deducible from the surface observations.³⁸ Inferences about unobserved properties are *always* based on assumptions about the relation between observed and unobserved properties. Bayesian models make this relation explicit by defining a joint distribution over observed and unobserved variables that encodes the modellers assumptions. In addition, Bayesian inference provides a natural way to quantify the uncertainty of conclusions, since it generally does not yield a point estimate but rather a distribution over unobserved variables.

An example might help to illustrate this point. In Section 2.2 we discussed the bigram model of harmonic progressions. The central idea of this model is that chord progressions consist of sequences of local transitions from one chord to the next. For a given chord, c_i there are several possibilities for the next chord c_{i+1} . The plausibility of the second chord given the first one can be expressed as a probability $p(c_{i+1} | c_i)$. Since all bigrams are treated the same, independently of their position in the piece, this transition probability can be expressed using a fixed transition matrix T such that $p_{c_{i+1}|c_i}(l | k) = t_{kl}$. The probability of a sequence $c = c_1, \dots, c_N$ is then

$$p(c_1, \dots, c_N) = p(c_1) \cdot \prod_{i=1}^{N-1} p(c_{i+1} | c_i), \quad (2.19)$$

which corresponds to a first-order Markov model. Given a corpus of chord sequences, we might wish to infer the transition probabilities T , which can be understood as a property of the style or musical language underlying the sequences. Note that the “true” transition probabilities are unknown, but the probabilistic model defines how they are related to the observed sequences. A common way to estimate T is to take the relative frequency of each transition as observed in the corpus. Let n_{kl} be the number of times the chord transition (k, l) is observed. Then the estimate for the corresponding transition probability is

$$\widehat{t}_{kl} = \frac{n_{kl}}{\sum_{l'} n_{kl'}}. \quad (2.20)$$

In the Bayesian framework, the parameters of the model (i.e., T) are considered to be

³⁸An exception to this rule are *descriptive statistics*, which (as the name indicates) are used to describe or summarize a sample (i.e., a corpus). However, as soon as a corpus study moves beyond pure description of the data and, e.g., seeks to provide analyses or explanations of observed phenomena, *inferential statistics* are necessary (of which Bayesian statistics are a special case).

random variables, so the probability of a sequence can be seen as the likelihood of the sequence given the parameters $p(c | T)$. The estimate \hat{T} happens to be the estimate that maximizes this likelihood for all sequences:

$$\hat{T} = \operatorname{argmax}_T p(c | T), \quad (2.21)$$

which is called the *maximum likelihood estimate* (MLE). The MLE differs from the posterior in two points: First, it does not consider the prior of T .³⁹ Second, the MLE is a *point estimate* since it provides a single value for T instead of a distribution over possible values. While \hat{T} is the “best” estimate for the transition probabilities (according to the likelihood of the observed chord progressions), there is not way to tell *how* good it is compared to other possible values, or how certain we can be that it is (close to) the true value of T .

In contrast, the posterior distribution does express this uncertainty. In the case of the transition probabilities T , the conjugate prior (and hence also the corresponding posterior) is a Dirichlet distribution for each antecedent chord (Gómez-Corral et al. 2015). The Dirichlet posterior for some antecedent chord k can be easily interpreted in terms of its parameters (one per consequent chord):

$$t_k | C \sim \operatorname{Dirichlet}(n_{k1} + \alpha_{k1}, \dots, n_{kL} + \alpha_{kL}), \quad (2.22)$$

where α denotes the parameters of the prior. The parameters of the posterior are obtained by adding to the parameters of the prior the absolute occurrences of the corresponding bigrams. How does this encode uncertainty? Intuitively, the parameters of the Dirichlet distribution correspond to the number of times we have observed each outcome. The more observations we have made in total, the more certain we can be about our estimates. For example, after flipping a coin 1000 times we can be much more certain about the coin being fair or biased than after flipping it 10 times. Accordingly, reporting the absolute occurrences of bigrams (e.g., de Clercq and Temperley 2011) provides more information than only reporting relative frequencies.⁴⁰ The difference becomes relevant when analyzing the transition patterns of rare antecedent chords. The total number of observations for that antecedent chord gives an indication of how reliable an analysis of its transition distribution is. It is worth noting that the Bayesian perspective does not necessarily contradict intuitive measures and analysis methods. In this case, it rather provides formal justification and interpretability for them.

³⁹The prior can be included in the term that is maximized. In that case, the estimate is called maximum a posteriori estimate.

⁴⁰To be precise, the absolute bigram counts correspond to the parameters of a Dirichlet posterior for an improper prior that expresses no previous observations of any bigram (see Gómez-Corral et al. 2015).

There is, however, an even deeper connection between music theory, corpus research, and Bayesian inference. Listening, analysis, learning, and theory building can all be understood as inference under uncertainty from observations to unobserved causes, parameters, or entities. Listening and analysis aim at uncovering latent structure in a particular piece, while learning (in the sense of becoming familiar with a musical language) and theorizing is inference to the regularities underlying a musical language, implicitly in the case of learning, and explicitly in the case of theory building. Bayesian modelling provides both a general methodology that can be applied in each of these scenarios, and a useful perspective on the relation between observations, latent entities, and inference. The generality of the Bayesian perspective has led to the idea that Bayesian inference is one of the fundamental principles of cognition.⁴¹ If music is understood as a cognitive phenomenon that involves agents such as composers, performers, listeners, and analysts, then the cognitive capacities of these agents (and by extension the Bayesian perspective on these capacities) is an important perspective for musical corpus research.⁴²

2.5 A Recipe for Model-based Corpus Research

In the previous sections we have advocated for an approach to corpus research that is based on explicit models and Bayesian inference. Although tying into a specific modelling framework may seem like a limitation, (Bayesian) modelling is in fact both very general and compatible with other methodologies. For example, probabilistic models might not seem to be a good choice for expressing relationships that are deemed to be deterministic. However, since deterministic relationships can be understood as a special case of probabilistic relationships, both can usually be mixed in a straightforward way, as can be seen, for example, in probabilistic programs. Similarly, models need not be generative (in the sense of factorizing into prior and likelihood that “generates” the observations) in order to do Bayesian inference. There are other ways to express joint distributions that cannot be interpreted as causal processes.⁴³ Finally, probabilistic models are not restricted to use a small set of known distributions, such as normal, categorical, multinomial, or Dirichlet distributions. Any well-defined distribution can be a valid model. However, known distributions are often both well understood in

⁴¹E.g., Tenenbaum et al. 2011; Vilares and Kording 2011.

⁴²For understanding music as a cognitive phenomenon, see for instance Huron 2006; Jackendoff and Lerdahl 2006; and Pearce and Rohrmeier 2012.

⁴³Examples include Markov random fields, which express the joint distribution as a normalized product of factors that encode “undirected” relationships and are interpretable as mutual influence rather than unidirectional dependence. Note that such models are still called “generative” in machine learning terminology, because they still describe a joint distribution.

terms of their interpretation (which helps building interpretable models), and have good mathematical properties (which helps with practical inference). On top of that, complex models can be built out of simple components, as we have seen in the examples above.

In terms of compatibility, Bayesian inference and other means of analysis (e.g., frequentist statistics) are not mutually exclusive and can both be model-based. For example, the bigram model of chord progressions can be analyzed in terms of both maximum likelihood estimates and posterior distributions. In fact, any kind of systematic analysis can be seen as an implicit model. The advantage of the Bayesian approach is that it makes the connection between analysis and model explicit. In many cases, the Bayesian perspective can even provide a reinterpretation of an existing method of analyses (such as maximum likelihood estimates), and reveal similarities and differences to genuinely Bayesian concepts such as posterior distributions.

To summarize this chapter, we suggest a “recipe” for approaching corpus research from a model-based perspective. These steps should not be understood as a set of conditions that model-based research must satisfy, but rather as a high-level guide that might provide some orientation when conducting a model-based project. For a given research question, the following steps are usually involved:

1. **Identifying the relevant entities and relations.**

This step involves finding the right level of abstraction for the purpose at hand. Observed entities may be restricted by what is encoded in a pre-existing corpus, but on the other hand might also inform what needs to be encoded in a yet-to-be-created corpus (e.g., in the form of annotations). The relations between the entities determine the fundamental structure of the model. It is important to be as explicit as possible about the interpretation of the assumed relations since the interpretation guides and justifies the way the relations are formalized.

2. **Choosing representations of entities.**

The formal representation of entities determines which information is encoded about each type of entity, and which relations can be expressed between the entities. For example, representing chords as absolute chord labels, scale degrees, or note sets has different implications on which relations between the chords or between chords and other entities (keys, sections, notes) can be expressed. As with the identification of entities in Step 1, the representation is constrained by (and in turn constrains) the corpus, e.g., in terms of the specific annotation format: observed entities in the model cannot contain information that is not present in the corpus (but one may always choose to ignore some of the available information).

3. **Formalizing relations.**

Relations encode the structural assumptions about the observed information in the corpus and any latent entities. The model should specify exactly under which conditions a certain relationship exists (or can exist) between two or more entities. For example, the bigram model posits a transition relation between every pair of neighboring chords in a harmonic sequence, while a grammar model states that a certain syntactic or semantic relation between two chords exists if and only if the corresponding rule is used to generate the two chords in a valid derivation of the sequence. Relations are constrained by the representation of the model entities. Bigrams, for example, presuppose that elements are organized in a sequence. In generative models, relations can be represented as steps in a generative process (e.g., the application of grammatical production rules). In a fully Bayesian model, relations are typically probabilistic and include the relationship between the observed entities of the corpus and unobserved objects and parameters that are the subject of the research question (such as functional categories, dependency relations, transition probabilities, chord profiles, etc.).

4. **Formulating questions and defining the analysis.**

The purpose of the analysis is to answer a research question based on the model. In the Bayesian case, this is usually achieved by restating the research question in probabilistic terms – i.e., using distributions over latent variables (posterior), new data (predictive), or even the model itself (hypothesis testing) – and applying a generic inference method to obtain a result. The quantities obtained by the analysis are linked to other entities within the model via the explicitly assumed relations. However, as we have seen above, it is also possible to perform analyses that are external to the explicit model. In that case it is important to consider and defend the implicit assumptions and relations that are expressed by the method of analysis. If the main research interest is in the model itself, the “analysis” consists in an evaluation of the models’ ability to capture the characteristics of the dataset.

Corpus studies typically involve more steps than the ones mentioned above, such as encoding, annotation, and implementation of the analysis. Moreover, the above steps will rarely be taken in the presented order, instead there is usually some feedback between the different steps. However, all of the points mentioned need to be addressed, in some form or another, in a corpus-based research project. Since Bayesian modelling enforces fully explicit models and provides a principled way of handling uncertainty, we believe that it presents a reasonable default for model-based corpus research. However, there may be good reasons to deviate from this default. Ultimately, it is most important that corpus research formulates its models as clearly as possible, regardless of the framework that is used to do so.

2.6 Acknowledgments

The work presented in this chapter resulted from the project “From Bach to the Beatles: Exploring Compositional Building Blocks and Musical Style Change with Hermeneutic and Computational Methods” funded by the Volkswagen Foundation (2018–2020).

Preliminary Studies

Part II

Interlude 1

The first two chapters have established the central questions and methodology of this thesis. The goal is to develop a model of tonal structure, identifying fundamental musical relations between the notes in a piece of music as well as between these notes and latent entities. The model takes the form of a generative process that produces the observed notes based on these structural relations. Reversing this process allows a listener to infer the latent structure that explains the observed notes and thus obtain an *interpretation* of a piece.

Before arriving at the final model, the second part of this thesis first explores the fundamental relations and latent entities that the model should capture. The following three chapters each present a case study on a specific aspect of structure. Chapter 3 investigates the relationship between latent entities and their surface forms, presenting a heuristic approach to finding voice-leading schemata. Voice-leading schemata are polyphonic patterns that are used as templates of scaffolds for segments of a composition and thus can be considered a type of latent structural entity, similar to chords. The presented approach is based on polyphonic skipgrams which are able to select groups of notes that match the schema prototype from an unstructured stream, ascribing vertical and horizontal connections to them in an ad-hoc fashion. A classifier then tries to distinguish accidental occurrences of the schema's interval pattern from true instances of the schema based on features of the selected notes, such as duration, metric weight, or regularity.

The core limitation of the schema matcher is that it does not consider schemata as *explanatory for* the surface but rather tries to identify schema occurrences *heuristically from* the surface. Chapter 4 explores the explanatory role of latent entities (harmonies in this case) by linking them to the surface through a generative process. It presents a corpus study based on a Bayesian model that characterizes chord types as distributions of both chord tones and typical non-chord tones occurring in the surface realizations of these chord types. The results show that different chord types have distinctive profiles of non-chord tones, and that these profiles can differ between different genres, which indicates that latent entities can influence the functional roles of surface notes.

Interlude 1

While the harmony model addresses the relation between latent harmonies and surface notes, it is not able to capture the direct relations between notes. Chapter 5 investigates this aspect of structure, proposing a grammar-based model of melodic elaboration in North Indian classical music and other musical styles that are centered around modes. The model shows how melodic complexity can arise from a small number of musically meaningful ornamentation operations within the context of monophonic melodies. Latent vertical structure in these melodies is explained by reference to the underlying mode and the relative stability of notes within this mode.

3 Finding Voice-Leading Schemata¹

Abstract

Musical schemata constitute important structural building blocks used across historical styles and periods. They consist of two or more melodic lines that are combined to form specific successions of intervals. This paper tackles the problem of recognizing voice-leading schemata in polyphonic music. Since schema types and subtypes can be realized in a wide variety of ways on the musical surface, finding schemata in an automated fashion is a challenging task. To perform schema inference we employ a skipgram model that computes schema candidates, which are then classified using a binary classifier on musical features related to pitch and rhythm. This model is evaluated on a novel dataset of schema annotations in Mozart’s piano sonatas produced by expert annotators, which is published alongside this paper. The features are chosen to encode music-theoretically predicted properties of schema instances. We assess the relevance of each feature for the classification task, thus contributing to the theoretical understanding of complex musical objects.

3.1 Introduction

Voice-leading schemata are frequently used patterns that can be found across historical periods, ranging from Renaissance, Baroque, and Classical to modern tonal music; examples include such well-known schemata as the Lamento, the Pachelbel, the descending-fifths sequence, and cadences (Forte 1979; R. Gjerdingen 2007; Caplin 2014; Jan 2013).

¹Originally published as:

C. Finkensiep, K. Déguernel, M. Neuwirth, and M. Rohrmeier (2020). “Voice-Leading Schema Recognition Using Rhythm and Pitch Features”. In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. ISMIR. Montreal, Canada (online), pp. 520–526. DOI: 10.5281/zenodo.4245482

The skipgram-based schema matcher was developed by CF, the classifier and its evaluation by CF and KD. The schema lexicon was produced by KD and MN, the annotations by MN and an external annotator. CF, KD, and MN contributed equally to Sections 1-4. Section 5 was written by KD and CF. Sections 6 and 7 were mainly written by CF with support from the other authors.



(a) A (true) Fonte at the beginning of K281-iii.

(b) A possible candidate for a Fonte in K283-iii.

Figure 3.1 – An example of a Fonte (a) with structural notes highlighted. The task is to decide whether proposed instances such as (b) are true instances or not.

A schema serves as a template for contrapuntal structure that can be elaborated in multiple ways.

At present, there is only scant quantitative evidence about the frequency and diachronic distribution of schemata across history (e.g., R. Gjerdingen 2007; Byros 2009); large-scale, machine-readable datasets on schemata are not yet available. For assessing the prevalence of schemata in a corpus of music, automated recognition of schema instances can be a time- and cost-efficient alternative to manually labelled data. However, there are two key challenges for computational approaches when seeking to uncover note patterns in music: (1) the multidimensional (polyphonic) structure of music as opposed to, for example, the sequential structure of written text (Meredith et al. 2002); (2) the highly flexible nature of these patterns, given that the structural notes in the individual voices can be elaborated in a wide variety of ways.

Voice-leading schemata can be defined as configurations of two or more voices that move together through a sequence of stages, forming particular patterns of successive vertical intervals that occur within a specific tonal context. Consider the example of the Fonte (e.g., R. Gjerdingen 2007): The Fonte is a four-stage pattern involving at least two voices. The bass moves through the scale degrees $\sharp\hat{1} \rightarrow \hat{2} \rightarrow \hat{7} \rightarrow \hat{1}$ of a major scale, while the soprano follows the pattern $\hat{5} \rightarrow \hat{4} \rightarrow \hat{4} \rightarrow \hat{3}$, thus producing the following sequence of vertical intervals: tritone \rightarrow minor third \rightarrow tritone \rightarrow major third. The schema prototype can be elaborated in actual compositions in many different ways. For instance, the notes belonging to one stage can be displaced in time; any number of elaboration notes can be inserted between the structural notes of one stage and between stages... An example illustrating the surface realization of a Fonte is shown in Figure 3.1a. While containing the correct interval pattern is a central property of any schema instance, it is not sufficient: the selected notes must also provide the contrapuntal template for its context, so that the notes contained in the time-span covered by the schema candidate can be meaningfully interpreted as ornamentations of the selected notes. Figure 3.1b shows a candidate for a Fonte instance. The task at hand is to decide whether or not

such a candidate is a true schema instance.

To tackle the problem of schema detection, this paper provides two contributions. First, we propose a novel dataset with hand-annotated schemata found in Mozart’s piano sonatas (Section 3.3). Second, we present a binary classifier that recognizes true schema instances among a set of proposed schema candidates based on rhythm and pitch features related to regularity, complexity, salience, and harmonic context (Section 3.4). We evaluate the impact of these features on the classification task using a logistic regression (Section 3.5).

3.2 Related Work

Automated discovery and recognition of musical patterns is a topic of ongoing interest in the MIR community (Lartillot 2016; Conklin and Bergeron 2010; Meredith et al. 2002; Cambouropoulos et al. 2001; Giraud, Déguernel, et al. 2014; Janssen et al. 2014; Lartillot 2008). Voice-leading schemata as a specific class of patterns have so far received only little attention; they have been approached with computational methods only very recently. For instance, Symons (2017) has developed an algorithm that recognizes schemata in a small corpus, pointing out the importance of rhythmic regularity. Finkensiep, Neuwirth, et al. (2018) tackle the problem of temporal displacement and free polyphonic textures using a two-dimensional extension of skipgrams, which have previously been proposed by Sears et al. (Sears, Arzt, et al. 2017; Sears and Widmer 2020). Recently, Katsiavalos et al. (2019) have presented a system that uses heuristics-based time-span reduction to discover and recognize schemata.

Several studies aimed at finding cadences, which can be viewed as a subcategory of voice-leading schemata, and evaluated the features relevant for the classification task. Bigo et al. (2018) evaluated a set of 44 features linked with the moment of cadential arrival, which are integrated using a support-vector machine for classifying beats as belonging to a cadence or not. Sears, Arzt, et al. (2017) use skipgrams on vertical slices to find cadences using a figured bass-like representation of the notes in each slice. Duane (2019) approaches cadences directly as voice-leading patterns by trying to recognize and learn them from melodic motion.

3.3 Dataset

Our dataset is based on the full set of Mozart’s piano sonatas encoded in MusicXML format. These 18 sonatas with 3 movements each (thus 54 movements in total) have been composed between 1774 and 1789 and constitute a prominent sample of the classical style. The pieces in the dataset contain 103,829 notes in total distributed over 7,500 measures, with 244 hand-annotated true instances (0.13%) and 190,994 automatically generated false instances (99.87%) for the selected schema types and subtypes (see Table 3.1).

3.3.1 Schema Formalization and Lexicon

For the present study, we selected 10 schema types and 20 subtypes (listed in Table 3.1) which have been suggested in the literature (R. Gjerdingen 2007; Caplin 2014; Rice 2015; Rice 2014). The approach presented here assumes that a schema type consists of (1) a fixed number of voices; (2) a fixed number of stages, whereby each stage contains one note per voice; and (3) a characteristic interval pattern between these notes. The prototype for each schema variant (or subtype) is specified using a formal notation. For instance, the prototype of the two-voice Fonte is encoded as:

```
"fonte.2": [[ "a1", "P5" ],  
            [ "M2", "P4" ],  
            [ "M7", "P4" ],  
            [ "P1", "M3" ]]
```

where ".2" indicates the two-voice variant of the Fonte. Each note is given as an interval to some arbitrary reference point. Since all possible transpositions of the schema are considered by the matcher, it is not necessary to know the reference key.

Schema instances are encoded as nested arrays of notes in the same form as the corresponding prototypes. Instances may deviate from the shape of the prototype if (a) a note that would repeat its predecessor (e.g., the second $\hat{4}$ in the Fonte) is held over or missing, or (b) two adjacent voices merge and are represented by a single note on the surface.

Schema	Variant	Occurrences
Do-Re-Mi	.2	5
	.2(.min)	10 (3)
Fenaroli	.2.flipped(.min)	43 (8)
	.2.melcanon(.min)	6 (2)
	.2.basscanon.min	1
Fonte	.2	49
	.2.flipped	2
	.2.majmaj	8
Indugio	.2	9
	.2.voiceex	5
Lamento	.2	2
Lully	.2	2
Morte	.2	1
Prinner	.2	32
Quiescenza	.2	46
	.2.diatonic	6
Sol-Fa-Mi	.2	4

Table 3.1 – List of schemata with their variants and number of occurrences in the Mozart Piano Sonata dataset.

3.3.2 Data Production

The dataset consists of two parts: manual annotations by experts and automatically retrieved candidates for schema instances, i.e., sets of notes with an interval pattern conforming to a schema variant. Both the annotations and the computed candidates share the same encoding format, namely a nested lists of note IDs (one sublist per stage, one note per voice) that corresponds to note elements in a MusicXML representation of the scores. While the manual annotations provide the true instances of the dataset, the false instances consist of all skipgram candidates that do not appear in the annotations.² The complete dataset is available on github.³

²This includes alternative versions of true instances with several possible note selections.

³https://github.com/DCMLab/schema_annotation_data

Expert Annotations

Two annotators (the third author and Adrian Nagel) provided their analyses by using a web-based annotation app that was specifically developed for the annotation process. The app displays a score using the Verovio toolkit (Pugin et al. 2014), and allows the user to select individual notes from the musical score to mark schema instances. Instances are automatically checked for conformance with the schema prototype in the lexicon, while permitting the deviations described in Section 3.3.1. The annotators also considered additional criteria such as harmonic signature, phrase structure, pattern repetition, and form-functional context.⁴

Computing Candidates with Skipgrams

In order to compute all candidates of schemata for the classifier, we base our work on the generalized skipgram model proposed in (Finkensiep, Neuwirth, et al. 2018), which enumerates two-dimensional note configurations that occur within certain temporal bounds. We use this algorithm to find configurations with a maximal note displacement of 1 bar per stage and a maximal distance of 1 bar between the onsets of two adjacent stages. The configurations are filtered for a specific interval pattern during enumeration regardless of the local keys. This method provides us with all candidates for a schema instance within a reasonable window. However, due to the exhaustive search and a high number of possible note combinations, our resulting dataset is extremely unbalanced. Because of the high combinatoric complexity, we restrict this study to two-voiced schema variants. Furthermore, we reduced the number of candidates to at most 25 per group of temporally overlapping candidates using a previous version of the model presented here.

3.4 Features and schema classification

3.4.1 Musical Features

By using precomputed schema candidates that are known to have the correct interval structure (which is all information that we consider for a specific schema type), the problem is narrowed down to deciding whether or not the candidate consists of the

⁴As detailed in the schema-annotation guidelines (https://github.com/DCMLab/schema_annotation_data/blob/master/manual/manual.pdf).

structurally important notes. To this end, we have defined a set of features with regard to rhythmic, pitch, and metric information, inspired in part by previous work (R. Gjerdingen 2007) and that we wish to evaluate with the classifier. These features attempt to measure the *recognizability* of the candidate as a structural pattern, assessing, for example, its complexity, salience, or regularity in various musical dimensions. For a schema candidate C that consists of a number of stages n_s and a number of voices n_v , let $C_{s,v}$ denote the note from stage s and voice v . Each note is represented by an onset, an offset, and a pitch. Whenever pairs of notes are compared, K denotes the numbers of compared note pairs (excluding pairs with missing notes).

The first feature can be considered a rough estimate of the *harmonic or modal signature* of the schema candidate. We define the *harmonic profile* of a candidate as the distribution of pitch-classes (relative to the transposition of the match) of notes that overlap with the time span of the candidate (excluding the matched notes themselves). The profile d_{ist} is defined as the Euclidean distance between a match's pitch profile and the average pitch profile of all true instances of the same schema. Thus, this feature uses training data to derive the prototype profiles instead of defining a harmonic signature a priori.

Three features address the *regularity* of pitch and rhythm between pairs of stages. r_{reg} measures the average rhythmic dissimilarity between each pair of stages. For a pair of stages, the rhythmic dissimilarity is defined as the sum of the temporal distance of the notes of the same voice, given the best alignment possible when projecting one stage unto the other. m_{reg} is defined very similarly, but here the alignment offset is fixed to whole beats to preserve metric position. Finally, p_{reg} measures the average pitch dissimilarity between each pair of stages. Similar to rhythmic dissimilarity of a pair of stages, pitch dissimilarity is defined as the sum of the pitch distances of the notes of the same voice, given the best pitch alignment possible when projecting one stage unto the other. These features are defined as

$$*reg = \frac{1}{K} \sum_{\substack{(s,s') \in \text{stages} \\ s \neq s'}} \min_{\delta} \left(\sum_{v=1}^{n_v} |\mu(C_{s,v}) - \mu(C_{s',v}) - \delta| \right), \quad (3.1)$$

where μ corresponds to onset for r_{reg} and m_{reg} , and to pitch for p_{reg} . For m_{reg} , δ is restricted to integer multiples of a beat.

We then define features corresponding to the *complexity* of the candidates in terms of displacement between pairs of notes. rd_{sums} and pd_{sums} respectively correspond to the average temporal and pitch distance between each note of the same stage. They are defined as

$$*d_{sums} = \frac{1}{K} \sum_{s=1}^{n_s} \sum_{\substack{(v,v') \in \text{voices} \\ v \neq v'}} |\mu(C_{s,v}) - \mu(C_{s,v'})|, \quad (3.2)$$

where μ corresponds to onset for rdsyms and to pitch for pdsyms. Similarly, rdsumv and pdsumv respectively correspond to the average rhythmic and pitch distance between each note of the same voice. They are defined as

$$*dsumv = \frac{1}{K} \sum_{v=1}^{n_v} \sum_{\substack{(s,s') \in \text{stages} \\ s \neq s'}} |\mu(C_{s,v}) - \mu(C_{s',v})|, \quad (3.3)$$

where μ correspond respectively to onset and pitch for rdsumv and pdsumv.

Another perspective at pitch displacement is provided by vdist, which measures the average amount of octave jumps within a voice from one stage to the next, and is defined as

$$vdist = \frac{1}{K} \sum_{v=1}^{n_v} \sum_{s=1}^{n_s-1} \left\lfloor \frac{\text{pitch}(C_{s+1,v}) - \text{pitch}(C_{s,v})}{\text{octave}} \right\rfloor. \quad (3.4)$$

While a certain complexity may be necessary to make a regular pattern recognizable in the first place, a more complex pattern can be more difficult to detect in the presence of other notes. For this reason, onsets counts the average number of distinct note onsets in the context of each stage. A low number of onsets allows the stages to be rhythmically displaced while still being recognizable as a unit. Given the number of distinct note onsets D_s for each state s , we have

$$\text{onsets} = \frac{1}{n_s} \sum_{s=1}^{n_s} D_s. \quad (3.5)$$

Finally, we define two features representing the *salience* of the candidate. First, we consider dur, which corresponds to the sum of all note durations in the candidate,

$$\text{dur} = \sum_{s,v} \text{offset}(C_{s,v}) - \text{onset}(C_{s,v}). \quad (3.6)$$

Then we consider mweight, a feature based on metric weight. We define our metric weight function as follows:

$$\text{mw}(C_{s,v}) = \begin{cases} 2 & \text{if onset}(C_{s,v}) \text{ is on a strong beat.} \\ 1 & \text{if onset}(C_{s,v}) \text{ is on a weak beat.} \\ \frac{1}{2^p} & \text{if onset}(C_{s,v}) \text{ is on a subbeat,} \end{cases}$$

where p is the number of prime factors needed to express the subbeat. Given that function, mweight corresponds to the average metric weight of each note of the candidate:

$$\text{mweight} = \frac{1}{K} \sum_{s=1}^{n_s} \sum_{v=1}^{n_v} \text{mw}(C_{s,v}). \quad (3.7)$$

3.4.2 Classification and Evaluation Method

The features described above are used as an input to a logistic regression model, a simple binary classifier model that uses a linear combination of the input features and applies a sigmoid to that score, yielding a value between 0 and 1 that indicates the probability of the input to be a true instance. Since a logistic regression is a special case of a neural network without hidden layers, this approach can be naturally extended to include more layers, allowing for more complex, non-linear feature combinations. However, preliminary experiments have shown that non-linear models (such as simple neural networks and support-vector machines) do not increase model performance and instead lead to overfitting, so we exclude them here.

The input data consists of expert annotations and skipgram candidates, produced as described in Section 3.3.2. To get consistent temporal information about the notes, we unfold all repetitions and jumps notated in the scores. Repeated occurrences of notes are disambiguated by selecting for every schema candidate those note occurrences that have a consistent temporal order and cover a minimal time span. Finally, matches that have incomplete stages (due to implicit notes, as described above) are converted into complete instances with missing notes marked explicitly.

To evaluate the model's performance, we use 5-fold cross validation.⁵ The pitch histograms used for `profiledist` are computed on the respective training set of each run. In order to get an unbiased model, we follow the advice given in (King and Zeng 2001) and balance our dataset by upsampling the true instances to match the number of false instances. The model is trained on the balanced training data using the Julia package `GLM.jl`⁶ and applied to both balanced and unbalanced test data. In addition, a prior-corrected version of the model (see King and Zeng 2001) is applied to the unbalanced data.

The code for the whole evaluation pipeline is provided online⁷, including a notebook⁸ that was used to generate all results and figures in this paper.

⁵A 5-fold split was chosen to balance the number of folds and size of the resulting test set.

⁶<https://github.com/JuliaStats/GLM.jl>

⁷https://github.com/DCMLab/schemata_code/tree/ismir2020

⁸https://github.com/DCMLab/schemata_code/blob/ismir2020/notebooks/ismir2020_classification.ipynb

3.5 Results and Discussion

3.5.1 Classification Performance

The overall performance of the model is shown in Table 3.2, aggregating over the predictions on all test sets. When applied to data balanced by upsampling, the model achieves a high classification performance with an F-score of 0.894 and a Matthews correlation coefficient (MCC) of 0.787. Since the model is trained on balanced data, applying it to unbalanced data simply scales the number of true positives and false negatives, resulting in a drastically reduced precision. Using the prior correction of the unbalanced dataset results in a very high accuracy; however, it introduces a bias to label matches as non-instances, which results in the false negatives dominating the false positives.

Figure 3.2 shows how the predicted probability of being a true instance is distributed for instances and non-instances (upper-left corner). Non-instances overwhelmingly receive low probabilities and instances are typically rated very high. This is in line with the model's good performance on raw data, but it also reveals why imbalance poses a serious problem: while the majority of non-instances are correctly discarded by the model, a minority remains indistinguishable from true instances under the model. When the skipgrams propose many more non-instances than instances, the small part of indistinguishable non-instances becomes huge in relation to the true instances. Note that simply reducing the number of matches does not necessarily improve the situation: taking away the matches with a rating < 0.5 still leaves us with the problematic cases.

A lot of non-instances are proposed as combinatoric variations around true instances. To test whether the problematic cases are variations of true instances or genuine non-instances, we group all matches according to temporal overlap (prior correction is based on the imbalance of the groups here). The results (Table 3.2) show that grouping drastically increases the performance compared to the ungrouped condition but does not get close to the performance on balanced data, indicating that there is still a significant number of indistinguishable true non-instances.

This effect of indistinguishability may be seen as an indicator that our list of features lacks those features that would help resolve the remaining cases and clearly separate the classes. However, it is not clear that finding such features is easily attainable. First, consider that while the existing features are already very informative, the information needed to distinguish the problematic cases would have to be much more precise. Even when the probability of getting a positive result for a non-instance is only 10^{-3} , a true instance proportion of 10^{-3} still leaves a 50% chance that a positive result is a false positive. Second, our annotators conformed to very strict standards in order to discard

Condition	TP	TN	FP	FN	accuracy	precision	recall	F-score	MCC
Balanced	171,400	169,867	21,107	19,574	0.893	0.890	0.898	0.894	0.787
Unbalanced	219	169,867	21,107	25	0.889	0.010	0.898	0.020	0.089
Unbalanced (corrected)	15	190,923	51	229	0.999	0.227	0.061	0.097	0.118
Grouped	220	6,009	2,663	12	0.700	0.076	0.948	0.141	0.218
Grouped (corrected)	43	8,596	76	189	0.970	0.361	0.185	0.245	0.245

Table 3.2 – Performance of the model in different conditions. TP = true positives, TN = true negatives, FP = false positives, FN = false negatives, MCC = Matthews correlation coefficient.

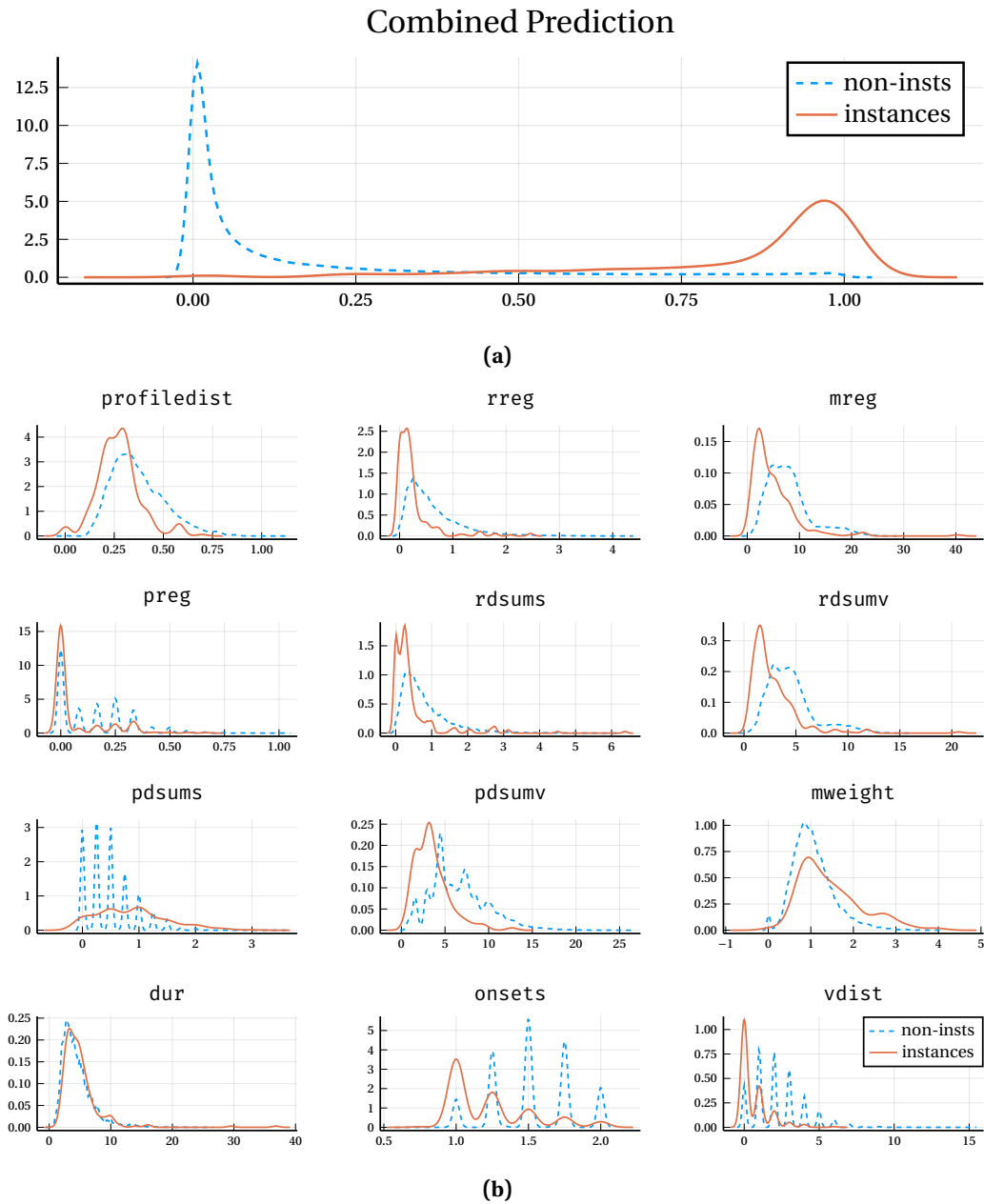


Figure 3.2 – Distribution of model prediction (a) and feature values (b) over instances and non-instances as a kernel density estimate. The more the curves tend in opposite directions, the better the two classes are separated.

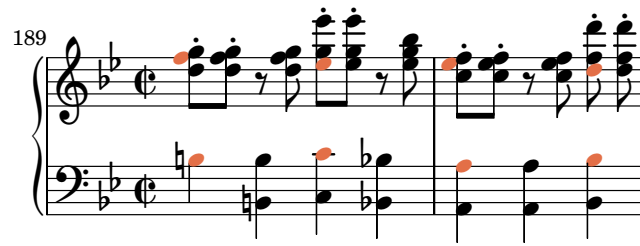


Figure 3.3 – An ambiguous Fonte match (K333-iii). While intrinsically this is a highly plausible instance (interval pattern, tonal context, melodic parallelism), the context discards it, as the pattern is in fact part of a larger descending-fifths sequence.

non-instances, restricting true instances to cases where the schema is a highly plausible template for the musical surface. Such judgments rely on implicit music-theoretical knowledge and intuition, which are difficult to model.

Finally, a look at some highly confident false positives suggests that if schema classification is defined as a binary task (a surface pattern is a schema instance or not), then the performance of this task can hardly be improved.

For example, the excerpt in Figure 3.1b may not look like a very plausible Fonte at first sight (and was not classified as such by the annotators). However, the last two bars clearly contain the correct contrapuntal pattern for the stages 3 and 4. The beginning can be interpreted as a melodic unfolding of an Em chord that is ornamented by the notes of a B⁷ chord, most clearly in the neighbor note d \sharp to e (i.e., the bass for the stages 1 and 2 of a Fonte). Therefore, it can be argued that this section shares its contrapuntal structure with the Fonte, even though the typical parallelism is missing. Another, converse, example can be seen in Figure 3.3: in isolation, the pattern is a clear instance of a Fonte, but it is continued in the manner of a larger descending-fifths sequence, which, depending on the definition used, may discard it as a Fonte. A negative definition like this is very difficult to check under the current paradigm.

3.5.2 Feature Evaluation

Figure 3.4 shows the influence of each feature in a model trained on the full balanced dataset. Overall, schemata seem to expose a high regularity and low complexity compared to non-instance candidates. The strong negative factors *rdsumv* and *onsets* disregard candidates with a large temporal extension and a high degree of non-simultaneity. Metric regularity (i.e., rhythmic regularity aligned to the metrical grid) has a strong positive influence, indicating a preference for a regular temporal organization.

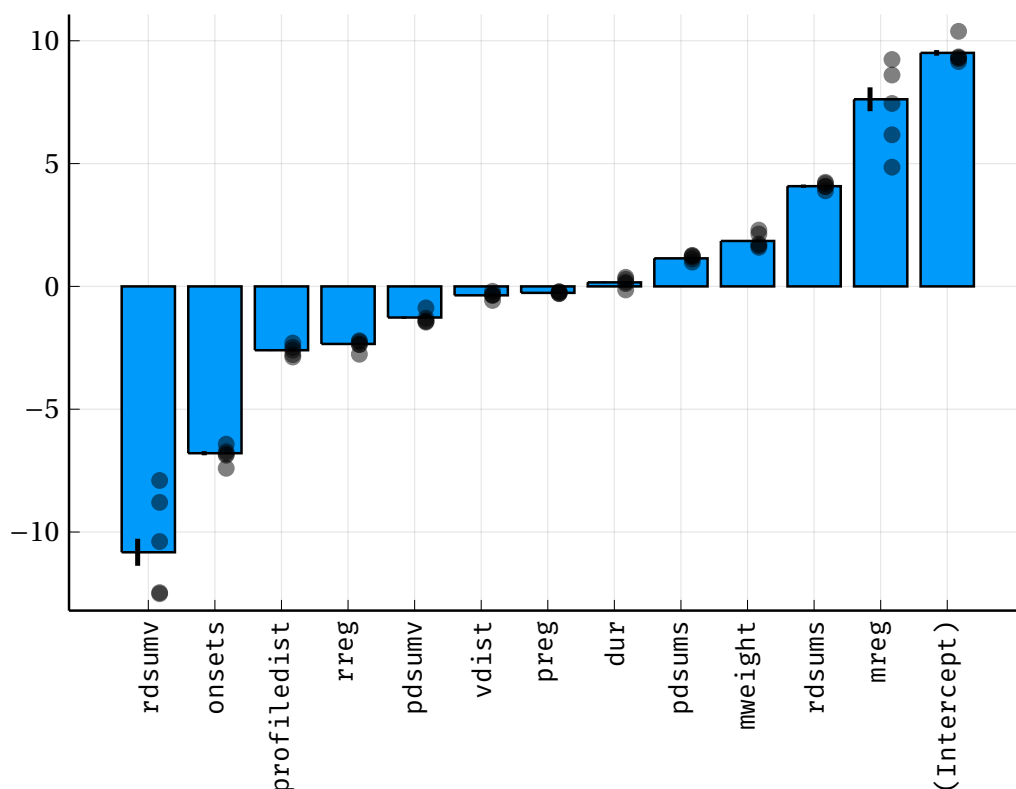


Figure 3.4 – The parameters $\vec{\beta}$ of the model trained on the full upsampled dataset (bars) and normalized by multiplication with the average value of the respective feature. Error bars indicate the 95% confidence interval of the fit. Black points indicate the normalized parameters for each model trained during cross validation.

The preference for simultaneity of the notes in the same stage is somewhat contradicted by the moderately positive influence of the `rdsums`, the average note displacement within stages. This is particularly surprising when looking at the distribution of this feature over instances and non-instances (Figure 3.2b), which shows that instances generally show less displacement than non-instances. One possible explanation of this phenomenon is that the combination of both features (`onsets` and `rdsums`) expresses a general preference for little displacement, but when the notes are non-simultaneous, then a higher distance is preferred, which may make the structural notes more recognizable.

Less important are features based on pitch (`profiledist`, `pdsum*`, `preg`, and `vdist`) as well as features that indicate basic salience (`dur` and `mweight`). Pitch features are likely of moderate to little importance because most of the relevant pitch-related information is already implied by the schema's interval structure. Interestingly, duration and metric weight (both properties that are taken from each note in isolation) play little to no role,

which is confirmed in Figure 3.2b. This indicates that local properties do not mark notes as structural, this role seems to depend *only* on how the note is used in relation to other notes.

3.6 Conclusion

As the results presented above show, distinguishing between incidental and structural note configurations based on a small number of musically and cognitively motivated heuristics works well in the vast majority of cases. Even if a number of misclassifications remain, a closer look at these cases provides valuable insights into the problem at hand. First, the main limitation of our approach is that the model assesses suggested schema instances individually, without considering, or comparing it to, alternative interpretations. In many cases, the main reason for human experts to reject a candidate does not seem to be a lack of plausibility of the match itself, but rather the availability of a “better explanation”, i.e. an alternative analysis of the match’s context that identifies a more plausible contrapuntal scaffold. This result is in line with the reduction-based approach of Katsiavalos et al. (2019). Since the features used in this study already proved useful for independent classification, they likely benefit from a general structural-analysis approach, in which schema instances are recognized in reductions of the musical surface.

A second insight concerns the idea of schema itself and its relation to a classification task. From a cognitive perspective, a schema does not need to be instantiated unambiguously or even completely. It is sufficient if listeners recognize the schema as the template for the surface events, or if they understand the composer’s intention to evoke the schema. In this regard, *discrete* binary classification into instances and non-instances may be as unattainable as it is undesirable, falling short of the complexity the relationship between schema and realization can exhibit.

3.7 Acknowledgements

The research presented in this paper is generously supported by the Volkswagen Foundation and Claude Latour. We also thank the anonymous reviewers for their helpful feedback.

4 Chord Types and Ornamentation¹

Abstract

Making sense of a musical excerpt is an acquired skill that depends on previous musical experience. Having acquired familiarity with abstract chords, a listener can distinguish tones in a musical texture that outline these chords (i.e., chord tones) from ornament tones such as neighbor or passing notes that elaborate the chord tones. However, music-theoretical definitions of chord types usually only mention chord tones, excluding typical ornaments. The aim of this project is to investigate (i) how knowledge about (chord-specific) ornaments can be incorporated into characterizations of chord types and (ii) how these characterizations can be acquired by the listener. To this end, we develop a computational model of chord types that distinguishes chord tones and ornaments and can be learned using Bayesian inference following methods in computational cognitive science. This model is trained on two datasets using Bayesian variational inference, comprising scores of Western classical and popular music, respectively, and containing harmonic annotations as well as heuristically determined note-type labels. We find that the proposed characterization of chords is indeed learnable and the specific inferred profiles match previous music-theoretic accounts. In addition, we can observe patterns in the use of ornaments, such as their distribution being related to the diatonic contexts in which chords appear and chord types differing in their predisposition to generate ornaments. Moreover, the differences in ornamentation distributions between the two corpora indicate style-specific peculiarities in the role and usage of ornaments. The different patterns of typical ornaments for specific chord types indicate that harmony and ornamentation are not independent.

¹To be published as:

C. Finkensiep, P. Ericson, S. Klassmann, and M. Rohrmeier (in preparation). “Chord Types and Ornamentation – A Bayesian Model of Extended Chord Profiles”. In: *Open Research Europe*

The Bayesian models were developed and implemented by CF after discussions between all authors. Data preprocessing was jointly implemented by PE, SK, and CF. The paper was mainly written by CF with contributions from the other authors, except for the second half of Section 4.1 (related work, written by CF and SK) and Section 4.2.3 (mainly written by PE).

4.1 Introduction

Listening to, performing, or thinking about music gives rise to a wide array of experiences that range from basic acoustic effects such as roughness, to cognitive aspects such as expectation and interpretation, to high-level cultural and social phenomena such as reference and emotional association. One such set of experiences arises from the need to *make sense* of the auditory input. Generally, making sense of a perceptual input, i.e., relating it to known categories and finding an explanation for how it came about, is a central task of a cognitive system (Chater, Tenenbaum, et al. 2006; Kersten et al. 2004). Music is no exception: When a listener is presented with a musical input, obtaining an interpretation of this input (consciously or intuitively) is crucial for the listener's experience. This includes a wide range of properties that are not directly represented on the musical surface, such as key and meter of a piece (Temperley 2007a), its internal structure (Abdallah, N. E. Gold, and Marsden 2016; Rohrmeier 2020a; Herff et al. 2021; Finkensiep, Déguernel, et al. 2020), or even the properties of underlying musical spaces (D. Hu and Saul 2009; Harasim, Moss, et al. 2021; Moss and Rohrmeier 2021).

One aspect of musical structure that is important for many styles of Western music is harmony (Aldwell, Schachter, et al. 2011), i.e., the organization of pitches into vertical sonorities and the succession of these sonorities. Interpreting a segment of music harmonically consists of matching the notes observed in the segment with a set of prototypical note configurations (*chord types*). The repertoire of chord types is generally style-specific (although some chord types can be shared across several styles), and, as such, must be obtained through some form of learning (Rohrmeier and Rebuschat 2012; Pearce 2018; Tillmann et al. 2000).

Western music theory generally characterizes chord types through a set of octave-equivalent pitches (the *chord tones*, CT) relative to a reference pitch, the *root* (Aldwell, Schachter, et al. 2011). However, musical segments that instantiate these chord types usually contain *non-chord tones* (NCT) as well as chord tones, which can often be understood as *ornaments* to the chord tones, e.g., suspensions, neighbor notes, or passing notes. Figure 4.1a, for example, shows a dominant-seventh chord on B resolving into an E minor chord. In addition to the chord tones (B, D \sharp , F \sharp and A), the musical surface contains the notes E and C \sharp , which can be interpreted as a suspension (the first E), and two neighbor notes (the second E and the C \sharp). Harmonic interpretations are often ambiguous. For example, the B⁷ chord could also be interpreted as a B major triad, which would make the A another non-chord tone, a passing note. The ornaments used to elaborate a chord can be specific to the chord type: A fourth suspension (Figure 4.1b), for example, is more typical for some chord types (e.g., major, minor, or dominant-seventh chords) than others (e.g., augmented triads).

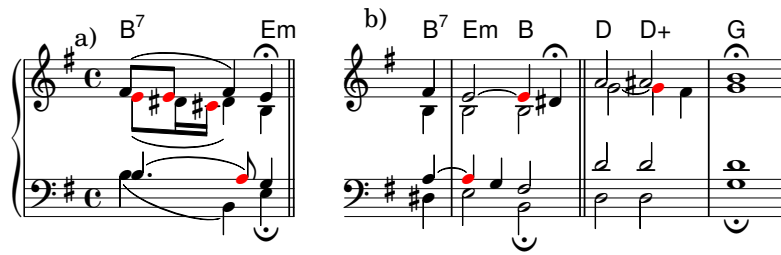


Figure 4.1 – Examples of ornaments in chords. a) A cadence from J. S. Bach’s chorale “Christ lag in Todesbanden” (BWV 278, m. 4). The dominant chord B⁷ contains a number of ornaments (marked in red), including a suspension and two neighbor notes. The 7th of the chord could also be argued to be an ornament (passing note) of a B major triad. b) A suspended 4th is a typical ornament in some chord types such as major and minor triads (here Em and B). In other chord types, such as augmented chords (D⁺), it is less common.

Consequently, this study addresses the following questions: How can a listener represent information about chord types (with both their chord tones and their typical ornaments) in a way that they can be linked with the musical surface (i.e., segments of notes)? How can these representations be learned from observations? How would these representations look like when obtained from real-world data? To this end, a computational model of chord-type representations is proposed, together with a learning process that simulates an ideal learner. We investigate the concrete chord types obtained from applying this model to specific musical corpora, what are typical ornaments, and how ornamentation differs between chord types.

We address these questions using Bayesian modelling (MacKay 2003), which is an established framework for the computational-level description of cognitive tasks (Chater, Tenenbaum, et al. 2006) such as reasoning (Jaynes 1988; Griffiths and Tenenbaum 2006; Oaksford and Chater 2018), perception (Kersten et al. 2004), and learning (Gopnik and Bonawitz 2015; Ullman and Tenenbaum 2020a). It combines a generic inference scheme with a problem-specific *model* – a characterization of the relationship between known (observed) and unknown (latent) entities in the form of a probability distribution. Often, this relationship is expressed as a *generative process* by which the observed entities are “generated” based on the latent entities. This perspective is useful when the latent entities cause or explain the observations, which in particular applies to harmonic types: A harmonic interpretation “explains” the notes observed in the musical input as either chord tones of a chord or ornaments. It should be noted that the generative process is not a model of the compositional process but rather of the listener’s presumed perspective on the relation between harmonies and notes. Inference and learning in the Bayesian framework amount to computing the posterior distribution over the latent variables. In the present study, information of the chord instances (notes, chord label, and the role of

each note) are assumed to be (mostly) known, while the characteristics of each chord type are unknown and must be learned from these instances.

The model of chord types presented in this study is based on tone profiles. Tone profiles have a long history in music psychology as representations of listeners' knowledge about abstract tonal hierarchies. Originally established in the probe-tone experiments performed by Krumhansl and Kessler (1982), tone profiles have since been used to describe tonal hierarchies in a variety of styles and traditions, including Jazz scales (Jarvinen 1995), North Indian rāgas (Castellano et al. 1984; Finkensiep, Widdess, et al. 2019), Turkish makams (Gedik and Bozkurt 2010), and Western classical and rock music (Vuvan and Hughes 2021). Tone profiles are known to reflect the distribution of pitch classes in pieces and corpora (Krumhansl 1990a; Huron 2006; Temperley and Marvin 2008) and have therefore found computational applications in key finding algorithms based on template matching (Krumhansl 1990a; Temperley 1999b; Bellmann 2006; J. Albrecht and Shanahan 2013) or generative probabilistic models (Temperley 2002b). Probabilistic models have also been used to infer tone profiles in an unsupervised fashion (D. Hu and Saul 2009; Harasim, Moss, et al. 2021; Moss and Rohrmeier 2021), simulating how a listener can learn to distinguish different modes based on statistical observations alone. Corpus studies have used distributional tone profiles to investigate the relation between tonal and metrical hierarchies (Prince and Schmuckler 2014), the development of major-minor tonality (J. D. Albrecht and Huron 2014; Harasim, Moss, et al. 2021) and the transposability of tone profiles (Quinn and C. W. White 2017).

In the present paper, the concept of tone profiles used to represent knowledge about the tonal structure of chord types rather than modes, which has precedents in music psychology (Parncutt et al. 2019) as well as computational models. Chord-type profiles are a common technique for chord identification from audio (Fujishima 1999; K. Lee 2006; Oudre et al. 2009; Demirel et al. 2019), usually called chromagrams in this context, but are also used in symbolic chord labelling models (Temperley 2009; Koops et al. 2020).

The relationship between chord tones and non-chord tones is usually not modeled using tone profiles but rather through a number of properties derived from theoretical accounts, such as metrical position, duration, and melodic context. Chuan and Chew (2011) and T. Hu and Arthur (2021) have developed NCT-identification algorithms for melodies that integrate such features using decision trees and logistic regression, respectively. Ju et al. (2017) use a neural network-based black-box approach with similar input features to identify NCTs in homophonic textures. Several chord-labeling systems have been developed that incorporate some form of NCT treatment based on a combination of the above features and theory-derived models of chord membership, which is either directly based on triads (Raphael and Stoddard 2004; Rhodes et al. 2009) or on some

form of third stacking (Temperley 1997; Sapp 2007; Mearns 2013). Some of these models allow the weights of different pitches in a chord to be learned, to some extent. For example, Rhodes et al. (2009) use a generative probabilistic model of pitch proportions that first decides the proportion of CTs vs NCTs (drawn from a Beta distribution) and then assigns the proportion of root, third, and fifth within the CTs (from a Dirichlet distribution). It thus resembles our model in that it contains an explicit decision to generate NCTs. However, all of the above models make assumptions about the possible structures of chords (either as fixed triads / seventh chords or as stacks of thirds), and all of them assume ornaments to be nominal NCTs (i.e., simply the negative of CTs) and to be equally important or likely.

Since the present study investigates the typicality of different ornaments for different chord types, the model presented here (a) allows ornaments to have different weights, and (b) separates the notion of ornaments and NCTs. The latter is important to capture the situation where a nominal chord tone is used as an ornament. For example, in a dominant-seventh chord, both 7 and 8 can be used as chord tones or as neighbor notes of the other (the 7 can even become a passing note that leads into the next chord). The resulting model consists of two independent tone profiles (Figure 4.2), one for chord tones (generating notes that are intended to be perceived as part of a chord) and one for ornaments (generating notes that are not intended to be part of the chord). As a consequence, the model does not make any assumptions about the internal structure of either the chord tones or the ornaments (e.g., stacking of thirds). This information is entirely represented in the data.

4.2 Method

4.2.1 Model

Adopting the Bayesian framework (MacKay 2003), this article presents a minimal probabilistic model that links surface notes, their harmonic interpretations, and the properties of chord types, and in particular encodes an explicit distinction between chord tones and ornaments. At the core of this model lies a simple process that generates a single note based on a given chord type (Figure 4.3). Before a note's pitch is selected, a Bernoulli trial determines the type of the note, chord tone or ornament. The pitch is then drawn from a chord-specific categorical distribution corresponding to its type. Thus, every chord type is characterized by two categorical distributions, one for its chord tones and one for its ornaments. These two distributions are independent, so that a single pitch need not have a fixed role as either a chord tone or an ornament. Instead, both distribu-

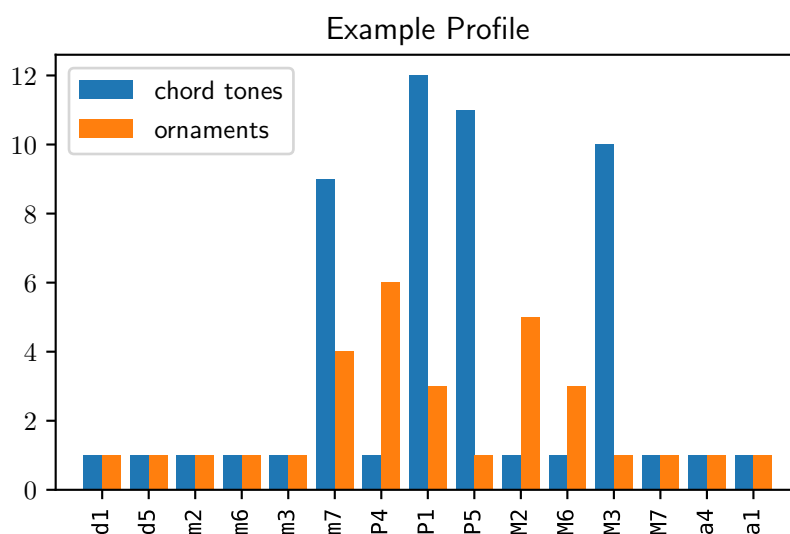


Figure 4.2 – A schematic example of a chord type (a dominant-seventh chord) as represented in the model. The chord-tone profile (blue) gives a high weight to the nominal chord tones (root=P1, P5, M3, m7). The ornament profile (orange) gives a high weight to pitches that are used as ornaments (P4, M2, M6, but also m7 and P1). The x-axis is arranged according to the line of fifths around the root (P1). Note that the absolute heights of the blue bars and the orange bars cannot be directly compared, as they indicate certainty *within* each profile rather than the prevalence of ornaments over chord tones in general. The ornament-chord tone ratio is captured in a separate set of model parameters.

tions can potentially generate every pitch (though usually with different probability). Furthermore, categorical distributions ignore any internal structure of their support, so the model is agnostic to theoretical assumptions about the internal structure of chords (such as always consisting of stacks of thirds). In particular, this allows chord types to encode “optional” chord tones that are not nominal chord tones but are still frequently used as stable, non-ornamental pitches, e.g., major 6ths, 7ths or 9ths with major chords in Pop or Jazz.

The model of harmonic types presented here is based on the idea that chords are latent entities that are related to the notes observed on the musical surface by a generative process. Since several surface constellations are possible for the same chord, this generative relationship is probabilistic and can be expressed as a conditional distribution $p(N|c)$, where N stands for the observed notes and c denotes the chord. A harmonic type is then represented by a set of parameters ϕ_c that characterize this conditional distribution for a given chord type c . In this case, ϕ_c consists of a chord-tone profile ϕ_c^{ct} , an ornament profile ϕ_c^{or} , and a parameter for the prevalence of chord tones over ornaments θ_c . The

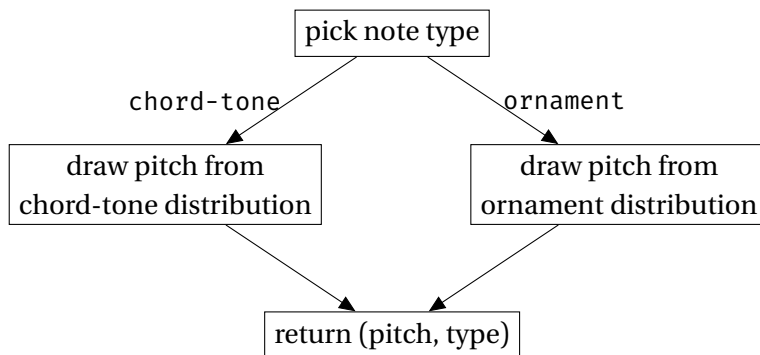


Figure 4.3 – The core process that generates a single note in a chord. The specific probability distributions here depend on the chord type, which establishes a relation between the chord type and the notes it generates.

quantity of interest is the *posterior* distribution of these parameters conditioned on a dataset of chord instances

$$p(\phi \mid \vec{N}, \vec{c}), \quad (4.1)$$

in other words, the properties of chord types that can be inferred from this dataset.²

Generally, the generative process from a chord type to the surface involves determining the exact occurrences of the surface notes with their positions in pitch and time, including both chord tones and ornaments to these chord tones. Since the focus of this study is on the pitch distributions of chord tones and ornaments, we adopt a simpler “bag of notes” representation (Harasim, Moss, et al. 2021), where temporal information and octaves are discarded and only the number of occurrences of each tonal pitch class³ is retained. In addition to its pitch, each note has one of two *note types*, chord-tone and ornament, according to the note’s role in the chord instance. The note types in the ground truth are derived using music theoretical knowledge about the chord types and a set of heuristics as described in Section 4.2.3. Since these heuristics may not determine the type of a note with certainty, it can also take the value *unknown*. Chord labels (i.e., the annotated chord types of specific chord instances) are represented by a categorical variable over some vocabulary of chord types that depends on the annotations given in the respective datasets. The root of the chord is not represented as an absolute pitch or

²Note that from the Bayesian perspective, we do not obtain a specific value for ϕ (i.e., a point estimate) but rather a distribution that quantifies the uncertainty about the true value of ϕ . However, this posterior distribution can still be characterized through a pitch profile, where the relative weight of each pitch correspond to the expected value of ϕ while their absolute magnitude indicates the variance of the posterior distribution.

³Tonal pitch classes (Temperley 1997) respect the spelling of a pitch, i.e., C \sharp and D \flat are considered different. This is in contrast to neutral pitch classes (often represented by numbers from 0 to 11), which identify enharmonically equivalent notes. In this text, “pitch class” refers to tonal pitch classes.

chord label	relative pitch class	note type	occurrences
dominant-seventh	unison (= root, P1)	chord-tone	3
	fifth (P5)	chord-tone	2
	major 3rd (M3)	chord-tone	2
	minor 7th (m7)	chord-tone	1
	fourth (P4)	ornament	2
	major 2nd (M2)	ornament	1

Table 4.1 – Representation of a chord instance as a data point. A chord label (dominant-seventh) is combined with a bag of notes, given by pairs of pitch classes (relative to the root) and note types. This example shows how the B⁷ chord in Figure 4.1.

pitch class, instead the pitch classes of all notes are encoded as interval classes relative to the root of the chord, which results in transpositionally invariant chord types. For example, the B⁷ chord shown in Figure 4.1a would be encoded as shown in Table 4.1. Note that, in principle, the same pitch class can occur several times with different note types, which would indicate that there are two notes in the chords with the same pitch class but different functions.

For the generation of the notes, we assume a very simple process that draws notes independently from two categorical distributions over pitch classes, one for chord tones and one for ornaments. For every chord instance, the type of the chord c and the number of notes n are chosen. Then, the note type t of each note is determined by a (biased) coin flip, and its pitch is drawn from the corresponding categorical distribution ϕ_c^t of the chord type. Finally, to account for unknown note types, another coin is flipped to determine whether the notetype is observed or not. The process for a single chord instance can be summarized as follows:

1. Choose the chord label $c \sim \text{Categorical}(\chi)$.
2. Choose the number of notes $n \sim \text{Poisson}(\lambda) + 1$.⁴
3. For each note $i \in 1, \dots, n$:
 1. Choose the note type $t_i \sim \text{Bernoulli}(\theta_c)$.

⁴The notation $n \sim \text{Poisson}(\lambda) + 1$ indicates that n is obtained by drawing a sample from a Poisson distribution and adding 1. This is to avoid that a chord with zero notes is generated. In our experiments, the number of notes is always observed, so choosing n is not really necessary. However, we include it in order to have a fully generative model. Note that since n is observed, the choice of its distribution is independent from all other variables and has no effect on the results of this study.

2. Choose the pitch class depending on the note type:

$$p_i \sim \begin{cases} \text{Categorical}(\phi_c^{ct}) & \text{if } t_i = \text{chord-tone} \\ \text{Categorical}(\phi_c^{or}) & \text{if } t_i = \text{ornament.} \end{cases}$$

3. Decide whether to observe the note type: $o_i \sim \text{Bernoulli}(\omega)$.
4. Return (p_i, t_i) if $o_i = \text{true}$, and $(p_i, \text{unknown})$ if not.
4. Count the generated pairs.

The generative process of notes resembles the process behind a multinomial distribution over pairs of pitch classes and note types, with the difference that here additional latent variables (t_i and o_i) are involved. However, since those variables are Bernoulli-distributed, they can be marginalized out analytically:

$$\begin{aligned} p(p_i, \text{chord-tone} | c) &= p(o_i = \text{true})p(t_i = \text{chord-tone} | c)p(p_i | c, \text{chord-tone}) \\ &= \omega \cdot \theta_c \cdot \phi_{cp_i}^{ct} \\ p(p_i, \text{ornament} | c) &= p(o_i = \text{true})p(t_i = \text{ornament} | c)p(p_i | c, \text{ornament}) \\ &= \omega \cdot (1 - \theta_c) \cdot \phi_{cp_i}^{or} \\ p(p_i, \text{unknown} | c) &= p(o_i = \text{false}) \sum_t p(t_i = t | c)p(p_i | c, t) \\ &= (1 - \omega) \cdot (\theta_c \phi_{cp_i}^{ct} + (1 - \theta_c) \phi_{cp_i}^{or}). \end{aligned}$$

The note-generation process can thus be replaced by a simple multinomial over $P \times \{\text{chord-tone}, \text{ornament}, \text{unknown}\}$, where the combined parameters ϕ_c can be expressed in terms of $\phi_c^{ct}, \phi_c^{or}, \theta_c$:

$$\phi_c = \begin{matrix} \omega & \cdot & (\theta_c \cdot \phi_c^{ct} & \circ & (1 - \theta_c) \cdot \phi_c^{or}) \\ \circ & (1 - \omega) & \cdot & (\theta_c \cdot \phi_c^{ct} & + & (1 - \theta_c) \cdot \phi_c^{or}), \end{matrix} \quad (4.2)$$

where \circ denotes vector concatenation and ω is the probability that the note type is observed. As a result, the generative process can be simplified as:

1. Choose the chord type $c \sim \text{Categorical}(\chi)$.
2. Choose the number of notes $n \sim \text{Poisson}(\lambda) + 1$.
3. Choose note counts $N \sim \text{Multinomial}(\phi_c, n)$.

To complete the model, we assume the following prior distributions over the parameter variables:

1. Choose the chord probabilities $\chi \sim \text{Dirichlet}(0.5)$.
2. Choose the mean number of notes $(-1) \lambda \sim \text{Gamma}(3, 1)$.

3. Set the type observation probability $\omega := 0.5$.
4. For each chord type h :
 1. Choose the probability of generating a chord tone $\theta_h \sim \text{Beta}(1, 1)$.
 2. Choose $\phi_h^{ct} \sim \text{Dirichlet}(0.5)$.
 3. Choose $\phi_h^{or} \sim \text{Dirichlet}(0.5)$.

For all categorical parameters we use an uninformative Dirichlet prior (Jeffreys' prior, where each hyper-parameter is 0.5). The prior $\text{Beta}(1, 1)$ for each θ_h is uniform between 0 and 1. Since it is always known whether a note type is observed (chord-tone or ornament) or not (unknown), the variable o_i is an observed variable and ω can take an arbitrary value without affecting the other parameters. Likewise, λ is conditionally independent from the other parameters since the number of notes in each chord is observed, so the choice of $p(\lambda)$ and $p(n | \lambda)$ does not affect the rest of the model. The full model is summarized in Figure 4.4.

4.2.2 Inference

In this study, the main focus is on the posterior distributions of the latent variables, in particular the parameters of the chord-tone and ornament distributions ϕ_h^{ct} and ϕ_h^{or} for each chord type h given the observed chords D . These marginal posterior distributions $p(\phi_h^{ct} | D)$ and $p(\phi_h^{or} | D)$ are approximately inferred using variational inference (Blei et al. 2017) with a mean-field variational family $q(\phi^{ct}, \phi^{or}, \lambda, \chi, \theta, t_{ij})$ that assumes the posterior distributions of each variable to be independent:

$$q(\phi^{ct}, \phi^{or}, \lambda, \chi, \theta, t_{ij}) = q(\lambda)q(\chi) \prod_h q(\theta_h)q(\phi_h^{ct})q(\phi_h^{or}) \prod_{i,j} q(t_{ij}). \quad (4.3)$$

The distribution family of each latent variable corresponds to its respective prior family in the model, i.e., gamma for λ , beta for θ , Bernoulli for t_{ij} , and Dirichlet for χ , ϕ_h^{ct} , and ϕ_h^{or} . Variational inference optimizes the parameters of these distributions to make q as similar as possible to the true posterior by minimizing the KL-divergence between the two: $KL(q(\text{latent}) \parallel p(\text{latent} | \text{observed}))$.

The parameters of the Dirichlet posteriors for the chord-tone and ornamentation probabilities have a straightforward interpretation as tone profiles. Each parameter α_p stands for one pitch class p , with the magnitude of the parameter indicating the importance of the pitch class in the chord as a chord-tone (for ϕ_h^{ct}) or ornament (for ϕ_h^{or}). Recall that a sample from a Dirichlet distribution corresponds to the parameters ϕ of a categorical or multinomial distribution, i.e., the unknown true pitch probabilities that characterize each chord type. The expected value of these parameters is given by normalizing the

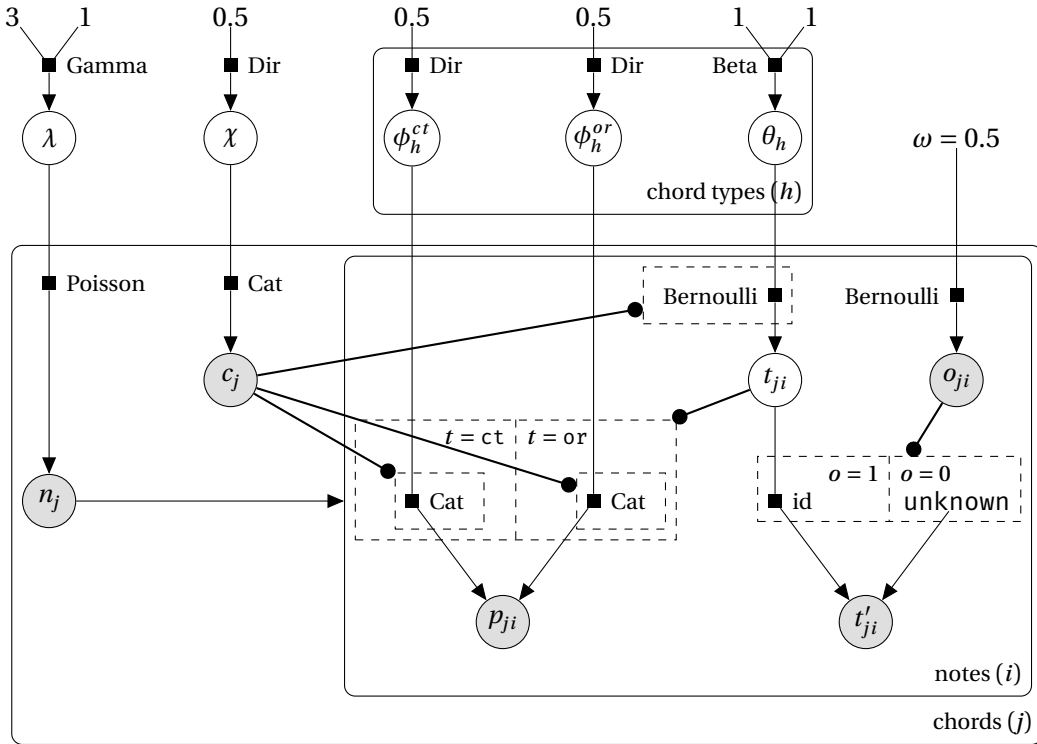


Figure 4.4 – A factor graph of the the chord model. Variables are shown in circles (grey if observed), the distribution a variable is drawn from is indicated by an arrow between the parameters of the distribution and the sampled variable. A rounded rectangle denotes that several copies of the included variables exist, one for each chord type, chord instance, or note (corresponding to “for each” statements in the generative process). Dashed rectangles indicate the a choice of parameter based on another variable, e.g., the choice of the chord type c_j determines which ϕ_h^{ct} (or ϕ_h^{or}) is used to generate a note, namely $\phi_{c_j}^{ct}$ (or $\phi_{c_j}^{or}$).

Dirichlet parameters:

$$\mathbb{E}[\phi_i] = \frac{\alpha_i}{\sum_j \alpha_j}. \quad (4.4)$$

The combined magnitude of the parameters determines the variance of the Dirichlet distribution, so large parameters indicate a high confidence that the true distribution is close to the expected value $\hat{\phi}$. The parameters $\vec{\alpha}$ can be interpreted as *pseudo counts* that express how often each pitch class has been “observed” as a chord-tone or ornament of a given chord type.

The model and the variational family are implemented as probabilistic Python programs using the Pyro framework (Bingham et al. 2019). Inference is performed using stochastic gradient descent on the evidence lower bound (ELBO) between p and q with respect to the parameters of q . The parameters were optimized using the Adam algorithm (learning

rate = 0.01, $\beta_1 = 0.95$, $\beta_2 = 0.999$).

Since the final parameters are quite large and the ELBO gradient is rather flat once the parameters have the correct proportions (but not magnitude), this algorithm can take a lot of time to converge when started from parameters close to 0 (i.e., using the parameters of the priors). However, since we use conjugate priors in the model and the corresponding distributions in the variational family, we initialize the parameters of q with estimates of the final parameters based on conjugacy. For some of the latent variables (λ and χ), the posterior parameters can be computed exactly from the data since they are conditionally independent from the other variables and have true conjugate priors. The other parameters are not conjugate due to the presence of unknown note types. They can be initialized approximately by observing only the notes with known types: The Dirichlet parameters for each ϕ_h^{ct} and ϕ_h^{or} are initialized with the counts of known chord tones and ornaments, respectively. The beta-distribution parameters for each θ_h are estimated by counting the total number of known chord tones and known ornaments for each chord type. During training, these parameters adapt to account for the presence of unknown note types, while the exactly initialized parameters for λ and χ do not change at all. As a result, the model converges relatively soon after ca. 200 iterations on both datasets (we use the parameters after 350 iterations to guarantee convergence). Because of this, we do not use subsampling but evaluate the full dataset in each iteration, which reduces the variance of the stochastic gradient and stabilizes the parameters of the converged model.⁵

4.2.3 Datasets and Preprocessing

The data used for training the model is not immediately available in existing corpora. In particular, notes are not explicitly distinguished as chord tones or ornaments. We implement a music-theoretically inspired heuristic for estimating this information from existing harmonically annotated scores. Note, however, that the model described in Section 4.2.1 is independent from this heuristic and could be applied to a dataset with manual note-type annotations.

With this preprocessing step, the model is applied to two datasets: The first (ABC+) is a collection of scores from Western common-practise music, professionally annotated with harmonic labels in the DCML chord-annotation standard (Neuwirth, Harasim, et al. 2018; Hentschel, Neuwirth, et al. 2021; Hentschel, Moss, et al. 2021). The second

⁵Training takes around 5 minutes for the ABC+ corpus and 10 minutes for the EWLD corpus on an Intel i7-7600 CPU using a single thread. Using a GPU or multi-threading may make the inference process even faster, but would make it not exactly reproducible due to concurrent random number generation.

current note n	other present notes		note type
any	$\nexists m.nb(m, n)$	(no neighbor)	chord-tone
$ct(n)$	$\nexists m.ct(m) \wedge nb(m, n)$	(no chord-tone neighbor)	chord-tone
$\neg ct(n)$	$\exists m.ct(m) \wedge nb(m, n)$	(has chord-tone neighbor)	ornament
otherwise			unknown

Table 4.2 – The note-type heuristic used to infer if an encoded pitch represents a chord-tone or an ornament, if known.

one (EWLD) a collection of user-submitted lead sheets (melodies with chord symbols) from a variety of mostly Western styles (including pop and jazz music) from the former Wikifonia database (Simonetta 2018). The preprocessed datasets are available online together with the model code.⁶

Estimating Note Types

An important part of the preprocessing of the data is in deciding whether a specific note is an ornament or a chord tone. This can be estimated automatically from the harmonic ground truth annotations. The basic procedure is the same for both datasets: based on the annotated chord label, a check is made using the current note, the other notes present during the current chord⁷, and the nominal notes of the present chord type. We assume that an ornament cannot exist without a chord tone it refers to. Thus, if a note has no “neighbor” (a note within step distance) in the current chord, it cannot be an ornament and is considered to be a chord-tone, regardless of its relationship to the current chord type.⁸ Further, if a note is not one of the prototypical chord notes, and it has a chord note neighbor present, it is assumed to be an ornament. Finally, if the current note is one of the prototypical chord notes and has no chord-tone neighbor, it is also encoded as a chord-tone. All other cases – a chord tone with a chord-tone neighbor (e.g., a seventh and a root, or a third and an eleventh in some chord types), or two neighboring notes neither of which is a nominal chord tone (e.g., a sixth and seventh degree with neither root nor fifth present) – are ambiguous and are encoded as unknown by the heuristic. Table 4.2 shows a summary of the heuristic.

⁶https://github.com/DCMLab/probabilistic_harmony_model/

⁷For the purposes of this heuristic, “present” means overlapping at some point – a long note that starts during one chord and ends during another is present during both.

⁸The assumption that a note that occurs without neighbors is a chord tone is made to allow a profile to learn chord tones that are not explicitly mentioned in the theoretical description of the chord. Otherwise, the profiles would simply recover the theoretical definition of a chord type.

ABC+ Dataset

The first dataset consists of a collection harmonically annotated corpora created at the Digital and Cognitive Musicology Lab (DCML) at École Polytechnique Fédérale de Lausanne, consisting of 747 full scores or piano reductions of Western classical music, from the Baroque to the Romantic period. The dataset includes the published annotations of the Beethoven string quartets (Neuwirth, Harasim, et al. 2018) and the Mozart piano sonatas (Hentschel, Neuwirth, et al. 2021). These pieces have been harmonically annotated, reviewed, and corrected by music theory experts. The remaining pieces have been annotated using a similar procedure but are not yet completely reviewed or published. However, since this study reduces the whole corpus to a small number of chord profiles, we deem remaining annotation errors and inconsistencies to be irrelevant for the problem at hand. Table 4.3 gives an overview of the collections included in the ABC+ dataset.

The harmonic annotation standard used in the ABC+ corpus allows for a wide variety of harmonic annotations, including suspensions, retardations, pedal points, local key changes, and various inversions, much of which is irrelevant to, or hard to reconcile with the model described above. The preprocessing script thus extracts the chord-type column from the data, where each chord is given a chord type from the following chord vocabulary: minor, major, diminished and augmented triads, all four combinations of minor and major thirds and sevenths, the half-diminished, fully diminished, and augmented sevenths chords, and the three varieties of augmented sixth chords (Italian, German, and French). Notably, suspended chords are not considered to be distinct chord types, nor are inversions of the same chord type considered distinct from each other. Each note is encoded as a (tonal) interval class relative to the chord root, and its note type is estimated according to the heuristic described in Section 4.2.3. In total, this yields 157,352 chords with 1,031,228 notes.

EWLD Dataset

The second dataset is extracted from the Enhanced Wikifonia Leadsheet Dataset (EWLD, Simonetta 2018), which in turn is a subset of the former Wikifonia database. It contains a collection of user-submitted transcriptions and lead-sheets of pieces from medieval to contemporary music, with an emphasis on jazz and pop music, submitted to the now defunct Wikifonia website. The EWLD is a subset of Wikifonia that focuses on monophonic scores and is described in (Simonetta et al. 2018). Discarding pieces with parsing issues, the dataset consists of 5,075 pieces, containing 199,050 chords with

Composer	Collection	Pieces
Bach	Suites	89
Beethoven	Sonatas	62
Chopin	Mazurkas	50
Corelli	Sonatas	53
Couperin	Concerts Royeaux	26
	Gouts réunis	58
	L'art de toucher	13
Debussy	Suite Bergamasque	4
Dvorak	Silhouettes	12
Gesualdo	Libro 6	3
Grieg	Lyrical Pieces	66
Kozenluh	Sonatas	48
Liszt	Années	11
Medtner	Tales	19
Mendelssohn	String Quartets	24
Monteverdi	Madrigals	27
Mozart	Sonatas	54
Pleyel	Quartets	2
Ravel		3
Schubert	Winterreise	24
Schumann	Kinderszenen	13
	Liederkreis	12
Schütz	Kleine geistliche Konzerte	55
Sweelinck	Fantasia Cromatica	1
Tchaikovsky	Seasons	12
W.F. Bach	Sonatas	4
Wagner		2

Table 4.3 – Subcorpora of the ABC+ corpus.

657,482 notes.⁹

A major difference from the ABC+ corpus stems from the fact that the chord symbols of a lead sheet often act as instructions for how to play an accompaniment, rather than a music-theoretical interpretation. As for the ABC+ corpus, it is likely that some harmonic annotations are questionable, and in general the annotation standards and choices in this corpus vary much more widely than in the ABC+ corpus. The chord labels in the EWLD corpus lack the extraneous information of the ABC+ corpus, are generally less strictly applied, and not always interpretable. Moreover, the MusicXML standard allows for a large chord vocabulary, with essentially unbounded additions, subtractions and alterations of arbitrary scale degrees, as well as alternate bass notes and inversions. As an additional preprocessing step, we condense this variety down to the following vocabulary of 25 chord types, entirely informed by the `kind` element of the MusicXML harmony elements:

- the four triads (major, minor, diminished and augmented) and the power chord (omitting the third);
- the suspended fourth and second chords (both replacing the third);
- major, minor and dominant-seventh chords, each optionally extended up to their corresponding, 9ths, 11ths or 13ths;
- augmented, half-diminished and fully diminished seventh chords, as well as the minor triad with a major seventh;
- major and minor sixth chords.

Out of these, there are very few exemplars in the data of the major and minor 13th chords in particular, with the major-minor, power, and the minor and dominant 11th chords also rarely appearing. There are no instances at all of the major 11th chord in the data.

4.3 Results and Discussion

The exact numeric results of our experiments are provided in human- and machine-readable form in the supplementary material.¹⁰ In this section, we will summarize and discuss some observations that can be made from the inferred parameters. The parameters that are not discussed here are shown in the appendix.

⁹We used the code provided by the EWLD authors at <https://framagit.org/sapo/OpenEWLD> to create the dataset. Due to licensing issues, we cannot provide the full source data in the supplementary material. However, we provide the preprocessed data together with a list of all EWLD files that were used.

¹⁰https://github.com/DCMLab/probabilistic_harmony_model/

4.3.1 Chord Profiles

The posterior distributions of the chord tones ϕ^{ct} and ornaments ϕ^{or} are shown in Figure 4.5 for the chord types common to both corpora (The remaining distributions are shown in Figure A.1 in Appendix A). Each bar shows the Dirichlet parameter that corresponds to a certain pitch in a certain chord type (either as a chord tone or as an ornament) and expresses the prevalence of that pitch in chords of that type. The true (but unknown) pitch probabilities within a chord type can be seen as a sample from this Dirichlet distribution. Since Dirichlet distributions with large parameters have a low variance, the true probability distributions are expected to have the same *proportions* as the posterior, but at the same time the magnitude of the parameters expresses the model’s uncertainty about these proportions. For example, frequent chords generally have larger parameters (resulting in low variance and high certainty) than rare chords (higher variance, lower certainty), because seeing more examples of a chord type leads to higher confidence in the underlying pitch probabilities. Therefore, Dirichlet parameters can be understood as the combination of a set of proportions over pitches (i.e., a tone profile) and a measure of certainty about these proportions.

The posterior chord profiles inferred from the ABC+ corpus (left column of Figure 4.5) show a strong prevalence of the chord tones as predicted by the theoretical definitions of the chord types. For example, the major triad has large chord-tone parameters for the root, the major 3rd, and the perfect 5th, but small parameters for all other chord tones. The ornaments are distributed around the chord tones, forming a bell-like shape on the line of fifths for most chord types, a pattern that is typical for tonal music (Moss, Neuwirth, and Rohrmeier 2022). At the same time, ornaments and chord-tones are often (but not always) mutually exclusive, i.e., most tones have a clear interpretation as either chord tone or ornament, especially when the tone is rather common in the chord type. Together, both observations suggest that in a common-practice setting, ornaments are mostly taken from the diatonic context in which the chord occurs, excluding the chord tones themselves. Diatonic collections form contiguous segments on the line of fifths, and most chord types can be associated with different diatonic collections. This is particularly visible with chord types that are often borrowed in a parallel key, such as the dominant-seventh chord on V (borrowed in minor) or half-diminished chord on II (borrowed in major). Both profiles use the minor and major versions of the second and the sixth as ornaments, where one pair comes from the “native” key and the other pair from the parallel key. The combination of several diatonic segments around the chord tones then could give rise to the bell-shaped distribution of ornaments: the central pitches are more common because they are shared between different diatonic contexts associated with the chord types.

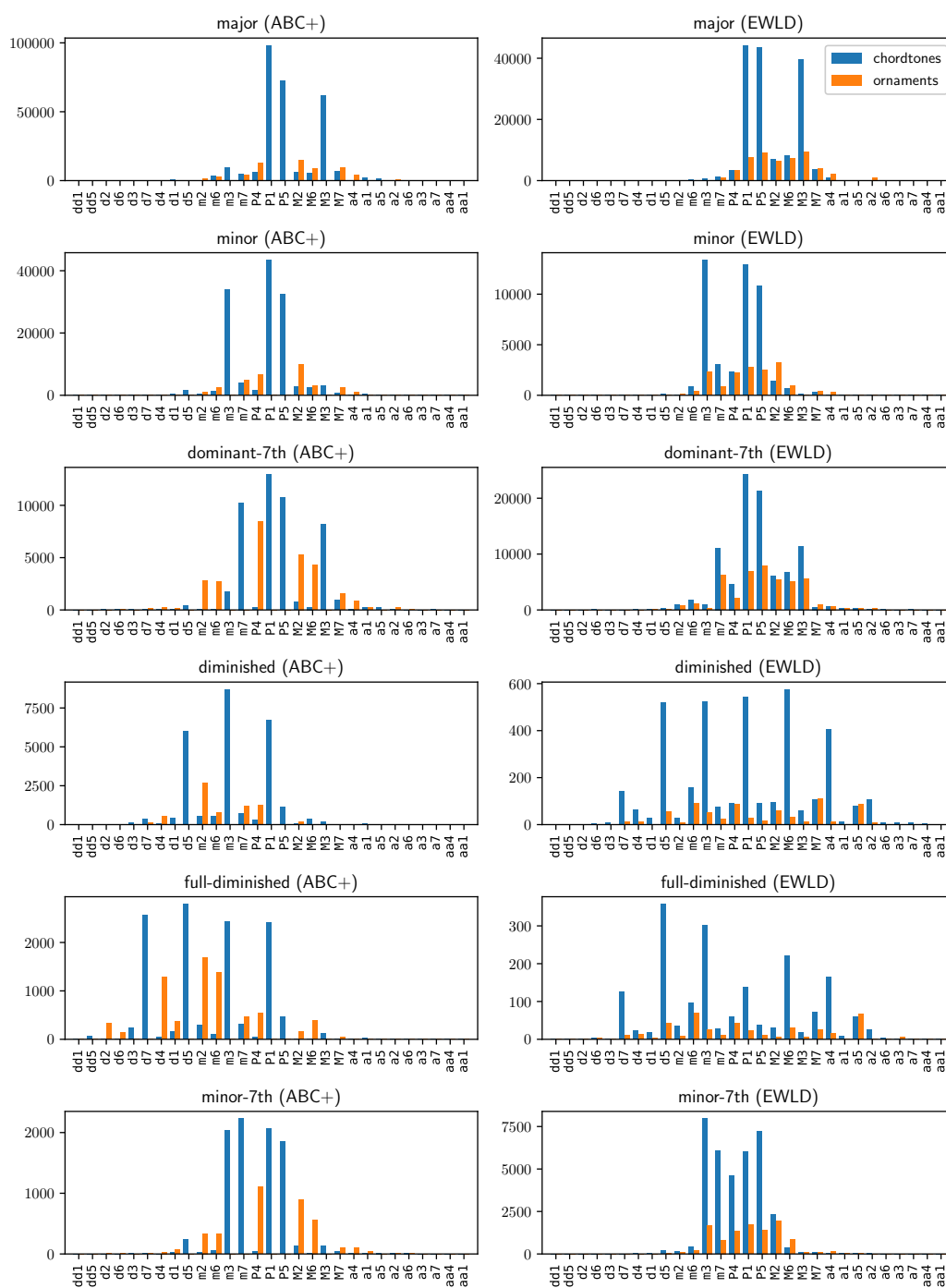


Figure 4.5 – The posterior distributions of the chordtones ϕ^{ct} (blue, left-leaning bars) and ornaments ϕ^{or} (orange, right-leaning bars) of the chord types that are common to the ABC+ (left) and EWLD (right) corpora. Pitches are ordered according to the line of fifths and expressed as intervals relative to the root (P1, unison). (Continues on next page.)

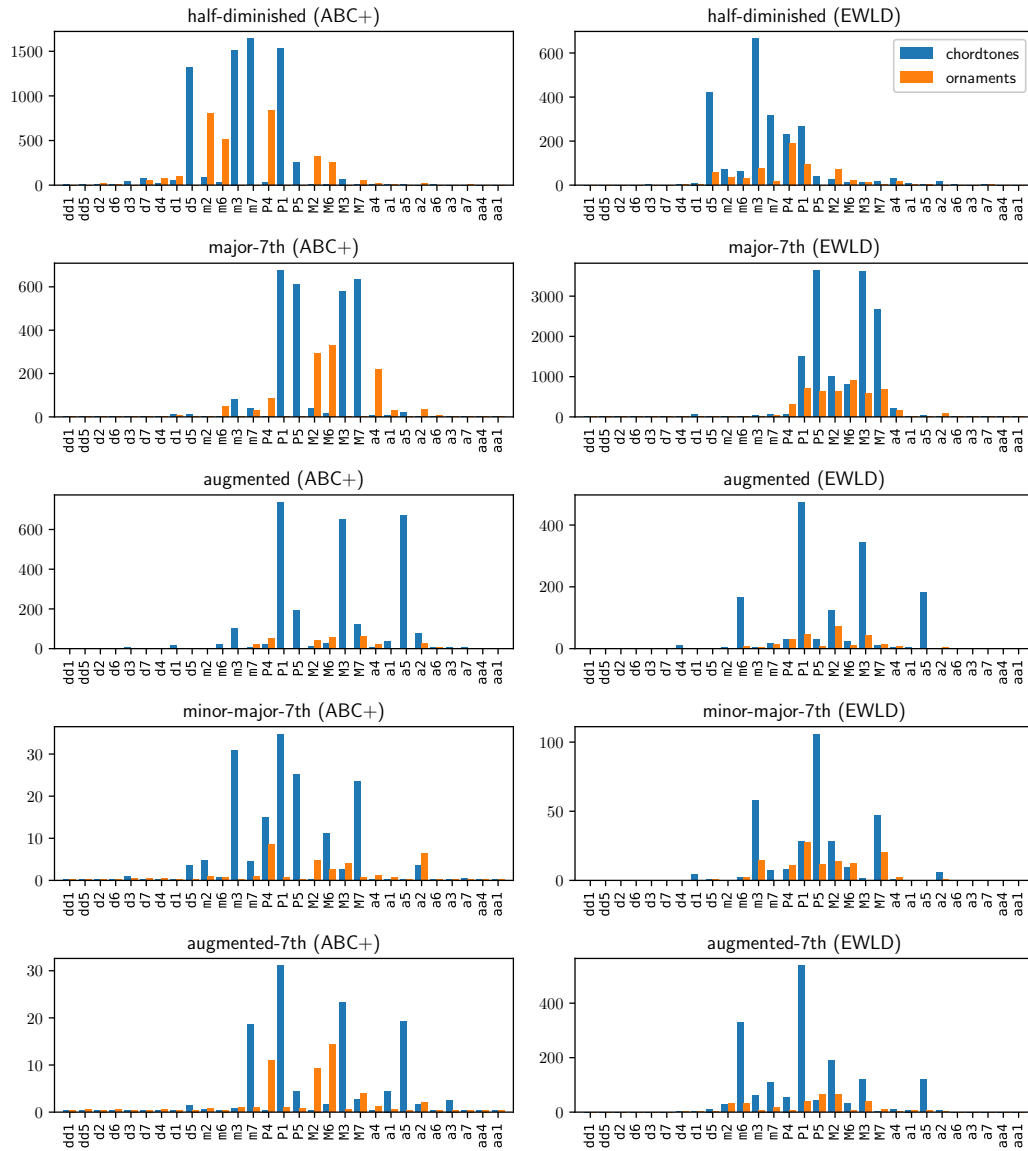


Figure 4.5 – (cont.)

Chapter 4. Chord Types and Ornamentation

It remains unclear whether certain ornaments are specific to a chord type as such, or whether ornaments are generally drawn from the local key (which in turn is correlated with the chord). The latter interpretation is consistent with the traditional view that ornamentation in classical music is mainly a result of contrapuntal elaboration, for which the mode is the main organizational principle (Aldwell, Schachter, et al. 2011). A few exceptions to the general diatonic bell shape (e.g., a strong m2 for the half-diminished chord) could point to chord-specific ornaments, but in many cases the prototypical ornaments of a chord type are at the same time in the center of its possible diatonic contexts (such as the 4th in the dominant-seventh chord).

The chord profiles of the EWLD corpus show similar patterns (e.g., the prevalence of the theoretical chord tones) but with some notable differences to the ABC+ corpus. For instance, ornaments are not mutually exclusive with chord tones. In fact, most of the common ornaments have a similarly high prevalence as a chord tone, while the theoretically predicted chord tones also have a certain prevalence as an ornament (though usually not as high as as a chord tone). This failure to clearly separate the classes might be due to the data consisting of melodies, which usually do not cover all chord tones of a harmony and preferable move in steps. Thus, the heuristics that were used to estimate the type of a note will identify fewer clear cases and rather report an “unknown” note type, from which the model cannot learn the difference between chord tones and ornaments. In addition, there might be an effect due to the different role of chords in pop and rock music compared to classical music, with a more flexible coordination between melody and harmony (Temperley 2007b) and less strong tonal hierarchies (Vuvan and Hughes 2021).

Most of the common chord types in the EWLD corpus exhibit a similar pattern for ornaments as in the ABC+ corpus, i.e., a continuous line-of-fifths region around the chord tones. However, in many cases, the span of ornament pitches is more narrow and less bell-shaped (e.g., for dominant-seventh, minor-seventh, half-diminished, or major-seventh chords), which indicates that chords are more strongly tied to a single fixed diatonic collection than in the ABC+ corpus. This phenomenon could be linked to the chord-scale idea from Jazz theory, which links certain chord types to fixed, local scales rather than an overarching mode that is shared by several chords (Levine 2011; Nettles and Graf 1997). A related phenomenon can be observed for the symmetric diminished and diminished-seventh chord, which rather than a diatonic region seem use an octatonic scale as the reservoir for ornaments. This is consistent both with chord-scale theory and with the application of Tonfeld theory to Jazz harmony (Rohrmeier and Moss 2021). Both observations together suggest that the relation of chords and ornaments differs between classical and modern styles: Whereas classical styles are rather flexible in associating chord types and diatonic contexts, using the same chord

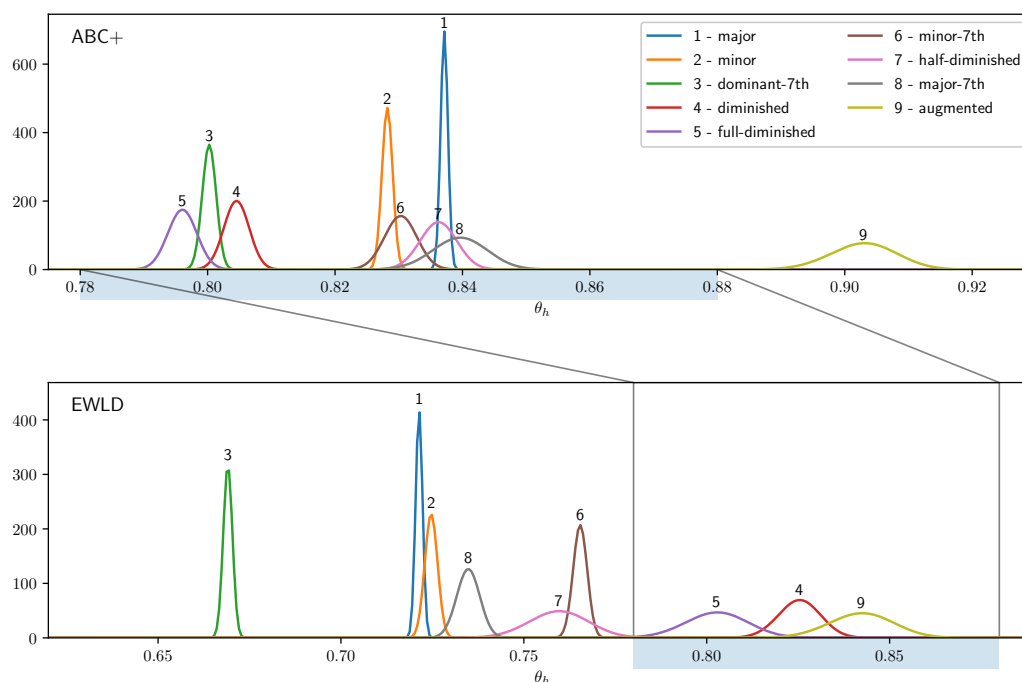


Figure 4.6 – The posterior distributions of θ_h for common chord types. The x-axis ranges are chosen to visually align corresponding chord types between the datasets (the highlighted regions cover the same range).

types on different scale degrees or even borrowing from other modes, modern styles more strongly associate chords with fixed scales. While this finding seems to be at odds with the melodic-harmonic-divorce hypothesis (Temperley 2007b), it should be noted that the EWLD corpus mixes different styles, which might follow different principles.

Another difference between the ABC+ and the EWLD corpus is an increased use of enharmonic equivalence in the latter, which can be observed most strongly for chords that are symmetric under enharmonic equivalence, i.e., (full-)diminished and augmented chords. While the corresponding profiles for the ABC+ corpus mostly use a single enharmonic interpretation of a pitch (e.g., a diminished seventh rather than a major sixth in the full-diminished chord), the EWLD profiles allow for the enharmonic equivalents of the classically preferred variants. For example, the profiles of the diminished and full-diminished chords are almost symmetric around the root for the EWLD corpus, while the corresponding profiles for the ABC+ corpus are strongly asymmetric.

4.3.2 Degree of Ornamentation

A phenomenon that can already be observed from the chord profiles is that the general prevalence of ornaments varies between chord types. The predisposition of a chord type to generate ornaments occurs explicitly in the model (as $1 - \theta_h$) and can thus be quantified exactly. Figure 4.6 shows the posterior distributions of θ_h for most of the common chord types of both datasets (excluding the rare minor-major-seventh and augmented-seventh chords). For the ABC+ corpus, the chord types form three clusters. Dominant chords (dominant-seventh, diminished, and full-diminished) show a comparatively high degree of ornamentation (i.e., θ_h is small) in contrast to non-dominant chords, while the augmented chord is considerably less ornamented. There are several possible explanations for this phenomenon. Dominants have a critical function in tonal music as preparing motion to the tonic, which is particularly visible in the context of cadences. Ornamenting the dominant can increase its tension (e.g., by delaying) and sense of forward motion (e.g., through passing motions), while ornamenting the tonic in the same way would be in contrast to the sense of arrival and stability it represents. There is a number of typical ornamentations of dominant chords that fulfill this purpose, such as 4- and 6-4-suspensions, 8-7 passing motions, and anticipations. Moreover, the dominant is a point of increased attention and ornaments might be more noticeable at such a point, so a composer might prefer to add ornaments where they are noticed. The augmented chord, on the other hand, allows for almost no ornamentation. This may be due to the instability of the chord: If ornaments are added to an augmented chord, it becomes difficult to identify as an augmented chord, since most neighbors would allow for a reinterpretation of the chord as a minor or major triad, or as an augmented-seventh chord.

The EWLD corpus exposes a similar clustering of chord types, although the level of ornamentation is generally higher (note the different x-axis ranges in Figure 4.6), which may be due to the observed notes being taken from melodies instead of full textures. The notable exception to this similarity is that diminished and full-diminished chords are now much less ornamented, similar to augmented chords. This may be due to a different use of the chords in the genres found in the EWLD corpus compared to the styles in the ABC+ corpus, but it might also be a consequence of a different use of chord labels as instructions for musicians rather than descriptions or interpretations (as in the ABC+ corpus). A possible explanation of this difference is that dominant-seventh and (full-)diminished chords occur in similar contrapuntal contexts and are thus treated similarly in voice-leading music. If the focus is on the specific sound of a chord, dominant-seventh and diminished chords may be perceived as very different. Generally, however, it is unclear whether the differences between the corpora can be fully explained by stylistic differences since the material (melodies vs. full scores), the

purpose of chord labels, and the annotation or transcription procedures and goals differ as well.

To further investigate the idea of different classes of chords that exhibit a similar amount of ornamentation, the original model was extended to a clustered model, where different chord types share the same θ if they belong to the same cluster (all other variables staying the same, in particular ϕ^{ct} and ϕ^{or} remain separate for each chord type). A number of candidate models was selected by starting with one cluster per chord type (of the types shown in Figure 4.6), iteratively merging the two most similar clusters (similar to hierarchical clustering) and refitting the model, until all chords are assigned to a single cluster. All of these models were then combined into a meta model, in which the choice of model is represented by a categorical variable M . The generative process then first chooses the model (i.e., the clustering) from a uniform prior. and then proceeds as before. We approximate the posterior model probabilities

$$p(M | \vec{N}, \vec{c}) \quad (4.5)$$

using the same variational inference method as before, but keeping the posteriors of all model parameters fixed, only optimizing the posterior probabilities of the model choice. From the model probabilities we can directly compute the Bayes factors between pairs of models¹¹, but for the present case it suffices to look directly at the posterior probabilities. Since the posterior model distribution in Equation 4.5 marginalizes over the model parameters, it automatically penalizes models with more parameters in a principled way.

Figure 4.7 shows the two cluster models with the highest posterior probability. Both datasets prefer its respective model with a high probability of 0.98 each. The best model for the ABC+ dataset groups chord types into four clusters, with the lowest and highest clusters corresponding to the groups hypothesized before (dominants and augmented). The middle region, however, still splits into two groups, which indicates that the respective chord types are sufficiently distinct in their ornamentation tendency. The EWLD dataset prefers a more complex model with 6 clusters, which suggests that its chord types are more distinguishable with respect to θ . This phenomenon is in line with the observation made above that the role of chord types might have changed in modern music, with each chord type standing for its own distinct sound rather than different (but related) chord types emerging in similar contrapuntal configurations.

¹¹The Bayes factor is the likelihood ratio between two models, which in this case is equivalent to the ratio of the models' posterior probabilities because of the uniform prior over the models.

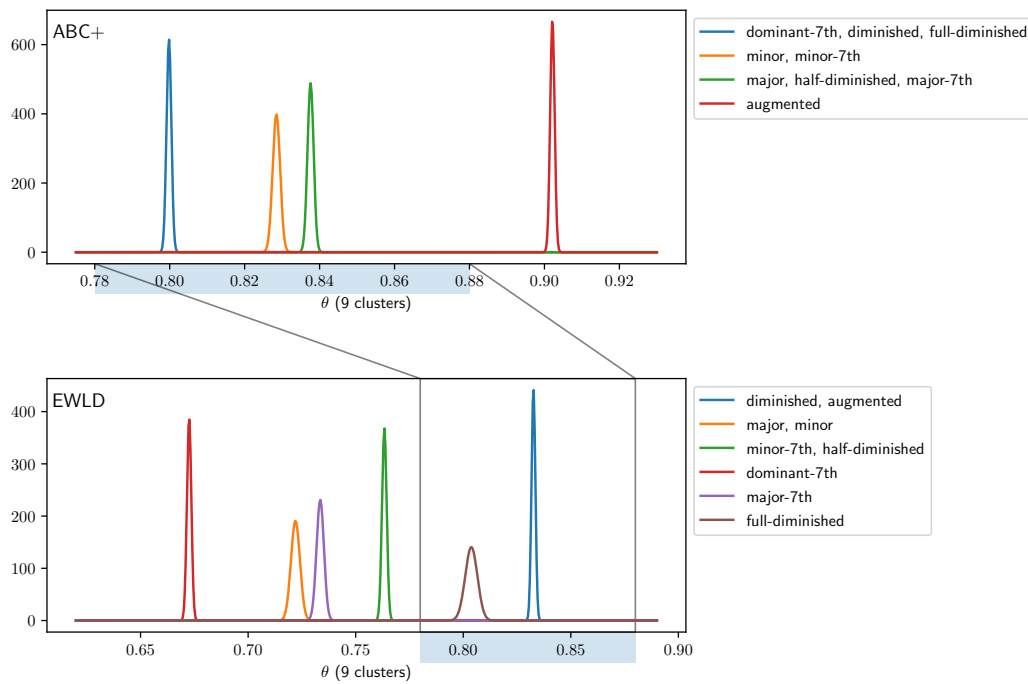


Figure 4.7 – The clustered- θ models with the highest probability for the ABC+ and EWLD datasets.

4.4 Conclusion

In this study, we presented a model that describes how an idealized listener could represent and learn the properties chord types in terms of chord-tone and ornament distributions, based on a generative probabilistic model. Using this model, we obtained chord profiles for two datasets, describing the ornamentation properties for a number of chord types in different styles. Preliminary analysis and suggest that ornamentation in classical styles might be related to diatonic contexts, while in modern styles the choice of non-chord tones could be more strongly tied to the chord itself. Furthermore, chord types differ in their disposition to be ornamented, with dominant chords showing a greater prevalence of ornaments than non-dominant chords.

The model that was used in this study was designed as a simple approximation to how chord types relate to pitches. Generally, the relation between harmonies and surface notes is more complex, involving rhythm and meter, the sequential order of notes, and the association of ornament notes with the respective ornamented notes. Generative Bayesian models can serve as a principled methodology for investigating the inference and learning in the context of complex structures, whether for statistical purposes

or as models of cognition. As the present study has shown, the use of probabilistic programming as a language to express probabilistic models makes it possible to move beyond standard models and implement precise and theory-driven model assumptions.

As a consequence, hypotheses about the way ornaments are organized may be implemented by adding structure to the model that reflects these hypothesized relations. For example, the idea that ornaments are determined by a diatonic context may be expressed by a model that explicitly selects ornaments from diatonic collections (or, more generally, other structured pitch collections such *Tonfelder*, Rohrmeier and Moss 2021) instead of a categorical distribution that assumes no relation between pitches. The simple model presented here can serve as a baseline against which to compare models that make additional structural assumptions. Bayesian modeling may also be applied to different learning problems, both by adding and by removing information in the data. Removing information leads to un- or semi-supervised settings that, for example, do not assume the chord labels or even the number of chord types to be given. Such a model would explore the types of chords that emerge from the data set, simulating how a listener implicitly learns to identify harmonic categories without explicit instruction (D. Hu and Saul 2009; Harasim, Moss, et al. 2021; Moss and Rohrmeier 2021). On the other hand, a more realistic model of ornamentation requires adding information such as temporal properties of notes, the octaves in which they occur, or the notes that are present before and after the chord.

Finally, the precise relationship between latent entities such as harmonies and the musical surface remains to be described. A generative model that also takes into account the exact placement of notes, contrapuntal structure, motivic phenomena, and other musical parameters would clarify the way in which harmonic types influence the surface on one hand (at least within one style or family of styles), and on the other hand explain how listeners potentially learn, represent, and recognize harmonic types.

5 A Graph Grammar for North Indian Melodies¹

Abstract

Hierarchical models of music allow explanation of highly complex musical structure based on the general principle of recursive elaboration and a small set of orthogonal operations. Recent approaches to melodic elaboration have converged to a representation based on intervals, which allows the elaboration of pairs of notes. However, two problems remain: First, an interval-first representation obscures one-sided operations like neighbor notes. Second, while models of Western melody styles largely agree on step-wise operations such as neighbors and passing notes, larger intervals are either attributed to latent harmonic properties or left unexplained. This paper presents a grammar for melodies in North Indian rāga music, showing not only that recursively applied neighbor and passing note operations underlie this style as well, but that larger intervals are generated as generalized neighbors, based on the tonal hierarchy of the underlying scale structure. The notion of a generalized neighbor is not restricted to rāgas but can be transferred to other musical styles, opening new perspectives on latent structure behind melodies and music in general. The presented grammar is based on a graph representation that allows one to express elaborations on both notes and intervals, unifying and generalizing previous graph- and tree-based approaches.

¹Originally published as:

C. Finkensiep, R. Widdess, and M. Rohrmeier (2019). “Modelling the Syntax of North Indian Melodies with a Generalized Graph Grammar”. In: *Proceedings of the 20th International Society for Music Information Retrieval Conference*. ISMIR (Delft, The Netherlands). Delft, The Netherlands, pp. 462–469. DOI: 10.5281/zenodo.3527844

The analytical and theoretical groundwork was done by RW and MR, while the mathematical formalization was done by CF in discussion with MR and RW. The generalized neighbor idea was suggested by Fabian Moss. CF was the main author of sections 5.2 to 5.6, the remaining sections were written jointly by the three authors.

5.1 Introduction

North Indian classical music (Hindustani music) provides valuable evidence for theories of syntactic musical organization. Like Western art music, it takes the form of aesthetic communication with an attentive and experienced audience, and is also a subject of theoretical discourse. Like most music outside the Western canon, it is normally unwritten, depending instead on memorization and improvisation. Instead of a system of chordal harmony or polyphony, Indian music comprises a solo melody against a complex background drone (of at least two pitches).

Melodic elaboration is prized as a means of musical extension and aesthetic enhancement: it operates at many levels, from the ornamentation of a single pitch, to the expansion of a phrase, to the architecture of a piece or performance. Melodic coherence is ensured by selecting one of a set of modes (*rāga*), each comprising a scale, a pitch hierarchy, and a set of licensed pitch transitions; any phrase that evokes a different *rāga* from the one selected is regarded as an error. It has been noted that Indian music resembles language in several respects (Powers 1980), and a *rāga* could be understood as a melodic grammar, in which melodies are constructed by recursive elaboration over a hierarchically organized set of pitches.

The idea of understanding music in a hierarchical fashion goes back to Schenker (1979), and has developed through the integration of impulses from generative linguistics and the theory of formal grammars since the 1980s (Steedman 1984; Baroni et al. 1983; Lerdahl and Jackendoff 1983; Rohrmeier and Pearce 2018; Pearce and Rohrmeier 2018). Approaches most commonly addressed harmonic structure (Steedman 1996; Rohrmeier 2011; Rohrmeier and Neuwirth 2015; de Haas et al. 2009; Granroth-Wilding and Steedman 2014) and melody (Gilbert and Conklin 2007; Marsden 2001). Several approaches proposed simplified formalizations of Schenkerian theory and corresponding computational implementations (Marsden 2001; Marsden 2007; Yust 2015b; Kirlin and Utgoff 2008). There is still comparably little discussion concerning the extent to which such hierarchical frameworks extend to non-Western forms of music. Narmour's theory of melodic processes is explicitly directed to capture melodies outside the Western canon as well (Narmour 1992). The application of Schenkerian methods to non-Western music has been discussed by Stock (Stock 1993). More recently, it has been proposed to adapt analytical tools from Schenkerian analysis and the GTTM to Indian music (Mukherji 2014; Clarke 2017).

This paper links with this discourse and proposes a generalized formal model of North Indian melodic and phrase structure. A common shortcoming in previous models of melodic elaboration is the treatment of leaps, which are usually either attributed to a

latent harmonic structure that is assumed to be known (Marsden 2001; Kirlin and Yust 2016), or modelled as probabilistic intervals (Gilbert and Conklin 2007; Groves 2016) without explicit restrictions. This paper introduces a formalism for relating leaps in North Indian music to a latent tonal hierarchy that is stated explicitly. With respect to this hierarchy, leaps can be viewed as instances of generalized neighbor- and passing-note relations that take into account the stability of a pitch in a scale. As will be argued, the generalized neighbor idea applies beyond North Indian music to some degree.

A central question for elaborative models concerns the representation of the music. Since formal grammars – the standard formalism for recursive elaboration – operate on strings of objects, most models of musical elaboration represent music as a sequence of objects, such as notes or chords. As a consequence, these models mostly focus on melodic (Gilbert and Conklin 2007; Groves 2016; Marsden 2001) or homophonic settings (Kirlin and Thomas 2015).

A desirable property of a formal grammar is that it is *context-free*, meaning that elaborations on a single object are independent from the objects around it. Systems that are based on strings of notes have problems with being context-free since some elaboration operations (such as passing notes) depend on two notes (Lerdahl and Jackendoff 1983). Because of this, more recent approaches have been based on strings of intervals (Marsden 2001; Yust 2015b; Gilbert and Conklin 2007), which allow elaborations of both single notes and pairs of notes while remaining context-free. However, in an interval grammar, notes are represented implicitly and redundantly (as part of an incoming and outgoing interval). In addition, all notes generated by elaboration are derived from two parent notes, which is unintuitive for single-sided operations. As a unification and generalization of both approaches, this paper suggests a graph-based representation in which both notes and intervals are represented explicitly, with a graph grammar describing the elaboration rules. This goes beyond descriptions of *derivations* as graphs, which is already an established practice (Marsden 2001; Yust 2015b; Kirlin and Utgoff 2008).

5.2 Melodic Operations

Melody in Indian music is based on a set of modes called *rāgas*. A *rāga* is not only a collection of pitches that may be used, it also establishes a hierarchy of stability among these pitches. Stable pitches are those that can serve as resting points, while less stable pitches tend to move towards their more stable neighbors. Some pitches in a *rāga* have a preferred resolution direction and must resolve to the closest pitch in that direction. An example of a *rāga* with its scale, tonal hierarchy, and directional constraints is shown



Figure 5.1 – Rāga Multānī with pitches in an approximate Western notation. The notated duration denotes the hierarchical level, i.e., relative stability, of each pitch; arrows indicate a constraint on the resolution direction of an unstable pitch.

in Figure 5.1. The relative stabilities indicated in Figure 5.1 is based on observation of normal practice in this rāga.

The melodic elaboration of a rāga is performed most completely and systematically (though not exclusively) in *ālāp*: a type of improvisation in which the scale and melodic features of the rāga are gradually exposed in phrases unfolding an arch-shaped trajectory, starting from the root (scale-degree 1) and reaching the octave above (or higher) before finally returning to the root (a process called *vistār* or “scalar expansion”, Widdess 1981). This background structure is filled and elaborated recursively, generating a complex foreground melody. Elaboration follows mainly two principles, inserting either passing or neighbor notes.

Passing notes fill intervals that are larger than steps. They can occur close to the surface (such as the $b2$ in $b3\ b2\ 1$), but can also be understood to characterize dependencies in the background (e.g., filling the top-level interval $1 - 1'$ with a 5). Two kinds of passing elaborations can be distinguished: Either a single note is introduced that subdivides the interval, potentially leaving non-step intervals that can be further elaborated; or the interval is filled with all scale notes enclosed by the interval.

Neighbor notes can be inserted before or after an existing note. While passing notes relate to both notes of an interval, neighbors are subordinate to single notes. When embellishing a note with a neighbor, a trade-off can be made between pitch proximity and stability: While unstable neighbors need to be very close to the main note’s pitch, more distant neighbors can occur if they are sufficiently stable. In general, a pitch can only be perceived as a neighbor to some reference pitch if no pitch in the interval between the two is more stable than the proposed neighbor in the given mode.

Figure 5.2 shows the steps needed to derive a phrase using neighbors and passing notes. Starting with a single 1, the note is duplicated and elaborated twice, first with a lower neighbor 7, then with an upper neighbor $b3$. Finally, the space between $b3$ and 1 is filled with a passing $b2$.



Figure 5.2 – A short Multānī phrase and its derivation.

$d \in D_M$	1	b2	b3	♯4	5	b6	7
$\delta_M(d)$	↑	↓	↑	↑	↑	↓	↓
$\lambda_M(d)$	4	0	2	1	3	0	2

Table 5.1 – A formal description of the rāga Multānī, showing the direction and hierarchical level of each scale degree (as shown in Figure 5.1). b2 and b6 are directed downwards and can therefore only be used before 1 and 5, respectively.

5.3 Modes and Generalized Neighbors

The idea of modes and generalized neighbors can be given a formal description: A *mode* M is a triple

$$\begin{aligned}
 M &:= (D, \delta, \lambda) \\
 \delta &: D \rightarrow \{\uparrow, \downarrow, \downarrow\} \\
 \lambda &: D \rightarrow \mathbb{N}
 \end{aligned}$$

where D_M is a totally ordered set of *scale degrees*, δ_M is a function indicating the *direction* in which a scale degree is allowed to move, and λ_M returns the *hierarchical level* of a scale degree. For example, the rāga Multānī (Figure 5.1) would be formalized according to Table 5.1.

The same scale degree can be used as a *pitch* in different octaves, so pitches are indicated as scale degrees together with “'” for octaves above and “,” for octaves below the default octave. The pitches of adjacent octaves are adjacent as well: 7, is directly below 1 and 1' is directly above 7. As a result, a mode gives rise to a set of pitches P_M , which corresponds to \mathbb{Z} while scale degrees correspond to $\mathbb{Z}_{|D|}$. For convenience, δ_M and λ_M are assumed to be defined on pitches as well and return the values of the corresponding scale degrees.

The set of pitches *between* a pitch p_1 and a pitch p_2 is the set of pitches in the open interval (p_1, p_2) that agree with the direction of the interval:

$$\Delta_M(p_1, p_2) = \begin{cases} \{p \in P_M \mid p_1 < p < p_2 \wedge \delta_M(p) \neq \downarrow\} & \text{if } p_1 < p_2 \\ \{p \in P_M \mid p_1 > p > p_2 \wedge \delta_M(p) \neq \uparrow\} & \text{if } p_1 > p_2. \end{cases} \quad (5.1)$$

The *neighbors* of a pitch $p \in P_M$ are then all pitches $n \in P_M$ that have a higher level than

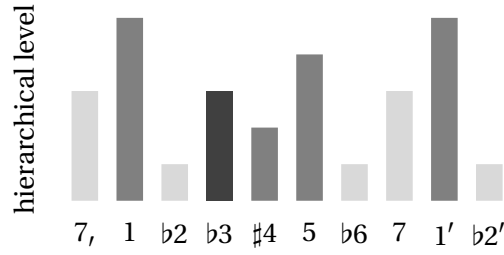


Figure 5.3 – The upper and lower neighbors (dark) of $b3$ (black) in the Multānī rāga. Only pitches that can be reached without skipping a more stable pitch are neighbors. $b2$ is not a neighbor since it is directed downwards and can only be a neighbor to 1.

all pitches *between* p and n . In addition, the direction of n must agree with the direction from n to p :

$$nb_M(p) = \{n \in P_M \mid p \neq n \wedge \forall q \in \Delta_M(n, p) : \lambda_M(q) < \lambda_M(n) \wedge n \rightarrow p\}, \quad (5.2)$$

where

$$n \rightarrow p = \begin{cases} \delta_M(n) \neq \uparrow & \text{if } p < n \\ \delta_M(n) \neq \downarrow & \text{if } p > n \\ \text{true} & \text{otherwise.} \end{cases} \quad (5.3)$$

Thus, every pitch is a neighbor to p only if it can be reached from the reference pitch without skipping a more stable pitch than the neighbor, as illustrated in Figure 5.3. Directed pitches can only be inserted as left neighbors since they must move towards their resolution.

When a single passing note is generated, the passing note must be a neighbor to both notes of the interval it is inserted in. However, in this case the inserted note is moving away from the first note, so the direction is not towards the reference note but towards the neighbor. A *reverse neighbor* $r \in rnb_M(p)$ is defined in analogy to a neighbor but with inverted direction:

$$rnb_M(p) = \{r \in P_M \mid p \neq r \wedge \forall b \in \Delta_M(p, r) : \lambda_M(b) < \lambda_M(r) \wedge p \rightarrow r\}. \quad (5.4)$$

For example, a passing $b2$ in the sequence $b3 b2 1$ is a neighbor to 1 but a reverse neighbor to $b3$, as it is directed away from $b3$ and towards 1.

Finally, a *fill* is the list of all pitches between two pitches p_1 and p_2 , sorted according to the direction of the interval (p_1, p_2) and restricted to pitches agreeing with that direction (as given by Δ_M).

$$\text{fill}_M(p_1, p_2) = \begin{cases} \text{sort}(\Delta_M(p_1, p_2), \text{asc}) & \text{if } p_1 < p_2 \\ \text{sort}(\Delta_M(p_1, p_2), \text{desc}) & \text{otherwise.} \end{cases} \quad (5.5)$$

5.4 A Formal Grammar of Rāga Melodies

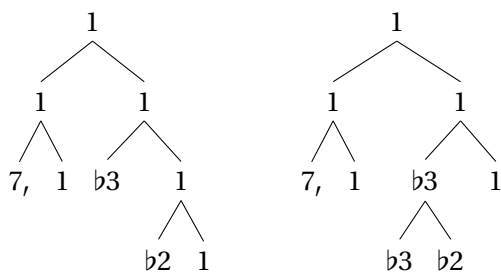
5.4.1 Representing Melodies as Graphs

As seen in Section 5.2, the two fundamental elaboration types – passing and neighbor notes – operate on two different musical entities: While neighbors elaborate single notes, passing notes fill intervals between two notes, elaborating both notes at the same time. As a consequence, two main formalisms describing hierarchical elaboration have emerged, note grammars and interval grammars.

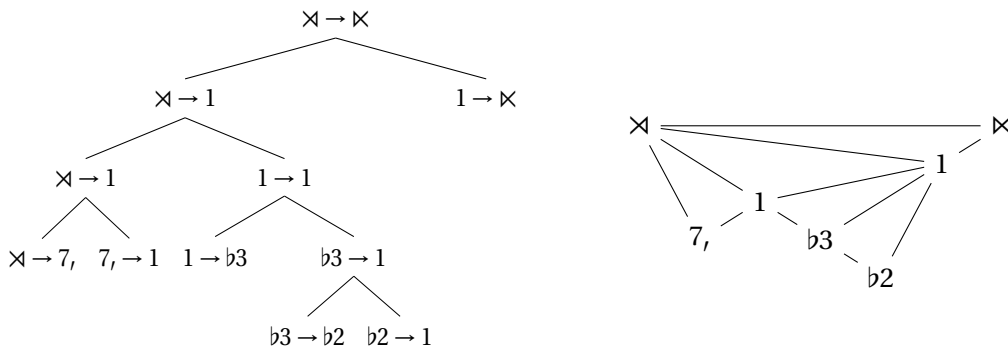
Note grammars generate strings of notes, with derivation rules replacing single notes by several new notes. The resulting hierarchical structure is a tree of notes as shown in Figure 5.4a. However, elaborating single notes is problematic for passing notes, as they elaborate two notes. Not only is the resulting hierarchy ambiguous (the passing note must be attached to either its predecessor or its successor), but from a generative perspective, a passing note can only be derived from one of its parents. Thus, deciding where a passing note may be inserted becomes a context-sensitive problem.

Interval grammars (Gilbert and Conklin 2007; Kirilin and Yust 2016; Groves 2016; Marsden 2001) solve the passing note problem (and two-sided operations in general) by elaborating pairs of notes, or intervals. Inserting a new note replaces an existing interval with two new intervals. The melody is then represented as a string of intervals with each note being represented twice, once as the second note of an interval and once as the first. To avoid this redundancy in notation, derivations are usually not given as trees (Figure 5.4b) but as outerplanar graphs (Figure 5.4c), giving each note two parents. However, for one-sided operations like neighbors, interval-based elaboration is conceptually misleading, as only one of the parent notes is considered while the other is ignored. This can lead to unwanted subordination of conceptually independent neighbors, as will be argued below.

As a unification and generalization of note- and interval-based systems, a graph-based representation of melodies is suggested here, representing notes as nodes and note



(a) Two analyses using note elaboration. The passing b2 must be attached either to the 1 on its right or to the b3 on its left.



(b) An analysis of the phrase using interval elaboration. The passing b2 is generated in the interval b3 → 1.

(c) The same analysis as in 5.4b, displayed as an outerplanar graph.

Figure 5.4 – Conventional formal analyses of the phrase in Figure 5.2.

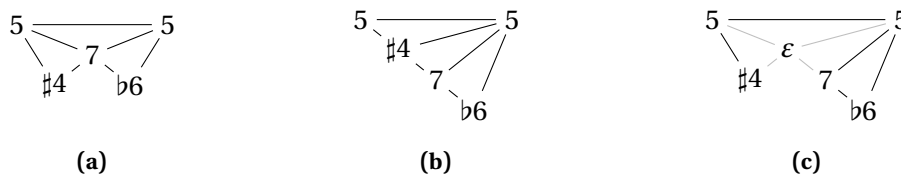


Figure 5.5 – Three possible derivations of 5 #4 7b6 5.

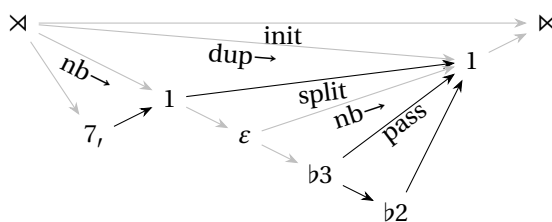


Figure 5.6 – An analysis of the phrase in Figure 5.2 using the rāga grammar. Dark edges indicate the subgraph induced by removing non-note nodes (X, X, ε).

transitions as edges. Using graphs as the basis for elaboration has both conceptual and practical implications. Conceptually, graphs represent both notes and note transitions explicitly, which allows the use of both entities as a starting point for elaboration. Practically, while graphs can represent strings of objects (such as melodies) as a special case, they can easily describe much more complex structures, which potentially allows the description of elaboration operations in non-monophonic music. However, special cases such as monophony can still be defined graph-theoretically, ensuring consistency under elaboration. Thus, graphs provide a common framework for both melodic grammars and more complex formalisms.

While graphs in principle allow operations on both nodes and edges, a much simpler and more consistent system is obtained by operating only on edges, resulting in an *edge-replacement graph grammar*. All operations are then defined on edges (i.e., node transitions) with one-sided operations ignoring one node of the edge. One-sided operations still introduce an edge between the unused note and the new one in order to allow further elaboration between them. In order to express the independency between the new and the ignored note, a dummy node (written as ε) can be introduced first between any two notes. A dummy node does not generate a note and is analogous to the empty string in a conventional grammar.

Only strictly one-side operations can be performed on edges adjacent to a dummy node. This restriction expresses the independence between one-sided elaboration notes and their opposite side, and permits a more appropriate hierarchy: Suppose two one-sided neighbors are generated between two 5s, a $\sharp 4$, as a right neighbor to the first 5 and a 7 as a left neighbor to the second 5 with a passing $b6$, resulting in $5 \sharp 4 7 b6 5$ (Figure 5.5). Without a dummy node, either 7 or $\sharp 4$ is subordinate to the other, depending on which is generated first (Figures 5.5a and 5.5b). By first introducing a dummy node, both neighbors can be derived independently (Figure 5.5c). Moreover, as dummy nodes are removed after the derivation, the resulting graph structure only retains edges that express elaboration dependence. Thus, dummy nodes allow the derivation to formally follow edge replacement while semantically expressing both one-sided and two-sided operations.

5.4.2 Formal Definition of the Grammar

A melody is formally represented as a directed linear graph with notes as nodes and transitions between notes as edges directed in time. The beginning and end of the melody are marked with the special nodes \times and \bowtie , respectively. The derivation is

started from a single 1:

$$\bowtie \rightarrow 1_M \rightarrow \bowtie,$$

with 1_M indicating the root of mode M .

Derivation rules follow an edge-replacement paradigm: edges can be replaced with new subgraphs, retaining the nodes adjacent to the original edge. Some rules use only one of the adjacent nodes. In this case, a wildcard symbol ($* \in P_M \cup \{\bowtie, \bowtie, \varepsilon\}$) is used for the ignored node. The special symbol ε represents the *empty melody* and can be used to split an edge into two parts that may be elaborated independently. Only one-sided operations can be used on edges adjacent to an ε , \bowtie , or \bowtie .

For a given mode M the rāga grammar $\mathcal{G}_M^{\text{rāga}}$ is defined as the graph grammar $(\mathcal{T}, \mathcal{N}, \mathcal{S}, \mathcal{R})$ with

$$\begin{aligned} \mathcal{T} &:= \{n_1 \rightarrow n_2 \mid n_1 \in P_M \cup \{\bowtie, \varepsilon\}, n_2 \in P_M \cup \{\bowtie, \varepsilon\}\} \\ \mathcal{N} &:= \{\} \\ \mathcal{S} &:= \bowtie \rightarrow \bowtie \end{aligned} \tag{5.6}$$

as terminals \mathcal{T} , non-terminals \mathcal{N} , and initial graph \mathcal{S} ; and the following replacement rules \mathcal{R} :

initialize:

$$(\bowtie \rightarrow \bowtie) \Rightarrow (\bowtie \rightarrow 1_M \rightarrow \bowtie)$$

duplicate left: $\forall p \in P_M:$

$$(p \rightarrow *) \Rightarrow (p \rightarrow p \rightarrow *)$$

duplicate right: $\forall p \in P_M:$

$$(* \rightarrow p) \Rightarrow (* \rightarrow p \rightarrow p)$$

left neighbor: $\forall p \in P_M, n \in nb_M(p):$

$$(* \rightarrow p) \Rightarrow (* \rightarrow n \rightarrow p)$$

right neighbor: $\forall p \in P_M, n \in nb_M(p) \wedge \delta_M(p) = \uparrow:$

$$(p \rightarrow *) \Rightarrow (p \rightarrow n \rightarrow *)$$

passing: $\forall p_1, p_2 \in P_M, n \in rnb_M(p_1) \cap nb_M(p_2):$

$$(p_1 \rightarrow p_2) \Rightarrow (p_1 \rightarrow n \rightarrow p_2)$$

fill: $\forall p_1, p_2 \in P_M:$

$$(p_1 \rightarrow p_2) \Rightarrow (p_1 \rightarrow f_1 \rightarrow \dots \rightarrow f_n \rightarrow p_2)$$

where $f_1, \dots, f_n = \text{fill}(p_1, p_2)$

split: $\forall p_1, p_2 \in P_M:$

$$(p_1 \rightarrow p_2) \Rightarrow (p_1 \rightarrow \varepsilon \rightarrow p_2).$$

In this description, rules are given as templates that are instantiated for all (combinations of) pitches. A more elegant and efficient description is possible, if rules are considered to

be functions on classes of structured symbols (Harasim, Rohrmeier, et al. 2018), allowing them to look inside their inputs.

Since the rāga grammar generates linear graphs, it is still possible to display derivations with outerplanar graphs. Figure 5.6 shows a derivation of the example phrase from Figure 5.2 using the rāga grammar. Each operation used to derive the phrase is written in the triangle formed by the old edge it replaces and the new edges it inserts. Later derivation graphs will omit operations and edge directions to remove visual clutter, as both are clear from the context.

In Figure 5.6, the ε inserted between the two 1s separates them and allows independent generation of neighbors. In particular, it would be possible to generate another right neighbor to the first 1 without subordinating it to the b3, or vice versa.

While the full derivation graph displays all derivation steps as they are formalized (i.e., as edge replacements), it does not distinguish one-sided and two-sided operations. Removing all non-note nodes ($\times, \bowtie, \varepsilon$) and the adjacent edges induces a subgraph in which two-sided operations still use two edges while one-sided operations adjacent to non-note nodes only use one edge. The resulting graph resembles both note trees and outerplanar graphs in different regions, depending on the type of operation being used there. Thus, using ε nodes is an analytical option that reveals independencies between adjacent parts of the graph.

The graph grammar $\mathcal{G}^{rāga}$ is a special case of a graph grammar that is formally equivalent to a context-free grammar on strings of notes. Therefore, it can parse melodies efficiently. The context-free grammar can be obtained in two steps: First, the graph representation is transformed to an interval representation in all parts of the grammar. Second, a set of rules is added for generating notes from intervals by taking the second note of each interval and generating empty strings (ε) where necessary.

In a directed linear graph, edges are totally ordered by their direction, so the graph can be transformed into a sequence of edges (e.g., $a \rightarrow b \rightarrow c$ becomes $(a, b)(b, c)$). Let e be the function that transforms linear graphs to sequences of edges. Then the context-free melody grammar $G(\mathcal{G}_{\mathcal{M}}) := (T, N, I, R)$ induced by a melody-graph grammar $\mathcal{G}_{\mathcal{M}}$ is

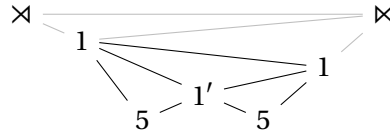


Figure 5.7 – The spine modeling the deep structure of the octave expansion in an ālāp.

defined as follows:

$$\begin{aligned}
 T_G &:= P_M \\
 N_G &:= (P_M \cup \{\varkappa, \varepsilon\}) \times (P_M \cup \{\varepsilon, \varkappa\}) \\
 S_G &:= e(\mathcal{S}_g) = (\varkappa, \varkappa) \\
 R_G &:= \{e(l) \Rightarrow e(r) \mid l \Rightarrow r \in \mathcal{R}_g\} \cup \\
 &\quad \{(x, p) \Rightarrow p \mid p \in P_M, x \in P_M \cup \{\varkappa, \varkappa, \varepsilon\}\} \cup \\
 &\quad \{(x, n) \Rightarrow \varepsilon \mid n \in \{\varepsilon, \varkappa\}, x \in P_M \cup \{\varkappa, \varkappa, \varepsilon\}\}.
 \end{aligned} \tag{5.7}$$

5.5 Discussion

A main motivation for introducing generalized neighbors is that they allow modelling leaps in the background structure of North Indian music. Figure 5.7 shows the architecture of a typical *ālāp* in rāga Multānī. The melody slowly ascends from 1 to 1' via 5 and returns back to 1 (again via 5). The upper 1' can be seen as a very stable and distant neighbor of 1 while the 5s in between are (again stable and distant) passing notes. Each stage of this *spine* is then further elaborated by neighbors and passing notes, using increasingly less stable pitches and smaller intervals (Figure 5.8).

North Indian music is not the only style of music in which melodies are based on hierarchical modes. If other mainly mode-based styles also follow the elaboration principles of passing notes and neighbors, then the grammar defined in Section 5.4 should permit sensible analyses for these cases. Consider, for example, the melody of *Nun komm der Heiden Heiland* (based on the Dorian mode) and a phrase from the Jazz standard *Moanin'* (based on a Blues scale). Their respective derivations (shown in Figure 5.9) suggest plausible reductions of the surface melody in both cases. Moreover, the proposed relations between notes match the intuitions of generalized neighbors and passing notes.

A natural generalization of the mode-based approach is to consider the mode as a latent variable that can change over the course of the piece but still organizes elaboration locally. Depending on the style, this hierarchy can be constant over longer regions of the piece or change rather frequently. The latter case occurs when harmonies are considered

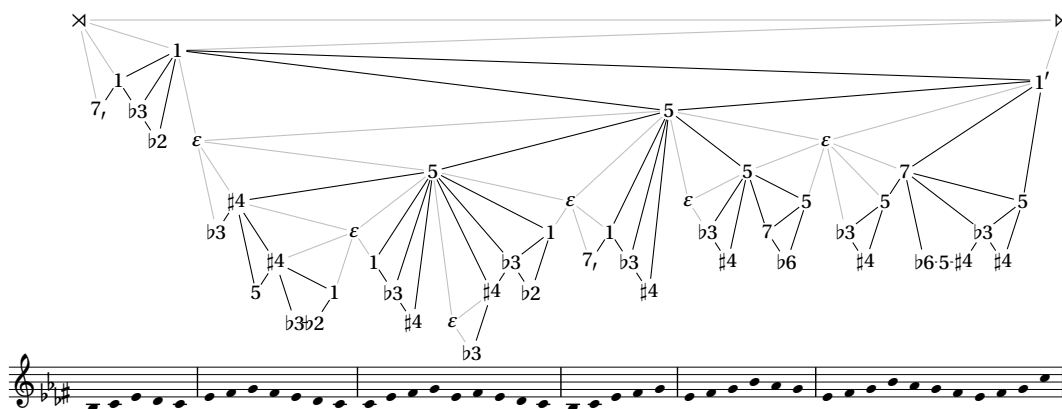


Figure 5.8 – This example represents selected phrases, in order of performance, excerpted from the ascending part of an ālāp in rāga Multānī, recorded by the sitarist Dharambir Singh (2019). For reasons of space, one phrase has been selected for each of the stations between 1, 5 and 1'. Surface ornamentation and rhythmic durations have been omitted.

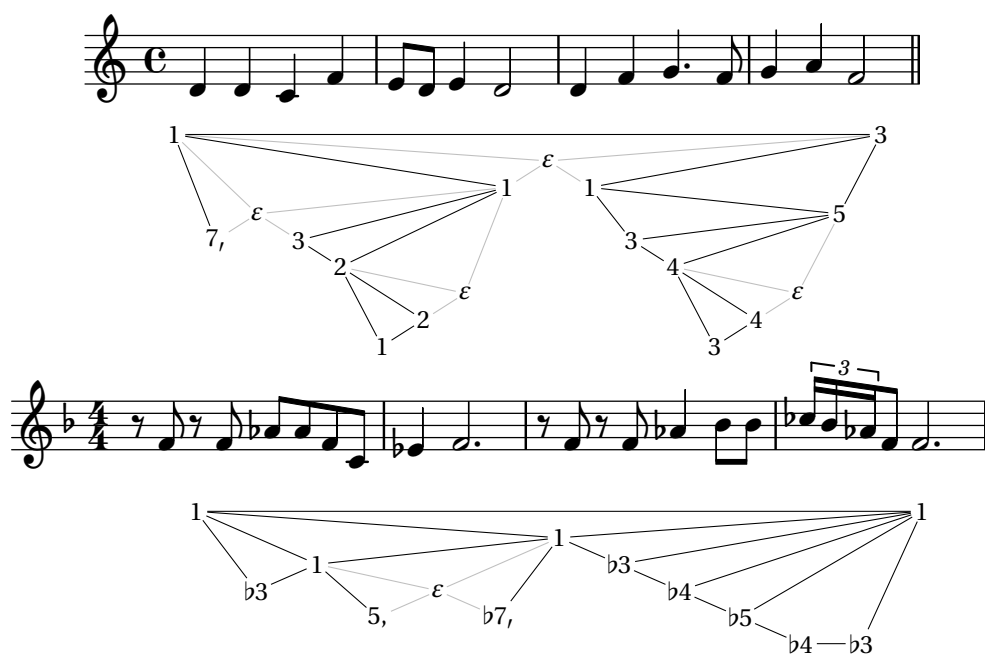


Figure 5.9 – The first two phrases from *Nun komm der Heiden Heiland* and *Moanin'* (without repetitions), and their derivations based on a Dorian and a Blues scale, respectively.



Figure 5.10 – The A part of *Take the A-Train* and a summary of its underlying lines.

as the latent structure, as they also define a tonal hierarchy, ranging from the root to non-chord notes.

However, there are two issues concerning generalized neighbor elaboration on harmonies. First, when the latent hierarchy changes, it is not obviously clear what should happen at the transition point. This is not an issue when these transitions are rare and elaboration across these boundaries is avoided. However, when harmonies take the role of the latent hierarchy, then transitions occur more often and elaborations frequently cut across harmonic changes. Moreover, as melodic elaboration happens on every level of reduction, it can even be considered to *generate* harmonic change in the background, such as the passing $\hat{2}$ in the *Ursatz*, generating a *V* harmony.

Second, not all leaps in melodies can be explained as generalized neighbors. The melody of *Take the A-Train* (Figure 5.10), for example, features several leaps which cannot be consistently explained as neighbors. While the initial G_4 and E_5 might be seen as neighbors to C_5 , the descent to E_4 in the end is left unexplained by that. Instead, it is more plausible to assume a set of several independent lines: A higher line descends from E_5 to C_5 , a lower line from G_4 to E_4 , and an intermediate line that connects G_4 and C_5 . Internally these lines behave according to elaboration principles (passing notes in this case), but the surface melody freely switches between the lines. This suggests that the organizing latent structure in this case is a set of implicit lines, although the elaboration and coordination of these lines might still be governed by a mode or a harmonic sequence as another layer of latent structure.

5.6 Conclusion

This paper proposed a generalized graph grammar formalism to model North Indian rāga music. We propose that passing and neighbor note elaborations are both necessary and (in their generalized form) sufficient operations of recursive rāga melody. This strengthens their status as fundamental musical principles across cultures. As the two operations are based on different objects (intervals and notes), models of elaboration should be able to represent both notes and intervals explicitly.

The notion of a generalized neighbor, based on a tonal hierarchy, shows that melodic leaps do not happen arbitrarily but can be related to a latent background structure. Understanding and modelling this background structure is necessary for a deeper understanding of melodic elaboration.

5.7 Acknowledgements

The research presented in this paper is generously supported by the Volkswagen Foundation and Claude Latour. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 760081 – PMSB. We thank Dharambir Singh for the permission to use his recording of an ālāp in rāga Multānī, and Fabian Moss for suggesting the generalized neighbor idea. We also thank the anonymous reviewers for their helpful feedback.

The Protovoice Model

Part III

Interlude 2

The three previous chapters have looked at different problems related to tonal structure. All of these problems touch the basic aspects of tonal structure as described in Chapter 1: recursive dependencies between surface notes (based on ornamentation), vertical and horizontal connections between notes, and the relation between latent entities and their surface expressions. While none of the three case studies has attempted to integrate all of these aspects, they can still help to better understand the problem of tonal structure, and give an idea of what an integrated model should look like.

The first case study, the schema matcher introduced in Chapter 3, attempted to model the relation between latent and surface structure in a heuristic way. While this approach did not fail entirely, the results still show that it is not sufficient to independently ask whether a certain set of notes instantiate a schema or not without considering alternative interpretations. Instead, a schema needs to part of a plausible explanation of a surface segment. This reflects an important aspect of interpretation that has already been raised in Chapter 1: Strong interpretations are not just derived from an observation (i.e., a piece), they *explain* the observation. For an integrated model, this suggests that latent entities such as schemata and chords² are explicitly used when generating the surface and thus must be represented in its derivation. In particular, the full surface is eventually derived from the latent entity, including ornamental notes that do not belong to the entity's prototype.

Chapter 4 presented an attempt to capture this relation between a latent entity and the ways in which they are elaborated. It was shown that different chord types come with different ornamental notes and can have different degrees of ornamentation. This suggests both that elaboration operations close to the surface may be influenced by latent entities that occurred earlier in the derivation, and that in turn particular latent entities may be more or less plausible depending on ornamental, seemingly unimportant

²This holds at least for those entities that involve the same types of relations as the rest of the model (e.g., simultaneity and sequentiality), which schemata and chords do. Other entities such as motives or themes have a different types of relations to the surface (e.g., literal or modified repetition) and will therefore not play a direct role in the model discussed here. A generative model that explains a piece based on form and motivic/thematic material, however, would directly represent those entities in a derivation.

Interlude 2

surface features. Much like the schema matcher, however, the ornamentation model ignored the direct relations between surface notes. An integrated model would therefore have to show how the surface is exactly derived from the prototype, that is how chord tones are distributed on the surface and how ornaments are introduced.

The model of modal melodies presented in Chapter 5 demonstrates that a grammar model based on edge elaboration can not only capture the hierarchical structure of a melody (i.e., which notes are derived from which other notes), but also the precise type of relation between parents and children (repetitions, neighbors, or passing note) that were left unaddressed in Chapter 4. Moreover, the latent pitch hierarchy of the mode further restricts which elaboration operations are permitted in a given situation, accounting for some form of (static and atemporal) vertical organization. Due to this static vertical structure, there are no vertical relations between melody notes in the modal model, that is notes are never considered simultaneous or concurrent, not even on a more abstract level. Thus, the modal model could be seen as a model of “properly monophonic” melodies. In general tonal structure, however, where even monophonic melodies can exhibit latent polyphonic structure, sequential relations between notes are not sufficient. Vertical structure becomes something temporal that may change over the course of the piece, and something that is explicitly modified during the generative process. This raises two questions for a non-monophonic model: How is latent vertical structure represented and modified? And how are temporal expansion and ornamentation coordinated between concurrent sequential progressions?

The third part of this thesis presents the *protovoice model*, a first attempt at a unified model of the three basic relations of tonal structure, sequentiality, simultaneity, and elaborative dependency. Just like the modal melody grammar from Chapter 5, this model is generative and thus provides an explanation (or *strong interpretation*) of a piece in the form of a derivation, a complex combination of a small set of simple operations that produce the piece’s surface notes and their basic relations. Higher-level concepts that go beyond the basic relations (e.g., the more specific chords and schemata) are not explicitly represented by the model, which means that the model works without presupposing them and in situations where they break down. However, it will still be shown that the derivations produced by the model can implicitly reflect these entities and their relation to the surface.

Chapter 6 introduces the model formally and presents an algorithm for obtaining all possible derivations of a piece. Chapter 7 complements this formal characterization with a number of music-theoretical examples and arguments, demonstrating how the model captures a range of phenomena related to tonal structure, implied polyphony, and latent entities. Chapter 8 briefly presents a web-based tool for manually annotating

derivations. Finally, Chapter 9 shows how Bayesian probability theory can be integrated into the model in order to make judgments about the plausibility of different analyses of the same piece. The overall results are summarized and linked back to the broader context of this research in Chapter 10.

6 The Protovoice Model¹

Abstract

Voice leading is considered to play an important role in the structure of Western tonal music. However, the explicit voice assignment of a piece (if present at all) generally does not reflect all phenomena related to voice leading. Instead, voice-leading phenomena can occur in free textures (e.g., in most keyboard music), or cut across the explicitly notated voices (e.g., through *implicit polyphony* within a single voice). This paper presents a model of *protovoices*, voice-like structures that encode sequential and vertical relations between notes without the need to assume explicit voices. Protovoices are constructed by recursive combination of primitive structural operations, such as insertion of neighbor or passing notes, or horizontalization of simultaneous notes. Together, these operations give rise to a grammar-like hierarchical system that can be used to infer the structural fabric of a piece using a chart parsing algorithm. Such a model can serve as a foundation for defining higher-level latent entities (such as harmonies or voice-leading schemata), explicitly linking them to their realizations on the musical surface.

6.1 Introduction

A basic observation about tonal structure in music is that notes tend to form vertical and horizontal relations, which are generally not explicit in representations of the musical surface such as a score or a recording. An example of these relations can be seen in Figure 6.1. The initial line of sixteenth notes in the right hand, for example, forms an arpeggiation of a D-minor chord. A reduction or simplification of the piece might realize

¹Originally published as

C. Finkensiep and M. Rohrmeier (2021). “Modeling and Inferring Proto-Voice Structure in Free Polyphony”. In: *Proceedings of the 22nd International Society for Music Information Retrieval Conference*. ISMIR. Online, pp. 189–196. DOI: 10.5281/zenodo.5624431

Model, implementation, and paper have been designed and written by CF under the supervision of MR.

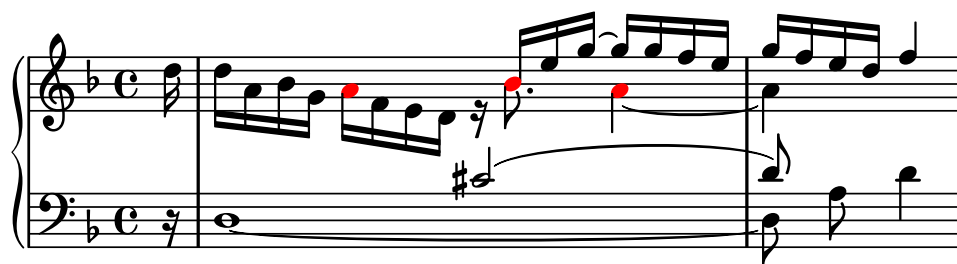


Figure 6.1 – An example of free polyphony in J. S. Bach’s Allemande BWV 812 I. Sequential structures (such as the A-B \flat -A motion across the first measure) are generally not explicit in the score.

this chord as a single vertical entity, but the vertical relation between the notes D5, A4, F4, and D4 is not directly encoded in the score. Similarly, the two A4s of this arpeggiated chord are part of a line that first moves to the neighbor note B \flat 4 before returning to A4 on the fourth beat of the first bar. Again, this connection is not explicitly represented in the score, much less so in a recording.

Sequential relations between notes are sometimes equated with *voices* (Huron 2016) that are either explicitly given (e.g., in monophonic melodies or strict polyphony), or inferred through voice separation (Chew and Wu 2005; de Valk and Weyde 2018; Guiomard-Kagan et al. 2015; Kilian and Hoos 2002; Kirilin and Utgoff 2005; Makris et al. 2016; McLeod and Steedman 2016; Temperley 2009). However, sequential relations do not always coincide with voices: A single voice can exhibit *implicit polyphony* (Aldwell and Cadwallader 2018, p. 367; Cambouropoulos 2006; also called implied or latent polyphony), i.e., consist itself of several implied sub-voices. For example, in the upper voice in Figure 6.1, the notes of the D-minor chord belong to separate voices on a more abstract level. Similarly, sequential connections can go across different voices, such as the A4 moving to B \flat 4 while the notated voice continues to C \sharp 4.

Implicit (or more generally free) polyphony is commonly understood as forming a set of parallel and independent *auditory streams* (Bregman 1990; Cambouropoulos 2008; Huron 2016) that are inferred from the musical surface by connecting notes into sequences. The present paper, in contrast, proposes a model of free polyphony that departs from this view in several respects: First, free polyphony is understood as a network of lines that can be connected to each other rather than a set of independent streams. Second, this network is not defined through inference from the surface, but rather explicitly constructed in a generative process that creates the network in successive steps. Inferring this network from a piece is then based on inverting this process, i.e., *parsing* the piece. Third, connections between notes are not based on continuing a stream, but instead follow from *elaboration* of existing structures through fundamental and musi-

cally interpretable operations, adopting a top-down view instead of a left-to-right view on voice-leading structure (Lerdahl and Jackendoff 1983; Rohrmeier 2011). We name the resulting lines in the network *protovoices*, since – like voices – they connect notes to sequential lines but cannot be themselves implicitly polyphonic. This paper presents a formal definition of the protovoice model as a recursive process, and describes a parsing algorithm that can infer the protovoice structure from a score. This model does not yet account for other musical aspects such as rhythm and meter, harmony, form, or motivic and thematic material. However, it is intended to further the understanding of polyphonic structure on formal grounds, and could potentially serve as a module in a more complete system for musical analysis.

The idea of modeling free polyphony as a recursively generated network of lines is central to Schenkerian analysis (Schenker 1979). However, the constructions in Schenkerian analysis are specific to Western Common Practice music and more high-level than the generic operations that give rise to protovoices. Thus, the protovoice model can be understood as a formal foundation for describing richer concepts of musical structure (such as the ones appearing in Schenkerian analysis or other analytical frameworks), and it is applicable to a wider range of musical styles that make use of implicit or free polyphony (such as Jazz or melodies in Pop/Rock).² However, because of these similar ideas, the model presented here is related to models that formalize sub-systems of Schenkerian analysis (Frankel et al. 1978; Smoliar 1979; Rahn 1979; Kirlin and Utgoff 2008; Kirlin and Yust 2016; Kirlin and Jensen 2011; Kirlin and Thomas 2015; Yust 2006; Marsden 2001; Marsden 2010), and to grammatical models of musical structure in general (Lerdahl and Jackendoff 1983; Rohrmeier 2011; Harasim, Rohrmeier, et al. 2018; Harasim, O'Donnell, et al. 2019; Finkensiep, Widdess, et al. 2019; Gilbert and Conklin 2007; Granroth-Wilding and Steedman 2014; Abdallah, N. E. Gold, and Marsden 2016; Melkonian 2019). While protovoices inherit some of their concepts, most notably the interval-replacement method developed in (Yust 2006), modeling the structure of free polyphony has yet been an unsolved problem.

²The principle of recursive ornamentation is also used in non-Western styles, such as Indian classical music (Finkensiep, Widdess, et al. 2019), the model presented here is specifically inspired by Western tonal music. However, some of the formal techniques presented here might also be useful for expressing structural relations specific to other styles.

6.2 The Protovoice Model

6.2.1 Constructing Protovoices

At the core of the model proposed in this paper are a number of operations that establish primitive and strictly stepwise horizontal relations between notes. These relations include *repetitions*, stepwise ornaments to a note (*neighbor notes*), and notes that fill larger intervals stepwise (*passing notes*). While the notion of a step generally depends on what is considered a step in the respective style, we consider a step to be a diatonic second for the purpose of modeling tonal music in the diatonic tradition.

All of these operations relate notes to one or two reference notes, or *parents*. Following Yust (2006), operations with two parents are represented by *edge replacement*: If the two parent notes p_1 and p_2 are connected by an edge $p_1 \rightarrow p_2$, then this edge can be replaced by a child note together with two new edges to the parents: $p_1 \rightarrow c \rightarrow p_2$.

Formally, protovoices are represented as a graph that contains one vertex per note, one vertex each for the beginning (\bowtie) and the end (\bowtie) of the piece, and two types of edges: *Regular edges* indicate a sequential connection between two notes (or \bowtie/\bowtie) that may be used for elaboration by introducing a repetition or a neighbor of either parent note (or of both if the parents have the same pitch). The interval along a regular edge is always within the range of a step (unless one of its vertices is \bowtie or \bowtie), and this property is maintained through the elaboration operations. *Passing edges* indicate connections between two notes with an interval that is larger than a step (introducing a new, subordinate protovoice). They must be filled with passing notes from either end until only stepwise connections remain.

The generation of a piece starts with the empty piece $\bowtie \rightarrow \bowtie$ and recursively applies one of several elaboration rules. *Single-sided* rules pick a note and insert either a repetition or a neighbor note to its left or right:

$$x \Rightarrow x' \rightarrow x \quad \text{repeat-before} \quad (6.1)$$

$$x \Rightarrow x \rightarrow x' \quad \text{repeat-after} \quad (6.2)$$

$$x \Rightarrow n \rightarrow x \quad \text{left-neighbor} \quad (6.3)$$

$$x \Rightarrow x \rightarrow n \quad \text{right-neighbor} \quad (6.4)$$

Double-sided rules pick an edge and insert along it one new note and two new edges:

$$\times_1 \rightarrow \times_2 \implies \times_1 \rightarrow x \rightarrow \times_2 \quad \text{root-note} \quad (6.5)$$

$$x_1 \rightarrow x_2 \implies x_1 \rightarrow x' \rightarrow x_2 \quad \text{full-repeat} \quad (6.6)$$

$$x \rightarrow y \implies x \rightarrow y' \rightarrow y \quad \text{repeat-before'} \quad (6.7)$$

$$x \rightarrow y \implies x \rightarrow x' \rightarrow y \quad \text{repeat-after'} \quad (6.8)$$

$$x_1 \rightarrow x_2 \implies x_1 \rightarrow n \rightarrow x_2 \quad \text{full-neighbor} \quad (6.9)$$

Passing rules, finally, fill passing edges with passing notes from either end until the progression is fully stepwise:

$$x \rightsquigarrow y \implies x \rightarrow p \rightsquigarrow y \quad \text{passing-left} \quad (6.10)$$

$$x \rightsquigarrow y \implies x \rightsquigarrow p \rightarrow y \quad \text{passing-right} \quad (6.11)$$

$$x \rightsquigarrow y \implies x \rightarrow p \rightarrow y \quad \text{passing-final} \quad (6.12)$$

In these rules, matching letters indicate matching pitches, indices disambiguate parent notes with the same pitch, and apostrophes mark inserted repetitions of parent notes. Neighbor notes n must be a step away from their parents, (disregarding their octaves to allow for octave displacement). Similarly, passing notes p must be a step from the parent(s) they are directly connected to and lie within the interval spanned by both parents. Note that none of these rules produce passing edges, which establish new connections between previously unconnected lines and thus require some additional structure (see Section 6.2.2). An example protovoice derivation of the previous example (Figure 6.1) is shown in Figure 6.2.

6.2.2 Temporal Organization

While protovoices model the sequential organization of notes, they do not specify when notes are simultaneous. On the musical surface, simultaneity of notes is implied by their onsets and durations. However, notes that are temporally displaced on the surface can often be regarded as forming a vertical sonority on a higher level of abstraction, such as the arpeggiated d-minor chord in the beginning of Figure 6.1. In order to express these latent vertical configurations, simultaneity is modeled through *slices*, segments of a piece in which the same notes sounds. A piece (or a reduction of a piece) is then represented as a sequence of slices. Notes that are simultaneous with several non-simultaneous notes (such as the bass note D in Figure 6.1) are split among the corresponding slices but remain connected by edges, thus ensuring that a surface note is generated through a single generation process.

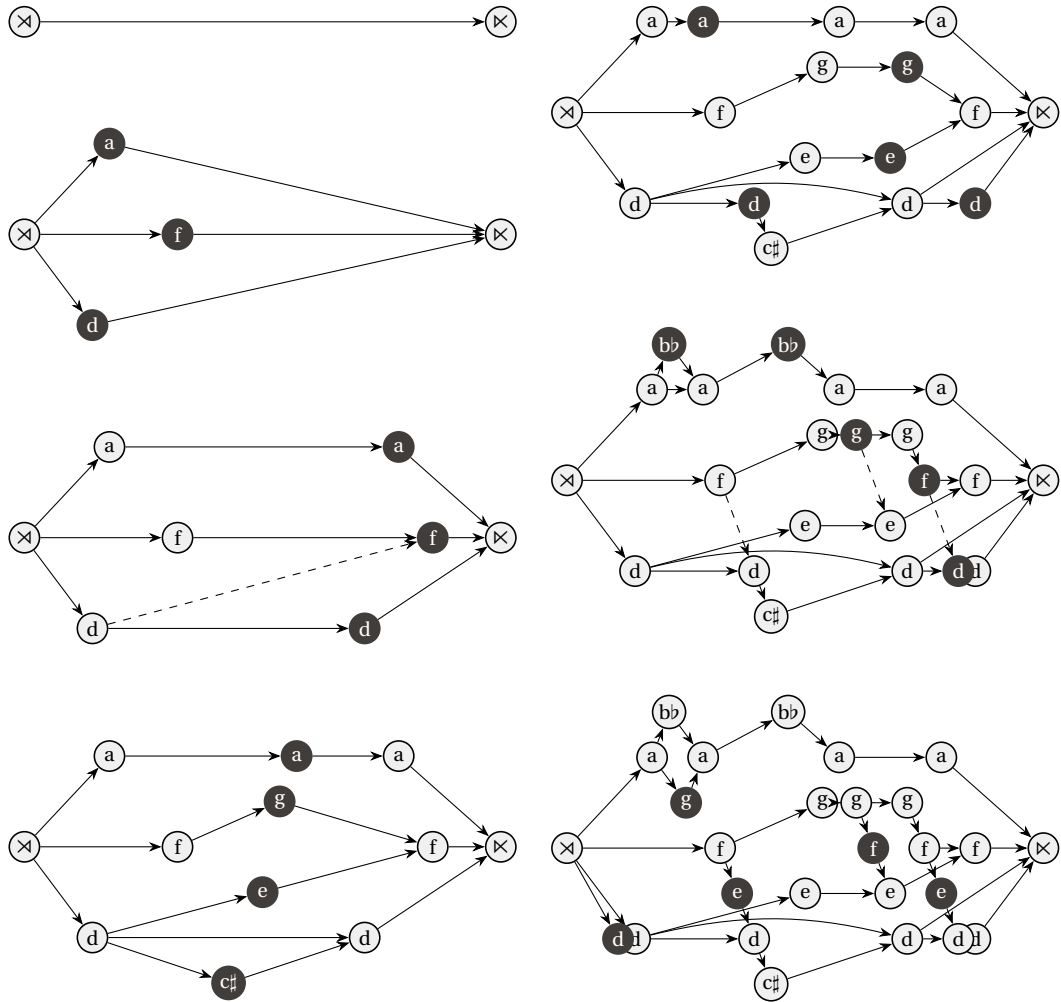


Figure 6.2 – A protovoice derivation of the notes in Figure 6.1. The position of a note is chosen to indicate its pitch and onset in the piece. Later derivation steps hide some edges from earlier steps in the interest of readability. Note that each note is shown exactly once here, unlike in the final model, which represents each note once per slice it occurs in. Furthermore, pitches in different octaves have been merged to simplify the graph.

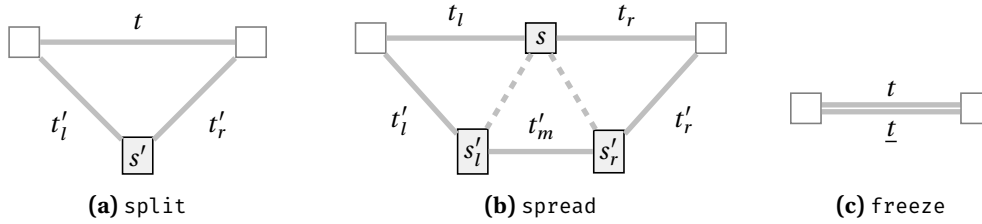


Figure 6.3 – The three operations on outer structure. The slices and transitions to be elaborated are shown at the top while the lower part shows the generated structure.

Protovoices are integrated into the slice structure by attaching their edges to the *transitions* between two slices. Note that transitions can only contain edges that connect notes in the slices adjacent to the transition. Long-distance edges are thus represented in latent transitions, i.e., transitions in a reduction of the piece. As a consequence, edges “vanish” in a well-defined manner during the generation process, namely whenever a transition is replaced through one of the generative operations. Since slices and transitions contain notes and edges, respectively, we call the slices and transitions *outer structure*, and the notes and edges *inner structure*.

Formally, a slice s is defined as a multiset (or bag) of pitches. A transition $t = (s_l, e, s_r)$ relates two slices s_l and s_r and a configuration of edges $e = (e_{\text{reg}}, e_{\text{pass}})$, which in turn consists of a set of regular edges e_{reg} (which must be used at least once by a subsequent operation) and a multiset of passing edges (which must be used exactly once).³

Outer structure is transformed by three operations: A `split` (Figure 6.3a) is a rule of the form

$$t \longrightarrow t'_l s' t'_r \quad (6.13)$$

that replaces a transition t by inserting a new slice s' and two new transitions t'_l and t'_r . During this operation, each edge in the transition and each note in an adjacent slice can be elaborated by one or more inner operations. The resulting edges can either be discarded, or kept to form the new edges of t'_l and t'_r . As a result, each transition only contains edges that will be used subsequently.

A horizontalization, or `spread` (Figure 6.3b) has the form

$$t_l s t_r \longrightarrow t'_l s'_l t'_m s'_r t'_r, \quad (6.14)$$

and replaces a slice s by distributing its notes to two child slices s'_l and s'_r . This way, a latent vertical configuration of notes can be sequentialized. In order to simplify parsing, a restriction is made on this distribution: At least one side must inherit all instances of a

³Passing edges are treated differently to avoid filling a single passing edge several times.

specific pitch, while the other may inherit fewer instances, i.e.,

$$p^k \in s \quad \Rightarrow \quad p^k \in s'_l \quad p^{k-m} \in s'_r \quad \text{or} \quad (6.15)$$

$$p^k \in s \quad \Rightarrow \quad p^{k-m} \in s'_l \quad p^k \in s'_r, \quad (6.16)$$

where k denotes the number of occurrences of pitch p in s , and $0 \leq m \leq k$. This way, the s can always be inferred deterministically from s'_l and s'_r by taking for each pitch the maximum number of occurrences in s'_l or s'_r .

In the process of a spread, passing edges may be introduced between arbitrary pairs of notes, and regular edges may be introduced between notes with the same pitch. This way, the introduction of passing edges becomes a local operation that is guaranteed to respect the temporal order of notes. Since all edges in a transition must be used, a spread is only allowed when no edges from the parent transitions t_l and t_r are lost by moving notes to the opposite side. While this operation does not change the contents of t_l and t_r , it replaces s with s'_l and s'_r respectively, which makes this operation context-sensitive.

Finally, a freeze (Figure 6.3c) marks a transition as terminal, stopping the generation process for this transition:

$$t \longrightarrow \underline{t}. \quad (6.17)$$

It is only allowed when the transition contains only repetition edges, which are turned into ties, creating notes that span several surface slices.

An example derivation using these three operations can be seen in Figure 6.4. We use a notation similar to the maximal outerplanar graphs (MOPs) introduced in (Yust 2006), with the root transition on top, surface of the piece on the bottom, and rule applications indicated by polygons. However, since the derivations here contain latent slices that do not occur on the surface, these derivation graphs are not outerplanar.

6.3 A Parsing Algorithm for Protovoices

6.3.1 Representing Derivations

The parsing algorithm for the protovoice model produces a set of possible derivations of the input score. Such a derivation can be represented as a list of rule applications in *leftmost derivation* order. This representation is known from context-free grammars: the result of the derivation is obtained by applying each rule in the list to the leftmost non-terminal symbol of the current sequence. This is possible because the derivation below

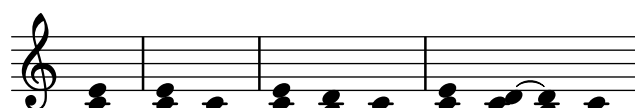
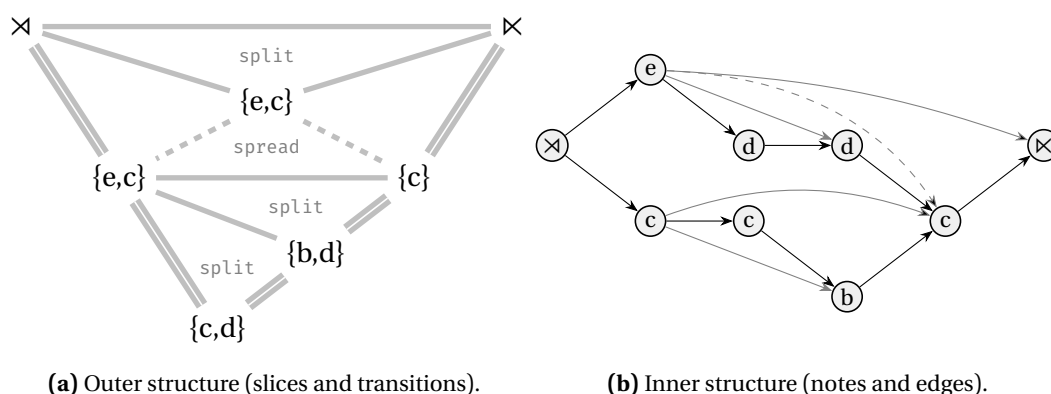


Figure 6.4 – An example derivation of a short cadential phrase. In each `split` operation, the edges of the elaborated transition (grey in (b)) are replaced using inner elaboration operations. The passing edge from `e` to `c` is introduced during the spread of the top-level `{e, c}` slice.

each non-terminal of a string is independent from the derivations below all other non-terminals of the string. In the protovoice grammar, this independence property does not hold, because the context-sensitive operation `spread` can link two otherwise independent transitions (and all their ancestors). However, the idea of a leftmost derivation can still be applied here.

The maximal left-hand side of a single rule consists of two transitions. Thus, instead of the leftmost non-terminal, we consider the two leftmost non-terminal transitions as the context for each rule application. Freezing the left of the two transitions moves the context to the right. A `spread` consumes both transitions of the context and pushes its children onto the list of open transitions. In order to allow the right parent of a `spread` to be the result of a `split`, `splits` can be applied to either the left or the right transition of the current context. However, in order to disambiguate the derivation order, we restrict right `splits` to always happen *after* left `splits` or `freezes`. If only a single transition is left, then only a `split` or `freeze` can be performed. Thus, the derivation shown in Figure 6.4a can be unambiguously described as the leftmost derivation `split, spread, freeze, split-left, split-left, freeze, freeze, freeze, freeze`.

Under these restrictions, certain configurations are not possible. In particular, the right parent transition of a `spread` cannot be the left child transition of another `spread`.

However, this *outer* configuration is equivalent – with respect to the resulting *inner* structure – to another configuration where the two spreads are applied in reverse order. Thus, the generative power of the grammar (with respect to protovoice structure) is not restricted by excluding this non-leftmost configuration.

A similar observation above can be made between *splits* and *spreads*: Whenever a *split* is made *after* a *spread* (i.e., on its left or right child transition), it could as well have been made before the *spread* (generating its left or right parent transition, respectively), generating the same inner structure. Therefore, we can add another restriction on the derivation order that forbids *splits* to be applied to the left or right child transitions of a *spread*, further removing the redundancy between (internally) equivalent derivations.

In a similar fashion, it is possible to reduce the number of derivations further by eliminating redundancy in the internal structure. For example, slices that are exact repetitions of one of their neighbors can be generated in two ways, either by a *split* that only uses *repeat* - * operations on one side, or by a *spread* that produces identical child slices. Since the latter is required for passing edges, the former case might be excluded as redundant. Similarly, the repeated horizontalization of a vertical configuration can generate the same surface configuration in many different ways, which can be prevented by restricting *spreads* to be strictly left- or right-branching (unless intercepted by a *split*). Both of these restrictions, however, exclude some derivations with slightly different semantics than their permitted counterparts, so it depends on the use cases whether such restrictions are appropriate.⁴

6.3.2 Parsing

Previous models of hierarchical tonal structure have relied on two approaches to structural inference: Grammar-based models use variants of classical parsing techniques such as chart parsing (Harasim, Rohrmeier, et al. 2018; Marsden 2010) while MOP-based models work with triangulations of polygons (Yust 2006; Kirlin and Utgoff 2008). The protovoice model can be parsed using a bottom-up chart parsing algorithm that is adapted to account for the context-sensitive *spread* operation. A *transition chart* stores all potential latent transitions, similar to the non-terminal chart in a context-free parser. In addition, a *verticalization chart* stores items that represent the “core” of a *spread*, i.e., the parent slice and the middle transition (including the two child slices). This core is then combined independently with the left and right child transitions, disentangling

⁴For example, with a strictly right-branching model, the expansion of the D-minor chord in Figure 6.1 must happen from left to right. If it is desired to split the chord first into quarter-note slices and then into eight-note slices (to respect the metric structure), strict right-branching does not work.

the two reductions and reducing the combinatorial complexity.⁵

The items in the transition chart are tuples (t, σ, I_l, I_r) , consisting of a transition t , a *score* σ , and two IDs I_l and I_r that express combination restrictions on the left and the right of the transition, respectively. By default I_l and I_r have a default value $*$ which indicates that they can combine with other transitions with the default value. The left and the right parent transitions of a spread, however, depend on each other through a common child (the spread operation itself). They are therefore marked with a special ID on their adjacent sides and can only combine with other transitions with a compatible ID. IDs are based on the *left side* of the verticalization, i.e., its left child slice and its parent slice. The details of the spread operation as well as the middle and right child transitions are stored in the item of the right parent transition, while the left parent transition only keeps a reference to the left child transition. This way, combining any pair of compatible left and right parent transitions restores a complete and valid spread operation with all its children. While this “trick” reduces complexity by exploiting some properties specific to the protovoice grammar, it is not known whether it reduces the overall complexity of the parser from exponential to polynomial in the number of input slices.

The score σ of a transition represents the set of leftmost derivations from the transition to the surface it covers.⁶ It is computed bottom-up by combining the scores of the transition’s children. When two transitions are combined, their scores are combined by concatenating each alternative on the left with each alternative on the right.⁷ When parsing a `split` operation, this result is prepended with the `split` itself, which yields the score of the parent transition. The score representing a spread operation, however, must be distributed across the two parent transitions. This follows the same scheme as described above: the left parent keeps the score of the left child L ; the right parent takes the scores of the right child R , the the middle child M , and the rule application the spread itself h . However, since the correct leftmost sequence of operations should apply the scores in the order $hLMR$ the scores of the parent edges are *partial*, and the parser ensures that these fragmented derivations are handled in a way that always restores the correct sequence of derivation steps when recombined.⁸

⁵For a given verticalization, instead of considering each pair of left and right transitions ($|L| \cdot |R|$ operations), the left and right transitions can be processed independently ($|L| + |R|$ operations).

⁶More generally, the parser uses semiring parsing (J. Goodman n.d.) to produce different kinds of results, such as the number of parses or the probability of a sequence. However, the set of derivations is both the most general and the most intuitive semiring, so we focus on it.

⁷In the parser, this operation is represented symbolically, which is more efficient than actually computing all combinations of alternatives.

⁸In particular, since fragmented derivation sets are not always re-combined right away, they need to combine with other operations such as `splits` and other `spreads`. The formal details of this are beyond the scope of this paper, but they are documented in the parser implementation.

Algorithm 6.1 The steps of the parsing algorithm.

```
 $V \leftarrow \{\}$   
 $T \leftarrow$  unfreeze each input transition  
for  $n$  from 2 to  $|\text{input}| - 1$  do  
   $V \leftarrow_{\cup}$  verticalizations of all  $T_n$   
   $T \leftarrow_{\cup}$  left vert. of all  $T_n \otimes V_{\leq n}$  and  $T_{<n} \otimes V_n$   
   $T \leftarrow_{\cup}$  right vert. of all  $V_n \otimes T_{\leq n}$  and  $V_{<n} \otimes T_n$   
   $T \leftarrow_{\cup}$  merges of all  $T_n \otimes T_{\leq n}$  and  $T_{<n} \otimes T_n$   
return  $T_{\mathcal{X} \rightarrow \mathcal{X}}$ 
```

The parser fills the chart bottom-up using the algorithm shown in Algorithm 6.1. Here, *merge* refers to the inverse of a *split*, *left* and *right verticalization* refer to combining a left or right child with a verticalization item, respectively. T_n and V_n refer to the sets of chart items with a surface coverage of n slices, and \otimes creates the pairs of those items that are adjacent (i.e., their connecting slices match with respect to position and content) and have compatible IDs.

The inner structure of each operation is parsed by inverting the operation, computing all possible inputs. For *spread* and *freeze*, this is trivial since their parent elements are unique, if they exist. For *split*, all possible parent transitions are computed that generate every note in s' using all mandatory edges in t'_l and t'_r (and possibly other edges that have been dropped and thus not included in t'_l and t'_r).

A reference implementation of the parser written in Haskell is provided.⁹

6.4 Discussion and Conclusion

The protovoice model is flexible enough to express highly complex configurations of free polyphony. However, this generative power comes at the cost of being highly ambiguous. The suspension sequence in Figure 6.4, for example, has 131 valid derivations, while the first half measure (including the upbeat) of the Bach example (Figure 6.1) already has 119,940 derivations. While this flexibility of the model allows analysts to express very subtle interpretative nuances, it also generates the problem that a single piece or excerpt has too many derivations to reasonably compare, and that any non-trivial piece takes far too long to parse exhaustively in practice. The first problem can be solved by introducing a probabilistic variant of the model that weights derivations according

⁹<https://github.com/DCMLab/protovoices-haskell/tree/ismir2021>

to their probability (Abdallah, N. E. Gold, and Marsden 2016; Harasim, Rohrmeier, et al. 2018). The second problem might be resolved by a heuristic parser that does not guarantee globally optimal solutions.

There are structural configurations assumed in some theories that require an even higher flexibility than what is provided by the protovoice model. For example, Schenkerian theory allows the *unfolding* (i.e., horizontalization) of entire progressions (such as the parallel thirds 3-2 and 1-7 in Figure 6.4) into a single sequence (such as 1-7-3-2(-1)). Such an operation would either require the progression to be represented as a single entity (to which the operation could be applied), or the ability to apply operations to non-entity contexts (similar to how spread is applied to two transitions and a slice).

The inner structure and operations of protovoices are similar to those of MOP-based approaches (Yust 2006; Kirilin and Utgoff 2008; Kirilin and Thomas 2015) for monophonic and homophonic sequences. From these, the model inherits the ability to represent double parents and, by extension, lines of notes with a start and a goal. However, protovoices use these ideas to solve the much more complex problem of free polyphony. The key insight that makes this extension possible is the separation of adjacency on the surface and adjacency in a line of notes, and the explicit representation of line adjacency in the protovoice graph. In monophonic sequences, surface and line adjacency seem to be the same, but even this assumption does not generally hold: As the example of implied polyphony shows, even monophonic voices can (and generally do) have a polyphonic latent structure. Put bluntly, there is no such thing as a monophonic melody.

The outer structure (and its integration of inner operations) is similar to an approach presented by Marsden (2010), that parses single-sided Schenkerian operations based on a grammar on slices. In particular, Marsden’s grammar uses *context notes* to model conditions of two-sided operations, which makes the grammar context-sensitive in a very similar way as protovoices.¹⁰ While Marsden’s model does not rely on explicit voices – and thus in principle can parse inputs in free polyphony – it also does not generate voice-like structure among the notes, but rather individual binary dependency relations. A similar point can be made for models working on piano-roll representations such as many neural network approaches (Chi et al. 2020; Z. Wang, D. Wang, et al. 2020; Z. Wang, Zhang, et al. 2020): While they can work with freely polyphonic inputs, they generally do not explicitly establish polyphonic *structure* among the notes in the score.

There is, however, a deeper, more philosophical difference between the protovoice model and the other approaches based on Schenkerian analysis: The protovoices attempt

¹⁰In (Marsden 2010), this context-sensitiveness is handled by parsing with a context-free parser and then removing inconsistent derivations, while the protovoice parser only constructs consistent derivations, but this is just an implementation detail.

to isolate and formalize the *structural principles* and *primitives* that give rise to free polyphony, instead of encoding the higher-level concepts and operations of a particular analytical framework. The two structural principles here are *elaboration* and *recursion*, where the former consists of the application of primitives and the latter just arises from the fact that elaboration can be applied to the output of a previous elaboration of the same kind. The structural primitives boil down to essentially two operations: stepwise insertion of notes (in all its variants) and horizontalization of simultaneous elements, which operate with the two basic relations on simultaneity and sequentiality in complementary ways.

These operations are *primitive* for two reasons: First, they provide what can be considered the lowest level of musically meaningful relations. Even more basic representations of music (such as audio or piano-roll representations) do not express musical relations (except incidental simultaneity) explicitly. Second, the basic entities and relations can be combined to express higher-level entities and relations from more specific analytical frameworks, such as different forms of harmonic analysis, Schenkerian analysis, or schema theory. A simple example of such a high-level concept can be seen in Figure 6.4, which constructs the voice-leading pattern of a 2-3 *suspension* in a principled way: first, a progression is generated that moves two voices down in parallel thirds, then another time interval is inserted in which the upper voice moves while the lower voice remains, creating the dissonant second. Similarly, the derivation in Figure 6.2 explicitly constructs an *initial ascent* (Cadwallader and Gagné 2011) from D to F and the harmonic progression $I - V^7 - I$, and describes their relation to the musical surface. The preparatory function of the dominant chord and its dependency on the tonic (Rohrmeier 2011) are even reflected by its notes, which are all ornaments of the following tonic harmony.

The structural principles and primitives postulated by this model are certainly not exhaustive. For one, they do not account for musical parameters such as harmony and key, timbre, or rhythm and meter. Furthermore, there might be additional structural primitives that establish other relations between objects than stepwise motion and simultaneity. Finally, there might be other relevant structural principles, such as abstraction of particular configurations into patterns, or the repetition of complex structures or patterns. However, since principles and primitives are generally orthogonal, the current model can be considered as a module of a more comprehensive model of musical structure.

6.5 Acknowledgements

We thank all members of the Digital and Cognitive Musicology Lab at EPFL (in particular Petter Ericson and Daniel Harasim) as well as the anonymous reviewers for their valuable feedback, and Claude Latour for generously supporting this research. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 760081 – PMSB.

7 Protovoice Theory

7.1 Introduction

The previous chapter introduced the protovoice model from a formal perspective, describing its mathematical definition and an algorithm for parsing pieces into protovoice derivations. This chapter complements this formal treatment with musical intuition. It motivates the design of the model from musical phenomena (such as implied polyphony and harmonic reduction, Section 7.2), provides a musical interpretation of the relations and entities it proposes (Section 7.3), and relates it to existing theoretical and formal frameworks (Section 7.4). The chapter then proceeds to present a number of analytical examples, discussing how exactly a protovoice derivation captures different analytical aspects: Section 7.5 shows the various ways in which monophonic melodies can (and usually do) exhibit an underlying polyphonic and harmonic structure, and how this phenomenon affects explicitly polyphonic pieces consisting of several such lines. Section 7.6 extends these ideas to pieces written in free polyphony, i.e., generally following voice-leading principles without a strictly polyphonic organization.

While voice-leading structure is notoriously difficult to model in the absence of explicit voices, applying the protovoice model to free polyphony reveals that free polyphony can be understood as a sibling of rather than a deviation from strict polyphony, expressing the same latent structures that are underlying monophony and strict polyphony. Section 7.7 then turns to the relation between latent entities (such as chords or voice-leading schemata) and the surface notes, arguing that latent entities are directly represented at intermediate states in a protovoice derivation. Finally, Section 7.8 addresses relations between latent structures, showing how functional harmonic relations emerge from protovoice-leading relations between chords, and how harmonic syntax is a special case of the tonal syntax described by the protovoice model.



(a) The full score.



(b) A reduction to the implied note simultaneities.

(c) The underlying abstract pattern (in pitch classes).

Figure 7.1 – The beginning (mm. 1 and 2.1) of Invention No. 13 in A minor, (J. S. Bach, BWV 784).

7.2 Motivation

One of the musical phenomena that the protovoice model seeks to explain is that of *implied polyphony* (also *latent*, *virtual*, or *pseudo-polyphony*, *compound* or *polyphonic melody*, or *melodic fission*; see e.g., Aldwell and Cadwallader 2018; Piston 1947; Huron 2016; Davis 2011). Figure 7.1a shows an example of a piece that exhibits implied polyphony, Johann Sebastian Bach’s Invention in A minor (BWV 784). Like all of the Inventions, the piece consists of two parts or voices, one for each hand of the keyboard player. However, each of the voices decompose into several concurrent lines: In the first half of measure 1, the right hand begins with two lines that move upward in parallel, one that starts on C5 and continues to D5 and another one that starts on A4 and continues to B4, before both lines meet on the C5 on beat 3. There can be different interpretations of how the two lines are continued (the upper line could be heard as continuing to E5 instead of going back to C5), but we will, for the sake of the example, go with the current interpretation for two reasons: first, the left hand repeats the pattern in the second half of the measure but does not contain a corresponding E4 (although the E4 in the right hand could be argued to take its place); and second, the D is harmonically a dissonant note, so it might be preferred to resolve downward (as shown in Figure 7.1c).¹ In between

¹This example shows a source of ambiguity that can be encountered in many cases. The line C5-D5 at the beginning of the piece can be very well heard to be continued to E5. However, this interpretation

these two lines, the right hand occasionally jumps back to the third line that remains on E4. The same pattern is repeated one octave lower by the left hand in the second half of the first measure. It seems that, while each of the explicit voices is technically strictly monophonic in the score, there is a polyphonic structure of several lines underlying each of them. Moreover, these lines give rise to a number of vertical entities that are not directly present in the score. The two upper lines, for example, seem to move in parallel thirds before meeting: (A,C)-(B,D)-C. Taking both hands together, a sequence of chords arises: Am, E⁷, Am, E⁷, Am. Even though at most two notes sound at the same time, “quasi-simultaneous” groups of up to 5 different notes emerge from the texture (Figure 7.1b).

While the phenomena of implied polyphony and implied harmony are well known, a number of observations can be drawn from the above example. First, the concept of a voice is more complex than a simple sequence of notes. Voices can exhibit additional internal structure such as several independent lines and notes that are presented sequentially but can be understood as simultaneous on a higher level of abstraction. However, the implied lines themselves cannot have this kind of internal structure.² Thus, unlike the name *implied polyphony* suggests, the implied structure does not consist of “voices” in the traditional sense, but is of a different, more fundamental kind which we call *protovoices*. As irreducible building blocks, protovoices are simpler than voices in that they always move stepwise³ (modulo octave displacement), but they can form a complex, hierarchical network between notes, with subordinate protovoices connecting higher-level lines.

An example of this network of lines can be seen in the abstract pattern that is underlying the beginning of the Invention (Figure 7.1c, expressed without specific octaves). On the most abstract level, a static A minor triad is held, with one proto-voice per chord

is challenged by external factors (such as the parallelism to the same pattern in the left hand), as well as general style regularities or conventions (such as the downward resolution of dissonant tones). Put more generally, the analyst has to balance “bottom-up” influences (the concrete notes at hand) with top-down influences (the context of these notes within the piece or within a style).

²It would be conceivable that implied voices are again implicitly polyphonic, recursively implying another level of underlying voices. However, if the listener is supposed to eventually resolve the latent structure, this nesting cannot continue ad infinitum and must thus reach a point where the implied voices are strictly non-polyphonic. This is the structure of interest here.

³The definition of a step depends on the context in which the model is applied. In the context of this chapter, which focuses on largely diatonic music, a step is defined as within the range of a generic second (including augmented seconds). In a different scale system, steps may be defined differently. Importantly, stepwise motion is, formally speaking, not an axiom of this model but a corollary. It follows from the fact that all elaboration operations (Section 7.3) maintain stepwise connections between notes. A similar model with different basic operations might therefore not have this property. However, these particular operations are chosen based on the assumption that in diatonic music the fundamental horizontal relations between notes (repetitions, neighbors, and passing notes) are based on diatonic steps.

tone (disregarding instances of the same chord tone in different octaves for now). This triad is embellished by inserting a number of ornaments: The top-level protovoices are ornamented with a repetition of E and two neighbor notes G \sharp and D. In addition, the passing note B connects the static lines on A and C, creating a subordinate protovoice.⁴ This example also shows that structure does not only arise within explicit voices, but can even go across them: In the second instantiation of the pattern, the neighbor note motion A-G \sharp -A is realized by the G \sharp 4 and the A4 from the right hand (beats 1.4 and 2.1), but there are two ways of identifying its first note: either as the A3 from the left hand (beat 1.3), or as an *implied* A4 that is not present in the score. For the sake of simplicity, we will generally avoid assuming additional notes, since the goal is to interpret the given notes of the score, but exceptions are made in cases where the score is an incomplete representation of a piece (such as a melody with an implied harmonization, e.g., in a lead sheet). Instead, the G \sharp 4 is here interpreted as a neighbor between the A3 in the left and the following A4, connecting the two surface voices.⁵

The relationship between the implied vertical structures and traditional notions of verticality (e.g., vertical intervals and chords) mirrors the relation between voices and protovoices. Implied verticalities are more general than chords as they can include sonorities that are not strictly chordal: On beat 1.2, for example, one might already (or retrospectively) hear an E⁽⁷⁾ chord with a suspended 4th (A3), resolving to G \sharp on the next sub-beat. At the same time, implied verticalities have a more complex organization than the literal surface verticalities, since they exist at several levels of abstraction, with the surface sonorities at the bottom and the static A minor triad at the top. An intermediate verticality can again be seen in Figure 7.1c. The E⁷ chord (generated by the ornamentation of the A minor chord tones) is at no point literally present on the surface but is implied by the notes during beats 1.2 (minus the suspension) and 1.4. On a higher level, however, the ornamental role of the E⁷ chord tones leave only the A minor triad as the implied verticality.

At first sight it might seem superfluous to introduce concepts such as “protovoice” and “implied verticality”. After all, more familiar concepts such as voices and chords already exist and seem to work well in everyday use. First, note that related concepts already exist in some analytical frameworks, e.g., linear progressions and unfoldings in Schenkerian

⁴From the reduced version of this pattern alone, a number of other explanations would be possible where B is another neighbor or D another passing note. However, the concrete realization of the pattern on the surface and a preference for dissonances to resolve downwards speak in favor of the passing note interpretation. Note that each of these interpretations can be expressed in the protovoice formalism.

⁵In a similar fashion, the pedal-like E4 in the first half of the measure is continued as both a repeated E5 (right hand) and a repeated E3 (left hand) in the second half of the measure. In this understanding, octave information is still generally considered when identifying relations between notes, but these note relations can at times go across different octaves.

analysis (Cadwallader and Gagné 2011). Furthermore, the need for more fundamental concepts becomes clear at points where the traditional concepts break down. This is, for example, the case when approaching the computational problem of voice separation (recovering the voice structure underlying a set of notes) with formal methods. As noted, for instance, by Cambouropoulos (2008), the answer to this question is not only ambiguous, it is not even clear what *kind* of answer is expected: the explicit voices notated by the composer, a set of *auditory streams* (Bregman 1990), or a network of protovoices? Each of these structures follows different principles and would therefore be recovered in different ways from the given surface notes. But even outside the domain of algorithms, traditional notions of voice leading seem insufficient: If voice leading is essential for tonal structure, what does that mean for contexts in which explicit voices are not given (free polyphony)? What if explicit voices are given, but they imply an additional latent polyphonic structure, as in the example above? The protovoice model answers these questions as follows: The fundamental structural categories of tonal music are not conventional voices or chords but protovoices and implied verticalities. When explicit voices and chords are meaningful, there is a clear relation to the underlying protovoice structure. When they break down, the protovoice structure still explains the fundamental relations between the notes.

An analogy may help to illustrate the relation between the different conceptual levels. The four conventional states of matter (solid, liquid, gas, and plasma) are useful and intuitive concepts for everyday purposes and cover most of the states of matter relevant to our daily lives. Knowing whether we come into contact with water, ice, or water vapor is essential, but knowing about other states of H₂O is mostly irrelevant to everyday life. From a physical point of view, however, there are many more states of matter than the above four. A more general perspective on matter is obtained by viewing matter as a collection of molecules, atoms, or even elementary particles and their relations rather than a substance with certain macroscopic properties. The particle view is less convenient for everyday purposes, but it is more fundamental than the conventional view and still offers an explanation in situations when the conventional states of matter do not apply (e.g., in glasses, liquid crystal states, or superfluids) when the objects of interest are the molecules, atoms, or elementary particles themselves, or when trying to explain how the macroscopic properties of a substance arise from the properties and the organization of the particles that it consists of (e.g., why some substances have a high electrical conductivity, or why they are transparent). At the same time, the configurations of particles can be related back to substance states and even reveal differences underlying similar high-level phenomena, such as different kinds of atomic structures that are “solid”. In this sense, the protovoice model can be regarded as a particle model of tonal structure. It seeks to provide the “atomic” (in the sense of the word) categories of tonal structure, describing the fundamental relations between notes

and clarifying higher-level concepts without invalidating their usefulness.

7.3 The Protovoice Model

The goal of the protovoice model is to formally describe the relations identified in Section 7.2, that is how individual notes are related to each other, and how the surface of a piece relates to latent musical entities, such as voices, harmonies, or schemata. While a complete formal specification of the model is given in Finkensiep and Rohrmeier (2021), this section will give an overview over the concepts that are relevant for understanding the following analyses. The model follows three principles:

1. The most fundamental structural relations between notes are *simultaneity* (vertical) and *sequentiality* (horizontal). These relations are expressed explicitly in the model and all of its operations transform them in a consistent way. Sequentiality implies more than non-simultaneity, it expresses that a note is thought of as a successor of another note. A note can be the successor and predecessor of several other notes, but the sequentiality relation always respects temporal order, resulting in a *directed acyclic graph* (DAG) structure. Simultaneity, on the other hand, can connect groups of several notes (called *slices*) that are considered mutually simultaneous, and each note belongs exclusively to one such slice. Long notes that overlap with several non-simultaneous notes are split into several parts.⁶
2. Each note (or note part) has a *function* in its context. More specifically, notes are considered ornaments of other notes (their *parents*). Structure is created by inserting new notes, elaborating and complexifying simpler configurations. A complete analysis of a piece assigns such a function to every note in the piece, giving each note an interpretation relative to the other notes in its context.
3. A note that is considered an ornament can itself become a new parent of an elaboration. This simple property makes elaboration *recursive*: a piece is generated by starting with a simple structure and repeatedly elaborating it into the full surface. The starting point for this process is a single slice of *root notes* that do not have parents.⁷ Repeated elaboration is also what links latent entities (e.g., harmonies or schemata) to the surface.

⁶When a surface note is split into several parts, these parts are connected by special sequentiality connection that indicates a tie so that the full note can be recovered. On higher levels, these tie connections do not exist, so tied and repeated notes are not distinguished in latent structure. However, the tie connections on the surface level ensure that different parts of the same note are still derived in a consistent way based on repetition.

⁷Formally, the parents of the root notes are the beginning and end of the piece, so they can be thought of as elaborating the empty piece.

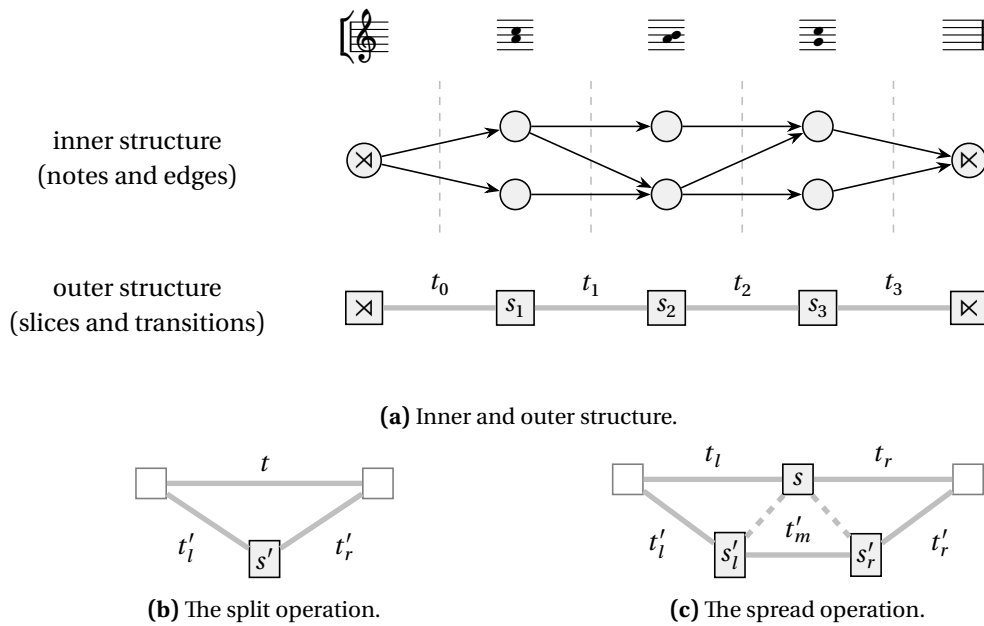


Figure 7.2 – The Representation of a piece during a protovoice derivation (a). At each point during the derivation process, the piece consists of a sequence of slices and transitions (outer structure), which contain notes and edges, respectively (inner structure). The outer structure can be transformed by splitting a transition (b), or by spreading a slice (c).

These three principles are formalized as a *generative process* that derives a piece from a number of elaboration operations. At each point of this process, the fundamental relations between notes are represented as follows: The piece consists of a series of slices, each containing a group of notes. Between two adjacent slices, a *transition* contains the *edges* that represent sequential connections, i.e., protovoices (Figure 7.2). At a given derivation step, only edges between adjacent slices can be represented; long-range connections are represented by direct connections at an earlier derivation step.⁸

The outer structure (slices and transitions) can be elaborated by two operations, *splitting* and *spreading*. When visualizing the derivation of outer structure, we use a notation that is adapted from Yust (2006). The starting point of the derivation (usually the “empty piece” or some given structure) is shown horizontally at the top. Transformations of existing structure by splits or spreads are shown below the old structure. The derivation, read from top to bottom, thus describes how a simple structure is transformed into the more complex surface structure.

⁸An example of this can be seen in Figure 7.3: After Step 2 (Figure 7.3c), there is a direct edge between the two As. After Step 3 (Figure 7.3d), this connection has been replaced with edges to and from the neighbor note G#, since the two As are not directly adjacent anymore. The full derivation, however, reflects both connections.

	Parent Object	Operation / Note Function	Result
split	single note ○ (left or right)	repetition of a single parent note	
		stepwise neighbor of a parent note	
	regular edge ○ → ○	repetition of both parent notes	
		neighbor of both parent notes	
		repetition of one parent note, keeping an edge to the other parent	
	passing edge ○ - - - > ○	incomplete passing note (left or right)	
complete passing note			
top-level edge ⊗ → ⊗	root note		
spread	single note ○ (parent slice)	inherit in left child slice inherit in right child slice inherit in both child slices	

Table 7.1 – Note-generating operations. Each operation transforms the parent object (a note or an edge) into the structure shown in the result column.

A split (Figure 7.2b) takes an existing transition t and replaces it with a new slice s' and two new transitions t'_l and t'_r that connect s' to the two adjacent slices. During a split, the new slice s' can be filled with new notes by creating ornaments (Table 7.1). Single-sided ornaments (one-sided neighbors or repetitions) are attached to a single parent in either of the parent slices, creating a new edge in t'_l or t'_r . Double-sided ornaments (neighbors and repetitions with two parents) can be created along existing edges in the parent transition. The parent transition may also contain passing edges, which must be filled exactly once with a passing tone that is either a step away from one of its parents (generating a regular edge on one side and a new passing edge on the other side), or connects the two parents if the interval between them is sufficiently small (generating regular edges on both sides). Since all ornamentation operations require notes to be at most a step away from their parents, regular edges are guaranteed to be stepwise too.⁹

⁹The notion of a step is generally flexible and can be adapted to the style or scale system used. In the

The generating operations ignore the octave of a note, so a C4 can be used as a repetition of a C3, and a Bb3 can be a neighbor of an A2. Nevertheless, the octave of each note is generally known and the register of a note can be used to make analytical decisions. Finally, new passing edges can be introduced in both child transitions, t'_l and t'_r , creating new subordinate lines between formerly unconnected protovoices.

Spreading is the complementary operation to splitting (Figure 7.2c). It replaces a slice s with two new slices s'_l and s'_r by distributing the notes in s to s'_l and s'_r , either moving a note to one side or duplicating it on both sides. This operation makes notes that are considered simultaneous on a higher level non-simultaneous. As opposed to the ornamentation operations in a split, the octave of a note is respected in a spread and must remain the same. The outer child transitions (t'_l and t'_r) are adapted from their corresponding parents (t_l and t_r) by removing edges connected to notes that have been moved to the other side. The inner child transition can be filled with regular edges between repeated notes (or notes of the same pitch class), or with new passing edges between other pairs of notes. Together, splitting and spreading correspond to transformations of the two basic relations: splitting modifies sequentiality by inserting new ornaments while spreading modifies simultaneity by horizontalizing the notes in a slice.

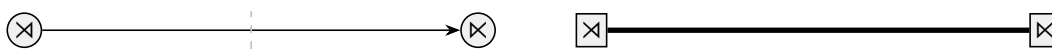
Figure 7.3 shows the step-by-step derivation of the abstract pattern in Figure 7.1c. Starting with an empty piece (Figure 7.3a), the top-level transition is split and the notes A, C, and E are inserted as root notes (Figure 7.3b). This top-level slice is spread, duplicating all of its notes on both sides and preparing the passing motions from A to C and from C to E (Figure 7.3c). The repetition edges between the Es and the As are also kept to use them for further elaboration in the next step. Finally, the middle transition is split again to insert the notes G#, B, D, and E as neighbors, repetitions and passing notes, filling the previously introduced passing edges (Figure 7.3d).

In most of the following examples, the outer-structure notation (as in the right column of Figure 7.3) will be used to concisely display a derivation. In addition, a selection of inner connections (drawn as curved arrows) and note-type markers will be used to highlight certain relations and functions of interest. An example is shown in Figure 7.3d, where the passing motion A-B-C is directly shown in the outer structure. Since this format cannot display all details of a derivation, a set of interactive visualizations that can be used to explore the full derivations will be provided in the supplementary material.¹⁰

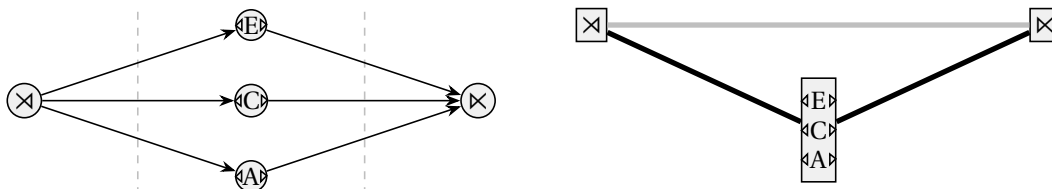
context of this paper it is assumed to correspond to diatonic steps, so that regular edges span seconds and unisons (including augmented seconds and altered unisons). This also implies that the “stepness” of an interval does not depend on its acoustic size but on how it is logically conceived in the diatonic system (an augmented second is a step, a diminished third is not).

¹⁰The examples can be found at <https://github.com/DCMLab/protovoice-annotations/tree/>

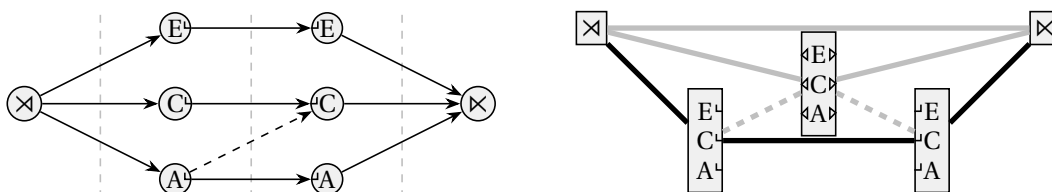
(a) The derivation starts with the empty piece.



(b) Step 1: The root notes (an A minor triad) are inserted through a split operation.



(c) Step 2: The root triad is spread and repeated on both sides, introducing a passing edge.



(d) Step 3: Another split inserts the E⁷ chord between the two Am slices.

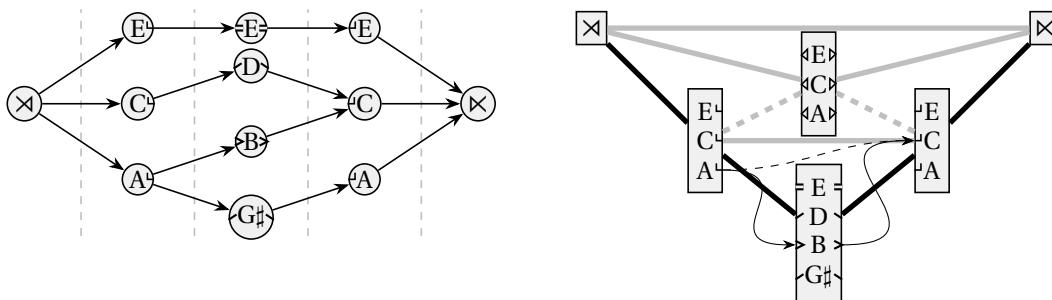


Figure 7.3 – The protovoice derivation of the pattern in Figure 7.1c. The left column shows the inner structure of the surface at the current step. The right column shows the full derivation of the outer structure up to the current step, with the current surface (as shown on the left) indicated by black transitions. Selected connections from the inner structure can be shown in the outer derivation as curved arrows (d).

7.4 Related Theoretical and Formal Frameworks

7.4.1 Theoretical Frameworks

The protovoice model as presented above builds on a number of ideas from other analytical frameworks. For one, it is closely connected to Schenkerian analysis (Schenker 1979; Cadwallader and Gagné 2011), sharing some of its motivating phenomena such as tonal relations between notes, the emergence of these relations from elaboration of simpler structures, and the generalization of voice leading principles to free textures without explicit voices. As a result the two frameworks have some commonalities such as the focus on stepwise relations, and a set of roughly corresponding entities: The Schenkerian *Zug* is approximately represented by a passing edge, connecting two endpoints with a stepwise line of notes. The concept of *unfolding* (*Ausfaltung*) is reflected in the spread operation and constitutes the primary mechanism behind latent polyphony in both cases (as will be discussed in Section 7.5).¹¹ However, not all Schenkerian concepts have a counterpart in the protovoice model. This includes non-stepwise operations such as consonant skips and arpeggiations, which are not assumed to establish a fundamental relation in the protovoice model (although they are, to some extent, captured in the spread operation); as well as higher-level constructions such as voice exchanges, which can be expressed in a derivation but do not have a specific corresponding model entity. Even for the operations that do have a protovoice counterpart, these counterparts are usually less restrictive. Not every spread is an unfolding, for example, as spreads can also be used to arpeggiate or simply prolong a slice.

Other differences between the two frameworks stem from the goals that they do not share. For instance, the protovoice model does not aim to capture contrapuntal rules in a more general setting. Instead, the concept of a voice in the traditional sense is abandoned altogether and replaced with the notion of protovoice connections, mapping “voice leading” phenomena to protovoices, which are more strict internally (requiring stepwise motion) but less restrictive in their organization (dropping contrapuntal constraints).

main/theory-article. Analyses are encoded as `.analysis.json` files, the `.musicxml` files contain the corresponding scores. Analyses can be explored using the protovoice viewer at <https://dcmlab.github.io/protovoice-annotation-tool/viewer/>. The reader is also invited to try encoding their own analyses of the provided scores using the annotation tool at <https://dcmlab.github.io/protovoice-annotation-tool/>. More information on annotation tool and viewer is provided in Chapter 8.

¹¹More generally, the notion of latent polyphony in which every melody is potentially polyphonic and the underlying polyphonic structure can be distinct from explicit surface voices (which is argued for in Section 7.5) is inherited from the Schenkerian practice to distinguish between surface voices (such as a written melody) and the structural voices (such as the structural soprano, bass, and inner voices) on a given level.

One reason for this design decision is the goal to make every aspect of the system fully explicit, which requires the exact conditions for the application of an operation to be known in every situation. In Schenkerian analysis, on the other hand, the applicability of an operation is not always well-defined and can depend on reasoning by analogy (such as the application of counterpoint rules to free textures), prior assumptions about the structure of a piece (e.g., the existence of an underlying fixed set of voices on a given level), or even a particular reading of the theory.¹² While it is debated whether Schenkerian theory can be rephrased as a consistent, explicit, and formal theory without departing too far from the original idea (Temperley 2011; Brown 2005; Westergaard 1975; Narmour 1977; Narmour 1983), at least making prior structural assumptions seems problematic for a general model of tonal structure and its perception. The protovoice model thus trades some analytical specificity for generality and consistency, as argued in Section 7.2.

A similar point can be made about the background structure that in Schenkerian analysis is assumed to underlie tonal pieces, in particular the *Ursatz*. From the perspective of interpretation, the understandability of a piece does not depend on the presence of an *Ursatz*. Instead, the use of an *Ursatz*-like background structure is a stylistic regularity, a compositional practice. When trying to understand a given piece, it cannot be assumed that this piece also follows this practice. For this reason, no primary background structure is assumed in the protovoice model when generating a complete piece from scratch. Moreover, the model itself is largely agnostic to the specific background structures and latent configurations that are used in a particular analysis. Instead, its focus is on the *connection* between the latent configurations and the surface notes. While a piece can be derived from “nothing” via a triadic or non-triadic root slice (or even using an *Ursatz*-like construction in the beginning of the derivation), the model can similarly derive a surface passage from a given harmonic sequence or voice-leading schema, as will be shown in Section 7.7.

Finally, the protovoice model is purely descriptive and does not make any musical quality or value judgments. Instead it defines a space of possible interpretations for a given piece in terms of tonal relations from a highly abstract and idealized perspective. This reverses the roles of a given stylistic ideal and a piece as a more or less perfect instantiation of that ideal to a given piece and a more or less meaningful interpretation of the piece: While the model can in principle be applied to any given piece, the resulting interpretations

¹²The protovoice model as applied in this chapter also relies on musical intuition for selecting one or several plausible analysis among many possible but implausible alternatives. However, the space of possible analyses, the structural relations each of them entails, and their interpretation within the context of the model are well-defined and fully explicit. Ranking the plausibility of the analyses is simply outside the scope of the model at this point, but possible ways of extending the model in this direction will be discussed in Chapter 9.

will only be meaningful for pieces from Western common practice tonality, styles that inherit some of its aspects (e.g., some modern Western styles), or styles that employ similar abstract principles.

In many respects, the protovoice model is closer to Westergaard's theory of tonal structure 1975, which inherits but reframes many Schenkerian ideas. Not only is Westergaard's theory strongly generative in the interpretation sense ("the successive stages in the generation process show how we understand the notes of that piece in terms of one another", Westergaard 1975, p. 375). The basic operations assumed by Westergaard (rearticulation, neighbor embellishment, connecting step motion, and arpeggiation) correspond to the basic ornament types in the protovoice model (repetition, neighbor, passing note) except for rhythmic information. In particular, Westergaard considers arpeggiation to always be the result of horizontalizing a vertical interval (or chord), which is reflected by the spread operation in the protovoice model. Westergaard's theory, however, is still presented in terms of classical voice-like "lines". While these lines can be combined to create compound lines, the nature of lines (what makes a "line" a "line") is less systematic than the characterization of protovoices given here, starting from contrapuntal voices but moving to increasingly loosely defined note sequences in the context of free textures.¹³

Besides Schenkerian analysis, the protovoice model is closely related to classical frameworks of harmony and voice leading (e.g., Aldwell and Cadwallader 2018), as well as to schema theory (R. Gjerdingen 2007). The idea of reduction, ornamentation, and latent entities is (more or less explicitly) present in all of these theories, e.g., through the notion of non-chord tones. The entities and relations of the protovoice model live generally on a more fundamental level than these frameworks, but interacts with them in ways that are described in the following sections.

7.4.2 Computational Models

Models of Voice Structure

On the side of computational models, the main approach to modeling polyphony is *voice separation*, which has a psychological counterpart in *auditory scene analysis* and *stream segregation* (Bregman 1990). The central metaphor of this approach is that of an idealized sound source that generates a stream of events. In auditory scene analysis,

¹³"We will use the word 'line' to refer to any series of notes that we have reason to understand as connected to form a single strand. There are different kinds of reasons for understanding one note as connected to another and, hence, different kinds of lines." (Westergaard 1975).

these sound sources can (less metaphorically) correspond to physical sound sources, and the task of the perceptual system is to identify these sources from the combined acoustic input. Through the mechanism of *sequential grouping*, acoustic events can be assigned to the same sound source and are thus perceived as a *stream*. Voice separation adapts this idea for identifying voices in a set of note events: a voice is a stream of events with a common idealized source that is obtained by sequential grouping (see e.g., Huron 2016), and a number of voice separation algorithms are based on this idea (e.g., Kilian and Hoos 2002; Kirilin and Utgoff 2005; Duane and Pardo 2009; Makris et al. 2016; McLeod and Steedman 2016; de Valk and Weyde 2018). This approach comes with several problems: For one, what constitutes a voice is not clear in the general case, especially in the presence of latent polyphony (Cambouropoulos 2006). Moreover, even when explicit voices are given, it is not guaranteed that these behave according to the principles of auditory streams. For example, the voices in a fugue are usually more complex, each voice constituting an independent melodic line, than in a chorale where bass, melody, and inner voices each have a different function and character. (The problem of the concept of voice will be discussed in Sections 7.5 and 7.6.)

Most importantly, however, streams are not the right object to look at when the interest is on tonal sequential relations. On one hand, relations such as a note being a neighbor of another note or a leading tone resolving into another note are can neither determined from stream assignment alone (which lacks the hierarchical parent-child quality of these relations), nor do they have to respect streaming (e.g., when a line is continued in a different octave). On the other hand, streams are an overly strict model for tonal structure, as they require maintaining the identity of a stream as an object distinct from other streams. In terms of tonal relations, however, there is no reason why two notes should not be able to ornament or resolve into the same parent note. Consequently, the assumption of a set of underlying “sources” seems inappropriate for these kinds of tonal structures. For this reason, the protovoice model abandons the concepts of stream and voice in favor of sets of binary relations, i.e., protovoice graphs, which can express the network structures seen in Figure 7.3.

Models of Hierarchical Structure

Perhaps the most extensive formal model of tonal structure is Lerdahl and Jackendoff’s Generative Theory of Tonal Music, or GTTM (1983). It defines a set of well-formedness and preference rules for deriving analyses of tonal structure from a given piece, covering four different aspects: *metrical structure* defines the metrical hierarchy and assigns a metric weight to each timepoint in the piece; *grouping structure* segments the piece into nested groups according to phrasing and form; *time-span reduction* augments grouping

structure by a hierarchy of *head* events, with the head being the dominant event in each group; and *prolongational reduction* expresses the hierarchy of events in terms of elaboration or prolongation, inspired by Schenkerian ideas.

Despite its name, however, the GTTM is not a generative model since it does not have an underlying generative process (see also Longuet-Higgins 1983; Rohrmeier 2007). Instead, its rules are entirely defined “bottom-up”, describing how to arrive at (and rank) analyses from a given piece. As a consequence, the exact relations between the events is not always interpretable. For example, when a *head* is selected among the candidate event in a group (the heads from the level below), there is no relation between the winner and the remaining candidates other than one being the head and the others not being the head. A protovoice derivation, in contrast, always provides an explicit and interpretable relation between superordinate and subordinate entities. Generally, the GTTM is interested in somewhat different aspects of music than the protovoice model, focusing on hierarchical ranking of events rather than the specific inter-entity relations. In particular, the GTTM does not model polyphonic structure at all, treating the surface either as a sequence of notes (for melodies) or homophonic slices with a privileged melody note. Despite these differences, there are some similarities between the two models, such as the *fusion rule* of time-span reduction, which roughly corresponds to the spread operation. The closest conceptual similarity exists between the protovoice model and prolongational reduction, both of which describe the dependencies between reference events (heads or parents) and adjacent events that elaborate these events. The details, however, differ drastically for the reasons already mentioned above. Protovoices operate on individual notes instead of events. In addition, elaboration relations have a more specific interpretation (prolongational reduction only distinguishes non-repetition and two forms of repetition) and are more constrained, which leads to an exact specification of which elaborations are legal in a given situation, and what each elaborations means.

One of the most notable shortcomings of the GTTM is its reliance on tree structures, which excludes a note being related to two parents. Lerdahl and Jackendoff themselves acknowledge this problem in the context of passing notes and two-sided neighbors (Lerdahl and Jackendoff 1983, p. 186). Solving this problem has been the motivation for a number of attempts at formalizing Schenkerian analysis using interval elaboration rather than note elaboration (Marsden 2001; Yust 2006; Kirilin and Utgoff 2008; Kirilin and Jensen 2011). The protovoice model is inspired by these approaches, inheriting their general transition-focused elaboration scheme as well as the corresponding notation. The main differences stem from the discrepancies between protovoices and Schenkerian analysis discussed above, as well as from the fact that the Schenker models are largely restricted to melodies. Yust (2015b) extends his previous monophonic model to several parallel voices, but still requires one primary voice that determines the hierarchical

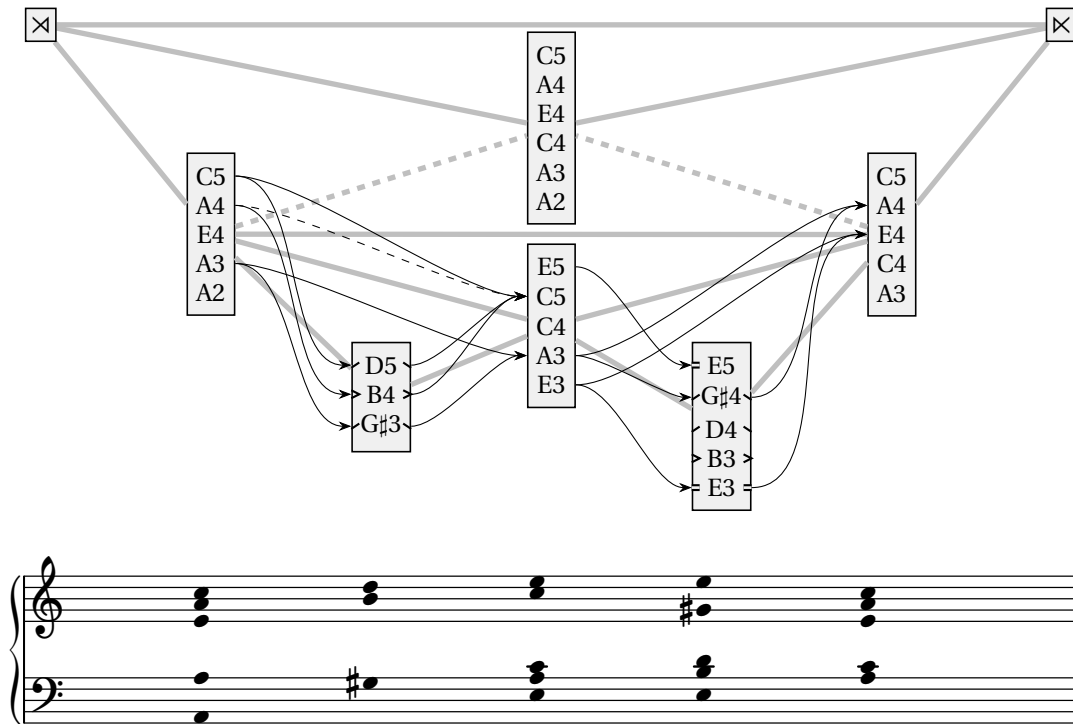
relations. Kirilin and Thomas (2015) replaces notes with slices of notes in a derivation to describe homophonic elaboration. This model is already very close to the *outer structure* of the protovoice model, but does not yet have its more flexible and principled internal graph structure. There have been other Schenkerian models that are in principle able to deal with polyphony, but those have other shortcomings, such as the single-parent problem (Marsden 2010) or being preliminary and incomplete (Frankel et al. 1978; Rahn 1979).

The second group of formal models that strongly influenced the protovoice model are formal grammars. Grammars tell an explicit generative story of structure: a sequence of items (and the latent relations between them) is generated through a derivation, a sequence of rule applications that elaborate an initial configuration. These rule applications can be associated with a semantic interpretation, e.g., functional relations between harmonies (Rohrmeier 2020a). The main limitation of classical grammars (such as context-free grammars) is that they can only generate sequences of objects, which is why they have been applied to harmonic progressions (Harasim, O'Donnell, et al. 2019; Rohrmeier 2011; Quick and Hudak 2013; Granroth-Wilding and Steedman 2014; Melkonian 2019) as well as monophonic melodies (Gilbert and Conklin 2007; Groves 2016; Abdallah, N. E. Gold, and Marsden 2016; Finkensiep, Widdess, et al. 2019; Nakamura et al. 2016) and rhythms (Melkonian 2019; Foscarin et al. 2019; Rohrmeier 2020b), but are not directly suitable for polyphonic structure, although extensions of context-free grammars are in principle able to work with structured categories and infinite alphabets (Harasim, Rohrmeier, et al. 2018). The main technical contribution of the protovoice model is the integration of the spread and split operation into a single grammatical model while still retaining parsability. Each of the operations alone could be described using an extended context-free grammar (one that elaborates either slices or transitions), but their combination falls outside the range of classical grammar formalisms.

7.5 Latent Polyphony

Section 7.2 presented an example of explicit voices giving rise to an underlying structure of lines. In Section 7.3, we discussed how this underlying structure can be derived using the operations of the protovoice model. Let us now turn our attention to how this underlying structure is related to the surface of Bach's A minor Invention (as shown in Figure 7.1a).

The first thing to notice is that both surface voices mainly use arpeggiations to turn the underlying vertical structures into two horizontal monophonic lines. In fact, the



(a) The derivation of the overarching structure.

Figure 7.4 – Derivation of the surface in Figure 7.1a, shown in three parts: the overarching structure connecting the two pattern instances (a), the first pattern instance including a suspension (b), and the more simple second pattern instance (c). The derivations in (b) and (c) are continuations of (a). Selected protovoice connections are shown as arrows to indicate tied notes on the surface and to highlight specific substructures. (Continued on the next page.)

derivation of the second pattern instance (beats 1.3 to 2.1) relies exclusively on this technique. Figure 7.4c shows a derivation of this segment starting from the pattern structure (now with octave information). In this derivation, a binary division of the top-level slices (through repeated spread operations) is used to create the arpeggiated surface. This analysis is suggested by the score since the binary metrical hierarchy is emphasized by the composed rhythms. In principle, however, other derivations would be possible too, e.g., repeatedly branching off slices with the duration of a 16th note from either side. Which derivation is chosen for an arpeggiation is an analytical decision and depends on which latent verticalities the analyst wants to assume.

The first instance of the basic pattern on beats 1.1 to 1.3 is slightly more complex since it involves a suspension (Figure 7.4b). This suspension is generated in a top-down fashion: From the first top-level slice to the second, two protovoices move in a consonance,

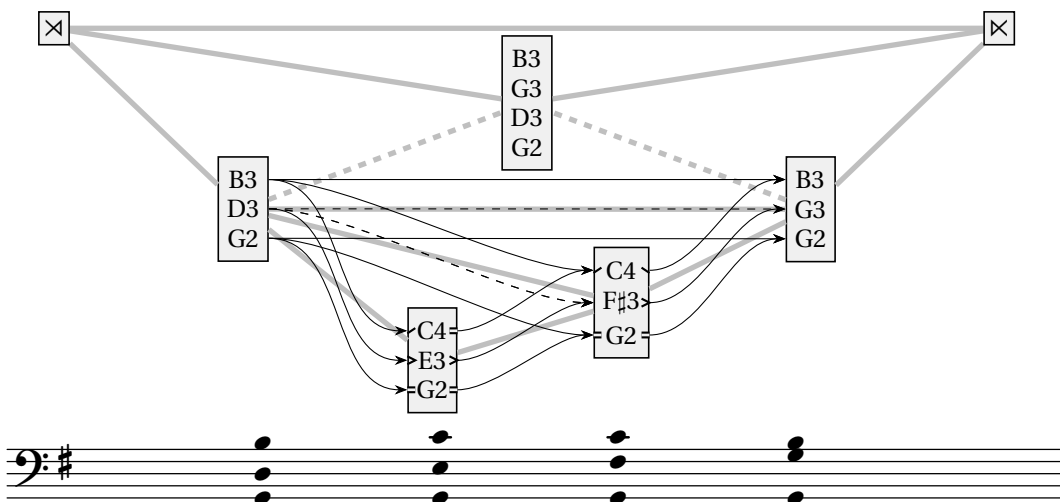
one from A4 to B4 and one from A3 to G \sharp 3. This progression is elaborated by a split in which the upper voice already moves on (by repeating the B4 from the right parent slice) while the lower voice stays (by repeating the left parent A3), creating a dissonance in this intermediate slice. The dissonant slice is then further arpeggiated to match the surrounding texture.

The protovoices are generally not aligned with the surface voices. This is particularly visible in the derivation of the overarching structure behind the two pattern instances (Figure 7.4a). For example, the first instance of the changing note progression A-G \sharp -A happens entirely, in the left hand (and within the same octave). The second instance, however, while mostly in the right hand (G \sharp 4-A4, borrows the initial note A3 from the left hand). Similarly, some of the static notes are repeated in several octaves, most notably E, which means that the analyst needs to decide whether occurrences in different octaves should be treated separately, or combined into one protovoice. For example, the E3 in the fourth surface slice (the second E⁷ chord, beat 1.4) is explained as a “full repetition” of E3 on the left and E4 on the right, since the right parent slice does not contain another E3 as a possible parent. A different way of treating this phenomenon is shown with E5 in the same slice, which is explained as a single-sided “right repetition” of the E5 in the left parent slice, not connecting it to the E4. These decisions do not always make a big difference for the overall analysis, but they can be used to indicate long-term structure. In a larger context, for example, the E5 could be connected to its next occurrence, which is the beginning of a descending line in a sequential progression.

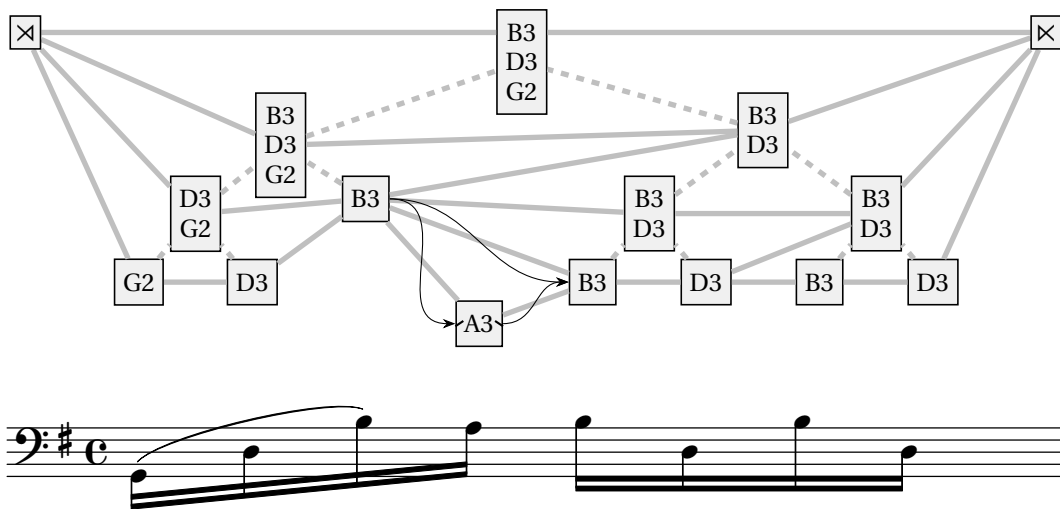
In the most extreme form of latent polyphony, a single (technically monophonic) melody outlines an underlying polyphonic structure. A famous example are Bach’s works for solo instruments. Figure 7.5a shows the well-known beginning of the Prelude from Cello Suite in G major (BWV 1007). In the first four measures, the melody uses a repeated pattern to switch between three lines: a pedal on G2, a neighbor motion from B3 to C4 and back to B3, and a linear connection from D3 to G3. The protovoice derivation of these three lines (Figure 7.5b) starts with the enclosing G major chord, which is spread to create the material for measures 1 and 4, distributing the D3 to the left and the G3 to the right, and creating a passing connection between the two. This connection is subsequently filled in by a series of splits: First, an F \sharp 3 is generated as a passing note to G3, together with C4 (neighbor to B3) and the pedal G2, to generate measure 3. Second, the passing line is completed with an E3 that closes the gap between D3 and F \sharp 3, creating measure 2 together with the repeated G2 and C4. Finally, a uniform pattern is used to generate the surface form of each measure (shown for the first measure in Figure 7.5c). As with the A minor Invention, this surface pattern uses spread operations to arpeggiate the three lines, together with a split that creates the neighbor note. The uniform application of such figuration patterns is not enforced by the model (both analytically and during generation)



(a) The score without pattern repetitions.



(b) Derivation of the underlying structure.



(c) Derivation of the surface pattern in m. 1 from the underlying G major triad.

Figure 7.5 – Mm. 1-4 of the Cello Suite in G major, I. Prelude (J. S. Bach, BWV 1007).

Figure 7.6 – Mm. 2-4 of the Partita in A minor for Solo Flute, II. Corrente (J. S. Bach, BWV 1013). The score can be derived from two lines that descend in parallel tenths. The E5 in the m. 4 occurs slightly later in the original score and has been added here to simplify the derivation.

but a sub-derivation such as the one in Figure 7.5c can be used to characterize such a pattern and reveal the shared structure of its different instances throughout a piece.

A slightly more complex example is shown in Figure 7.6. In this case, the underlying structure does not form chords, but rather two descending lines of parallel tenths (A to E and C to G \sharp). In this case, the surface realization is not uniform between the stages of the pattern. After each stage has been made sequential using a spread, some additional ornaments are added to the third stage (measure 3): The neighbor motion A - G \sharp - A is created by first duplicating the A, then inserting the G \sharp . An incomplete neighbor B ornaments the A before it moves on to G \sharp . Similarly, the lower line takes a detour through D, which is explained as an incomplete neighbor to the final E, thus approaching it from two directions, F and D. F and D are themselves connected by another passing E in measure 3.

In the previous two examples, the latent polyphonic structure was highlighted by large melodic leaps and the different registers in which the underlying lines were realized. The example in Figure 7.7 shows that this need not be the case for a melody to have an underlying polyphonic structure. While the melody of the A part of *Fly me to The Moon* (Bart Howard, 1954) moves mostly stepwise, it follows a sequential pattern of three descending lines (shown at the top) that correspond to the harmonization of the melody. Note that the middle voice is not just assumed for harmonic reasons (although the harmony validates it), but also because of the melodic skips in measures 2 and 6,

The figure illustrates the harmonic structure of the A part of "Fly Me to the Moon" (measures 1-8). It consists of three main parts: a chord progression, a simplified rhythmic notation, and a detailed harmonic analysis diagram.

Chord Progression (Top Row): Am⁷ | Dm⁷ | G⁷ | C^Δ | F^Δ | B⁰ | E⁷ | Am⁷

Simplified Rhythm (Middle Row): A single staff showing the melodic line with simplified rhythmic values (quarter notes, eighth notes, and a half note).

Harmonic Analysis Diagram (Bottom Row): This diagram maps the implied notes for each measure. Solid boxes represent the notes explicitly written in the melody, while dashed boxes represent implied notes. Arrows indicate the voice leading between notes in adjacent measures.

- Measure 1:** Implied notes include C⁵, A⁴, G⁴, and B⁴. The melody starts with C⁵.
- Measure 2:** Implied notes include F⁴, A⁴, G⁴, and C⁵. The melody moves to F⁴.
- Measure 3:** Implied notes include B⁴, A⁴, G⁴, and F⁴. The melody moves to B⁴.
- Measure 4:** Implied notes include B⁴, G⁴, E⁴, and F⁴. The melody moves to B⁴.
- Measure 5:** Implied notes include A⁴, F⁴, E⁴, and G⁴. The melody moves to A⁴.
- Measure 6:** Implied notes include D⁴, F⁴, A⁴, and E⁴. The melody moves to D⁴.
- Measure 7:** Implied notes include G^{#4}, F⁴, E⁴, and D⁴. The melody moves to G^{#4}.
- Measure 8:** Implied notes include A⁴, E⁴, C⁴, and D⁴. The melody moves to A⁴.

Figure 7.7 – The A part (mm. 1 to 8) of *Fly Me to the Moon* (Bart Howard, 1954) with a simplified rhythm.^a The reduction (top row) contains some harmonically implied notes.

^aAdapted from *The New Real Book Volume II*, 1991, Sher Music Co., Petaluma, CA.

which indicate a switch of protovoice. Unlike the two solo pieces by Bach, the melody of *Fly Me to the Moon* is not intended to be performed without harmonic accompaniment. It thus does not contain all notes of the underlying structure in its surface form, as the missing notes would be provided by the harmony. The analysis in Figure 7.7 therefore assumes a number of implied notes since a reduction of the full surface (including accompaniment) would contain these notes too.

A similar latent structure with two lines can be observed in the A part of *Con Alma* (Dizzy Gillespie, 1954; Figure 7.8). This may initially seem surprising since the melody moves almost exclusively stepwise. In fact, the melody alone can be derived from a single descending line $A\flat$ to E, in which the upwards-directed figurations in measures 3 and 7 are understood as nested neighbors and the $A\flat$ in measure 4 as an escape note (a one-sided neighbor to $G\flat$). The harmony, however, implies a different understanding of the melody: The $A\flat$ that starts as the third of $F\flat^{\Delta}$ becomes the seventh in $B\flat^7$. Consequently, the $B\flat$ in m. 3 is interpreted as the root of the chord and thus as a dissonance to the $A\flat$, which is correctly resolved in the next measure by moving the $A\flat$ down to G. Similar suspension-like structures are implied in measures 4 and 7, resulting in two implied lines that descend in parallel thirds with intermediate suspensions (Figure 7.8b). The derivation of the melody shows how the dissonant seconds in the suspension slices are spread into what appears to be melodic steps.

The latent protovoice structure of a melody does not need to move, as the two examples in Figure 7.9 show. In both cases, a static triad is elaborated by ornamenting the chord tones with neighbors and connecting them with passing notes. While the underlying structure revealed by these analyses might not be very exciting, they demonstrate how a melody is constructed within a mode, by anchoring to the more stable tones (the tonic triad) and using less stable tones for ornamentation.

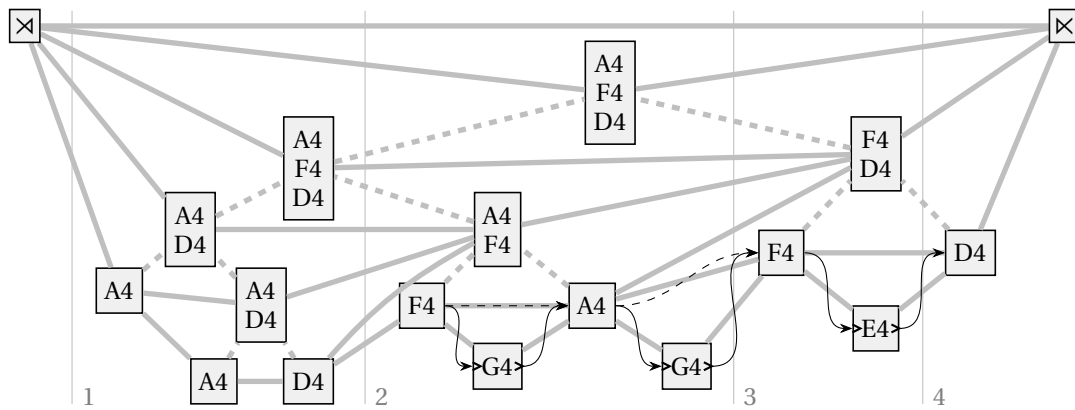
More importantly, these examples show that seemingly monophonic melodies or voices being supported by a latent structure of concurrent lines (static or moving) is not an exception but rather the default case. In this sense, melodies are almost never truly monophonic in the structural sense. To generate goal-directed motion, two notes are required, a starting point and an end point. The vertical interval between these notes serves as a frame for the melody, so even the simple and archetypical melodies such as the main theme of Smetana's *The Moldau* (Figure 7.10a) have a latent structure that consists of at least two notes. Figure 7.10b shows the generation of the underlying arch shape (black transitions, open note heads). The ascending part of the line is generated by filling the interval between the low point (E) and the high point (B), while the descending part fills the interval between the high point and the second low point. These two points thus form an implicit verticality that is prolonged from the beginning to the end and

(a) The leadsheet of mm. 1-8. Note and chord spellings have been chosen to avoid enharmonic exchange.^a

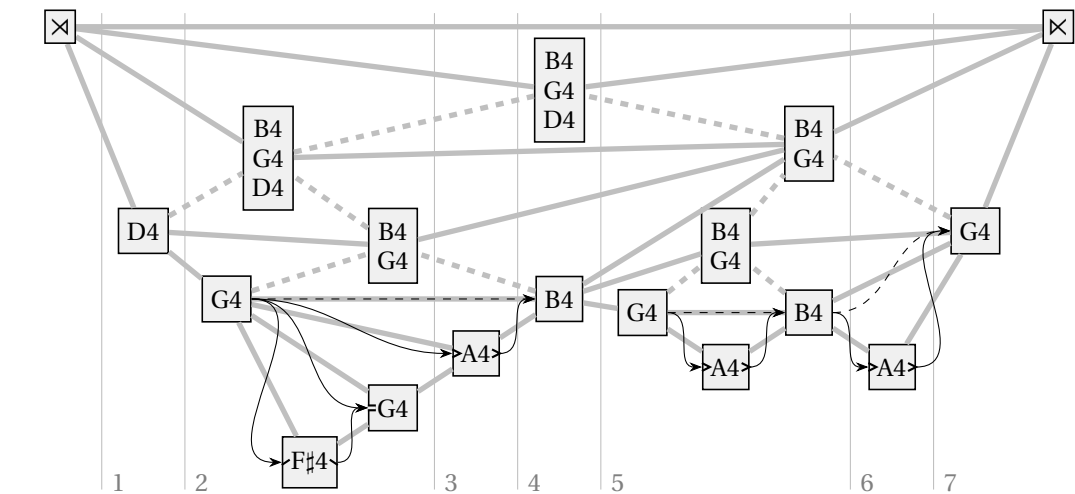
^aAdapted from *The Real Book – 1980 Totally Revised Edition*, The Real Book Press, Soysset, NY.

(b) The melody can be derived from two latent voices that move in parallel thirds. The repeated half notes are only derived once.

Figure 7.8 – The A part (mm. 1-8) of *Con Alma* (Dizzy Gillespie, 1954).



(a) The first phrase (m. 1-4) of *Hinunter ist der Sonne Schein* (Melchior Vulpius, 1609).

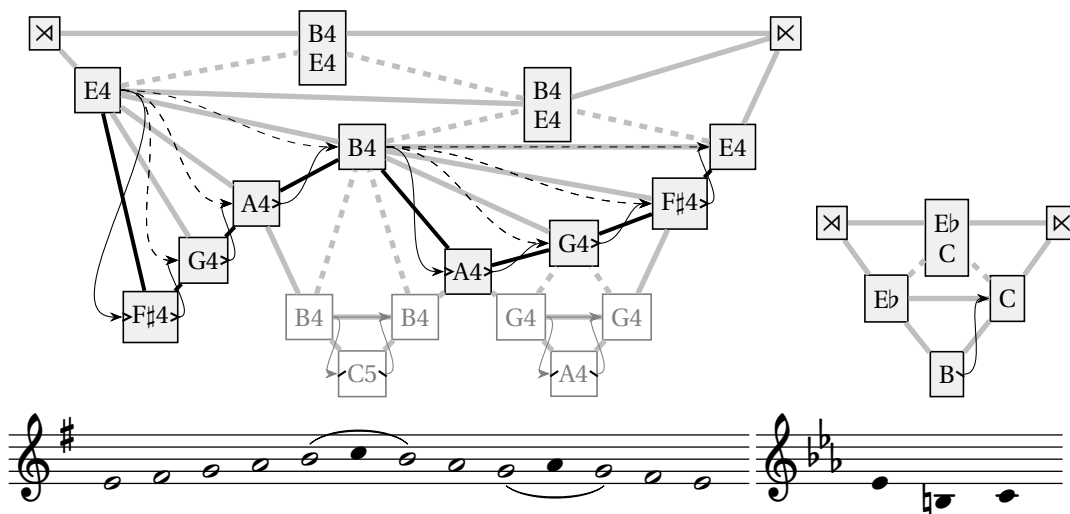


(b) The first part (mm. 1-7) of *Er weckt mich alle Morgen* (Rudolf Zöbele, 1941). Repeated notes have been merged and are only generated once in the derivation.

Figure 7.9 – Two examples of melodies with static latent structure.



(a) The main theme from *Vltava* (engl. *The Moldau*, Bedřich Smetana, 1874).



(b) Derivation of the melodic archetype and contour of the theme in (a). (c) A rare melodic interval.

Figure 7.10 – Two basic melodic phenomena arising from protovoices: (a,b) Basic melodic shapes are driven by a static structure of at least two reference points. (c) Rare melodic intervals arise through context switching between protovoices.

provides a frame of reference for the melody to unfold in.

Conversely, the protovoice perspective can also simplify some seemingly complex melodic phenomena. For example, rare intervals (such as the diminished fourth in Figure 7.10c, extensively featured, for example, in the 6th movement of Mahler's *Lied von der Erde*) can often be explained as the result of a single-sided elaboration ($B \rightarrow C$) on one side of a simpler interval ($E\flat - C$). The uncommon interval that emerges on the other side is a result of the context switch between two concurrent lines: the upper line containing the $E\flat$, and the lower line containing the progression $B \rightarrow C$.¹⁴

As the introductory example of the A minor Invention shows, latent polyphony affects all voices in a polyphonic piece. Moreover, these voices do not imply independent latent structures, but rather one underlying structure (which may contain connections across

¹⁴In this example, the two clashing protovoices are completely unrelated. There can, however, even be situations in which the two protovoices meet on the final note, e.g., in a double neighbor motion $G - A\flat - F\sharp - G$. It still holds that the notes forming the rare interval ($A\flat$ and $F\sharp$) are not part of the same protovoice (i.e., connected by an edge) and the melodic interval between them is incidental.

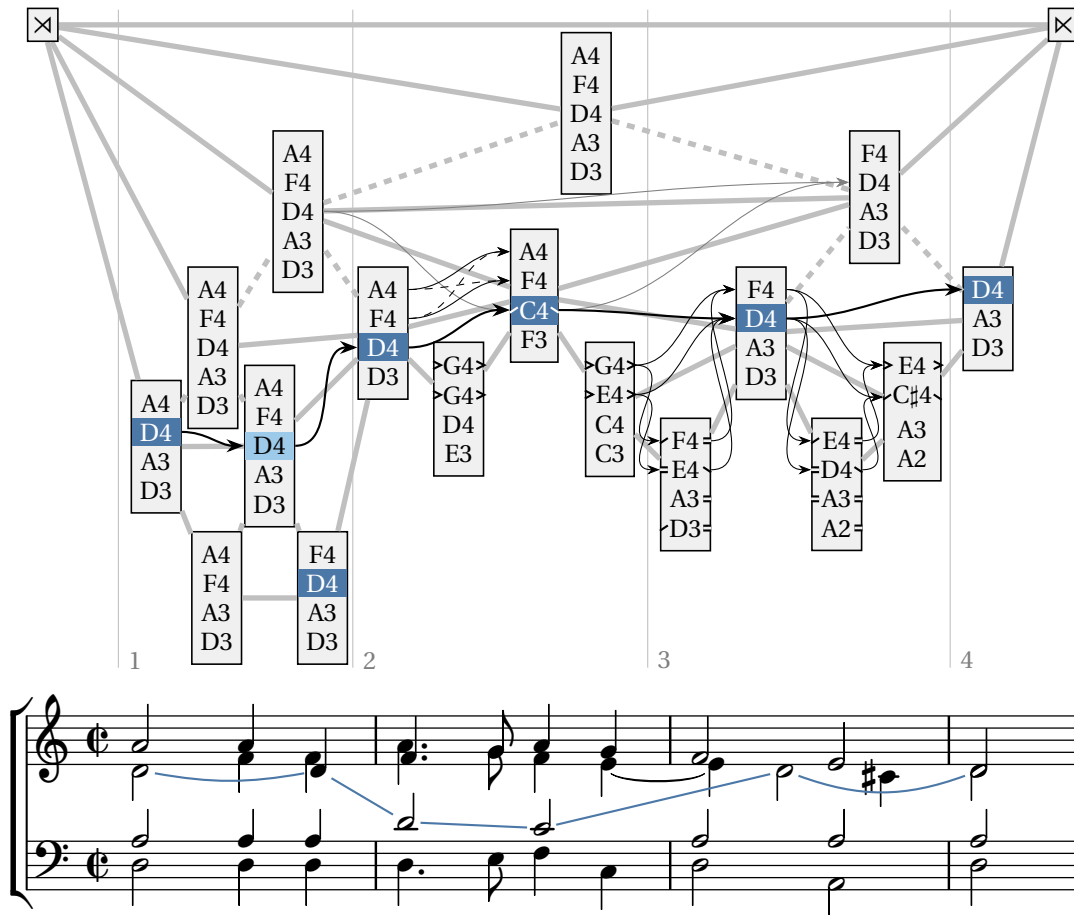


Figure 7.11 – The first phrase (mm. 1-4) of *Hinunter ist der Sonne Schein* in four parts.

voices) is jointly implied by the surface voices. An example of this phenomenon is the second G \sharp in the Invention (Figure 7.4a), which has one parent in the left hand (A3) and one in the right hand (A4). Another example is the four-part version of *Hinunter ist der Sonne Schein* as shown in Figure 7.11. In the beginning of the phrase, a D minor triad is realized with different voicings. In each of these voicings, the note D4 (blue) is sung by a different voice, first by the alto (1.1), then by the soprano (1.4) and finally by the tenor (2.1). The following neighbor motion D4-C4-D4 also involves two surface voices, tenor and alto, while on the final chord the single D4 is even shared by soprano and alto. Finally, the voice exchange between soprano and alto in measure 2 shows how the surface voices literally exchange their underlying protovoices. As these examples show, even in an already polyphonic setting the basic tonal relations between notes do not need to coincide with the surface voices.

7.6 Free Polyphony

As shown in the previous section, the protovoice model can be used to describe the internal structure of explicitly notated voices, both in monophonic melodies and in strict polyphony. Music written for keyboard instruments is often composed in a less rigid manner without a strict separation of notes into a fixed set of voices. Even without explicit surface voices, this music can exhibit horizontal relations as known from strict polyphony. However, due to the increased flexibility of the surface, these structures cannot be described in a framework based on a fixed set of concurrent streams anymore.

An example of a piece written in *free polyphony* is Schumann's *Träumerei* (Op. 15 No. 7), the beginning of which is shown in Figure 7.12. The surface structure of the piece combines aspects of both horizontal and vertical organization. On one hand, the number of notes that sound at the same time is flexible, ranging from 1 to 6, which hints at an underlying harmonic structure where each chord is realized with a varying number of surface notes. On the other hand, these chord tones are not sequentially isolated but are ornamented and connected by neighbor and passing notes (such as the E in measure 1 or the passing motions in measures 2 and 3) and form horizontal connections across chords (such as a descending line C-B \flat -A-G in the right hand in measures 2 and 3).

One of many possible derivations of the score is shown in Figure 7.12, together with a selection of slices and note relations from the derivation (shown above the graph). Surface ornamentations such as neighbor notes (e.g., E4 in m. 1 or D3 in m. 4) or passing motions (e.g., at the end of m. 2 and the end of m. 3) are directly represented using neighbor and passing notes embedded into the surrounding structure. In particular, the voice exchange in measure 3 (G4/B \flat 3 to B \flat 4/G3, marked with a * and highlighted in orange) is realized as a pair of passing motions between two representations of the same chord (C⁷), where one of the parent slices (here the left one) is interpreted as a modified copy (with G and B \flat flipped) of the other one, generated through a split operation. The descending line from C5 (beat 3.1) to G4 (beat 4.2) is reflected by a series of connections through the protovoice graph. Unlike the voice exchange, however, not all of these connections are generated on the same level of abstraction. In this particular analysis, the region between the B \flat -major chord in m. 2 and the C-major chord in m. 4 is interpreted as the prolongation of a six-four chord over C (†). The progression from C5-B \flat 4-A4 is a result of this prolongation: The six-four chord is spread (introducing a passing edge between C5 and A4) that generates the intermediate C⁷ chord on bb. 3.2-3.3, including the passing note B \flat 4 (highlighted in blue).

Other examples of free polyphony can be found in Bach's French Suites. Figure 7.13 shows the beginning of the Allemande from the D minor Suite (BWV 812), which, like

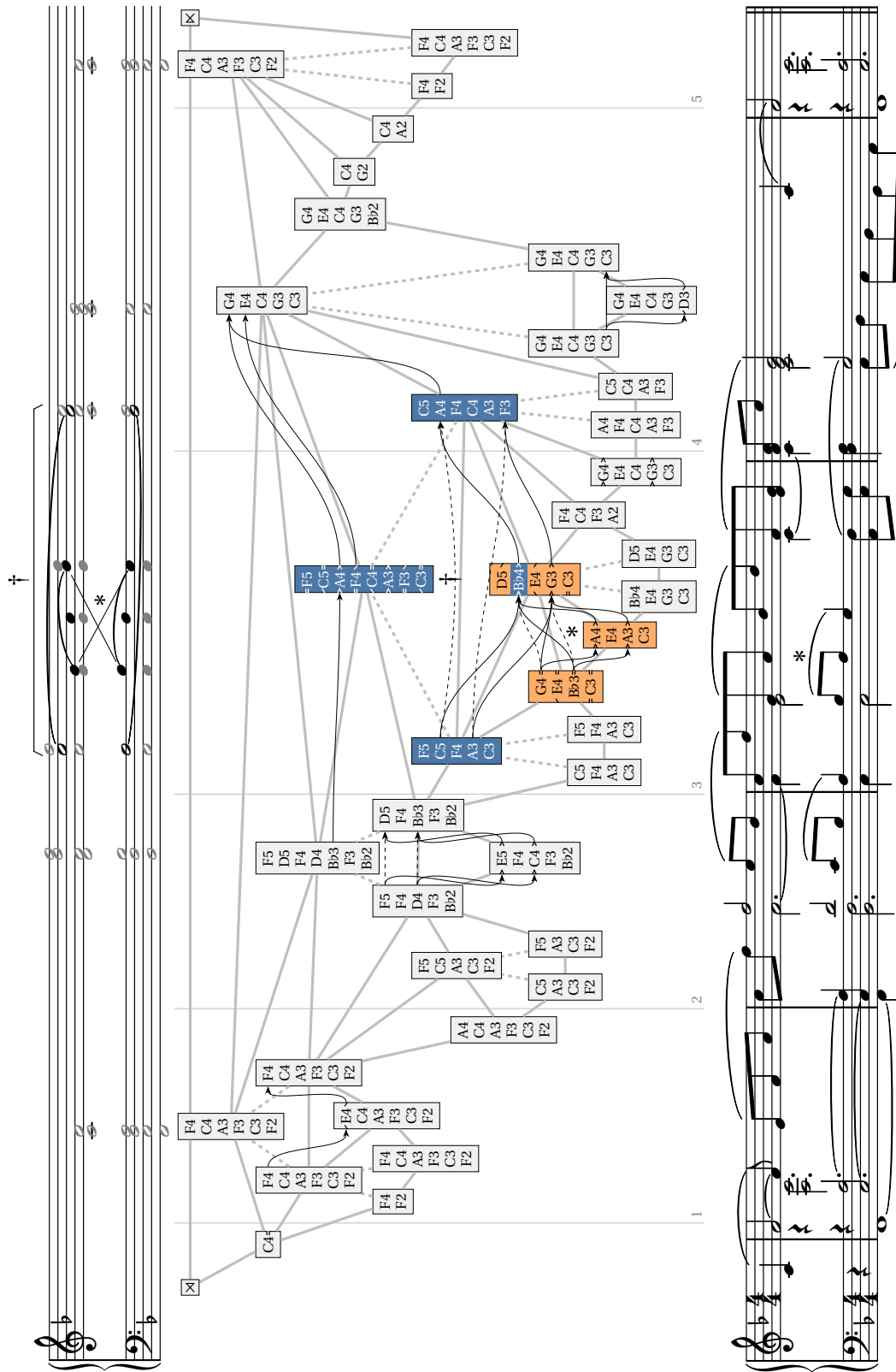


Figure 7.12 – Mm. 1-5 of *Träumerei* (Robert Schumann, Op. 15 No. 7) with minor simplifications.

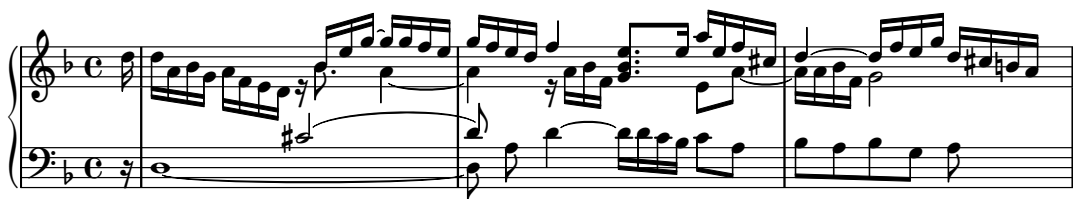


Figure 7.13 – Mm. 1-3 of the French Suite in D minor, I. Allemande (J. S. Bach, BWV 812). Analyses are shown in Figure 7.14 and Figure 7.17.

Träumerei, exhibits a complex surface realization of simple underlying structures. The beginning (up to b. 2.2) uses a variant of the same pattern that is used in the A minor Invention (Figure 7.1c and Figure 7.14), followed by a highly ornamented realization of a lamento (bb. 2.1 - 3.3, Figure 7.17). Both of these background patterns have a clear voice-leading structure, especially the lamento schema, which consists of three independent voices. On the surface, however, this voice structure is not directly visible anymore. Instead, fragments of voice-like structures jump between the underlying lines, ornamenting them and connecting them with passing notes. As in the previous example, the number of notes occurring at the same time varies constantly, making a partitioning of the score into a set of explicit voices impossible. The derivation of the surface from the underlying structure (Figures 7.14 and 7.17) shows how a protovoice network can express voice-leading relations between the notes without presupposing strict voices. This works much like in the case of latent polyphony, as discussed in Section 7.5: Ornamentation operations provide a function (and explanation) for every note and connect it to its context. At the same time, spreading and splitting can rearrange the given material and evoke new groupings of notes on the surface, such as the series of 16th-notes in the right hand. The only difference between latent polyphony in melodies and free polyphony is that the former produces a monophonic explicit surface voice while the latter does not.

As argued before, even strictly polyphonic pieces (such as Bach's two-part Inventions) have an underlying protovoice structure that is generally independent from the surface voices, as the surface voices themselves are implicitly polyphonic. The existence of free polyphony, which has the same kind of protovoice structure but does not have explicit voices, shows that voices are not fundamental for tonal structure. Rather than seeing free polyphony as an extension of strict polyphony, both (as well as melodies or "single-voice" pieces) should be understood as expressions of a more general common principle. In this sense, using strict voices is a special case, an additional restriction that composers follow out of convention or practical necessity, without direct relevance to the interpretability of tonal relations in a piece. Free polyphony, on the other hand, simply abandons these constraints.

While voices may not be fundamental entities, stylistic conventions related to voices

(such as melodic shapes, fifth movement in the bass, or counterpoint rules) as well as constraints of instruments and singing voice still shape compositions, even in free polyphony. Elaboration operations are usually not applied arbitrarily but may be chosen to generate surface melodies, bass lines, or explicit voices that satisfy these conventions. The direct relation between protovoices and surface voices, however, is rather flexible with surface voices being able to freely switch between concurrent protovoices, and protovoice connections going across surface voices.

From an analytical perspective, the surface voice leading is usually informative about the underlying structure. For example, the analysis of the voice exchange in *Träumerei* (Figure 7.12, *) follows the surface in that it places the A4 between G4 and B♭4, and the A3 between B♭3 and G3 rather than vice versa. This need not be because of an explicit coordination between surface voices and protovoices, but rather because an analysis that places passing notes in the same register as their respective parents provides a simple explanation of the surface than one that exchanges registers. Consequently, surface voices will often align with protovoices for stepwise movement but go across protovoices when larger intervals are involved.

7.7 Latent Entities

7.7.1 Harmonies

The previous sections have shown how the protovoice model can be used to capture direct relations between the surface notes. However, analytical accounts of music often use abstract musical entities such as harmonies, keys, schemata, lines, motivic or thematic material, or formal segments. These entities are usually not explicitly marked in the score (and much less in a recording), so just like the note-to-note relations they are *latent* and must be inferred by the reader or listener. Some types of latent entities (e.g., harmonies and schemata) have known prototypes that are shared between pieces, so the problem that the listener is faced with is to recognize the instantiations of these prototypes, which are often elaborated. Motives and themes, on the other hand, are usually not shared across pieces and therefore do not have a known prototype. Here, the problem is to discover the motive or theme as such.

What does it mean that a segment of music is an instance of a certain chord? A common approach to this problem is to assume some kind of resemblance between the chord prototype and its instances: A template is matched against some representation of the potential instance (e.g., by checking for the presence of certain notes), while allowing

deviation from the template (e.g., non-chord tones) to some extent.¹⁵ In this approach, the difference between prototype and instance is essentially treated as random noise that is not further explained.

There are two problems that come with this form of template matching: First, if the structural relations between the surface notes are not considered, different harmonic interpretations can become more difficult to distinguish. Consider, for example, the beginning of the Allemande in Figure 7.13. The notes in the first half of the first measure can be interpreted as a Dm chord that is elaborated with a number of ornaments: two neighbor notes B \flat and G, and a passing note E. However, the given notes would also match the template of a Gm⁷ chord, interpreting the notes D, B \flat , G, and F as chord tones and the two As and the E as non-chord tones. In both cases, three notes are considered to be non-chord tones. Taking into account the metrical strength of the assumed chord tones (which is almost identical in both cases) and the fact that the root of Dm is more common (and the bass note) than the root of Gm⁷ might give a slight advantage to the Dm interpretation, but it is not guaranteed that these disambiguating factors will always be in favor of the more plausible interpretation (see Chapter 3). However, once B \flat and G have been identified as neighbors of the two surrounding As, it is clear that Dm is the only possible harmonic interpretation of the segment (as opposed to Gm⁷).

The second problem arises when notes that disambiguate two possible prototypes are missing. This is also the case for the first chord of the Allemande: When only considering chord tones, it is impossible to know whether the chord is minor or major until the occurrence of the F. Even if the B \flat is taken into account, the chord remains ambiguous for the first few notes before the F. Contextually, however, already the single D5 on the upbeat must be interpreted as a part of the overarching Dm chord. Template-based approaches have to deal with this problem by picking reasonable segments to analyse and by taking the adjacent segments into account (e.g., Raphael and Stoddard 2004).

Protovoices offer an alternative account of harmonies as reductions of the surface. Figure 7.14 shows a derivation of the beginning of the Allemande up to beat 2.2. As described before, this analysis interprets the B \flat and G as neighbor notes to the As and the E as a passing note between F and D. Through a series of spreads, the whole first part of the measure is derived from a single slice containing three Ds, an F4 and an A4, i.e., a direct, un-elaborated instantiation of a D minor chord (marked with Dm₁). In a similar fashion, the second Dm chord in measure 2 (Dm₂) is realized by deriving an ornamented surface from a direct instance of the chord (marked in blue), and the region in between (orange) is derived from a slice that contains an A⁷ chord with a pedal. Thus, harmonies

¹⁵See, for example, Temperley 1997; Fujishima 1999; Raphael and Stoddard 2004; Sapp 2007; Oudre et al. 2009; Rhodes et al. 2009; Temperley 2009; Mearns 2013; Demirel et al. 2019.

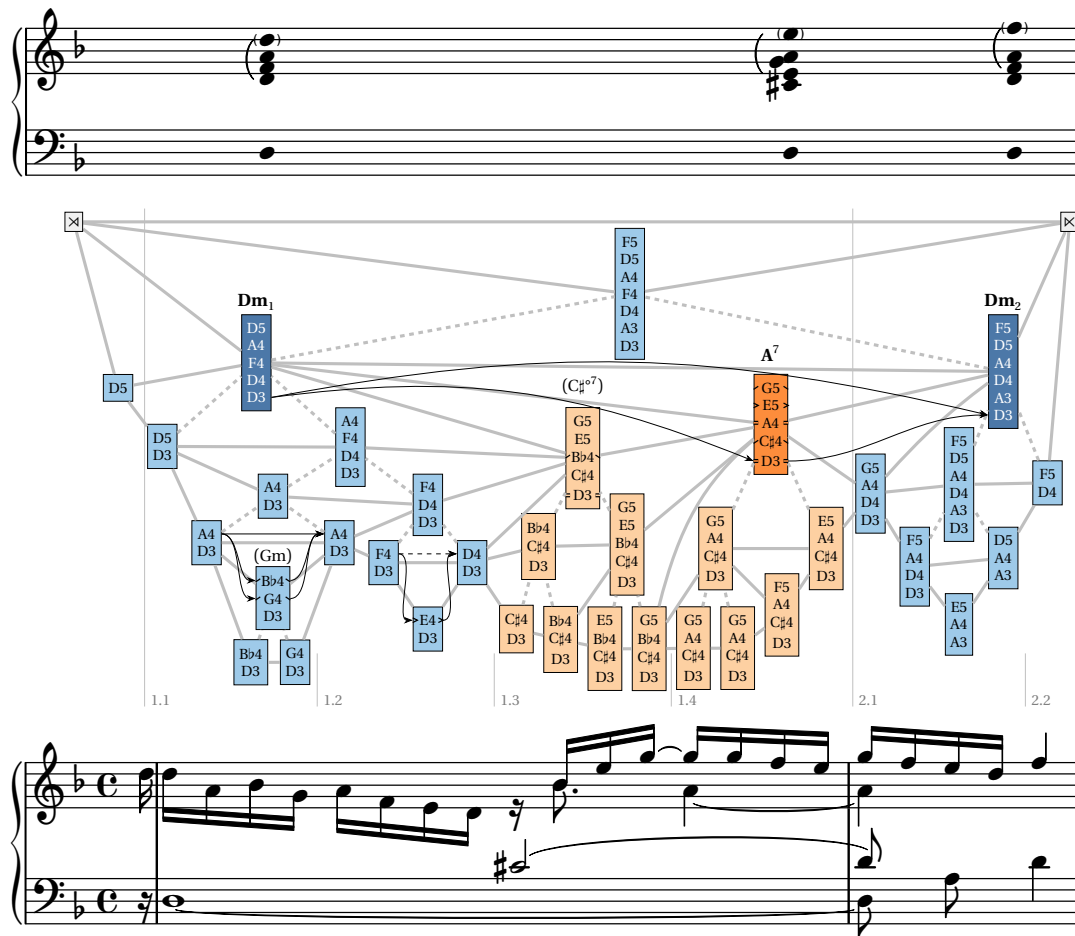


Figure 7.14 – Mm. 1-2 of the French Suite in D minor, I. Allemande (J. S. Bach, BWV 812).

as latent entities are directly represented by latent slices in a protovoice derivation, and saying that a segment of the surface realizes a certain harmony amounts to saying that there is a latent slice with the corresponding harmonic interpretation from which the surface segment is derived. This does not mean, however, that harmonies and slices are the same thing as there can be slices without harmonic interpretation. Harmony is a strictly higher-level concept than verticality and recognizing that a slice constitutes a harmonic event requires additional interpretative work. Thus, harmony is not reduced to protovoice relations, but it is realized and instantiated by them.

Conceiving harmonies as latent slices addresses both kinds of ambiguity that are problematic for template-matching approaches. Internal ambiguity is resolved because non-chord tones are explicitly *explained away* as ornaments. In the latent slice, chords can be directly recognized without having to assume any additional “noise”. Likewise,

latent slices can integrate the note material of several surface slices below them, so incomplete surface slices can be disambiguated if the missing notes are provided by the context. In addition, the derivation resolves a third type of ambiguity regarding the granularity of analysis. Instead of being a part of the surrounding D minor chord, the B \flat and G on beat 1.2 of the Allemande could be interpreted as an inserted passing Gm chord (together with the D in the bass) in a more fine-grained analysis. Similarly, beat 1.3 can either be interpreted as an independent C \sharp ^{o7} chord, or as part of the following A⁷ (with B \flat being another ornament). On an even higher level, the whole segment up to beat 1.2 could be heard as a long Dm chord where the apparent A⁷ chord tones are just treated as ornaments. The derivation in Figure 7.14 unifies all of these interpretations because it represents the segment on several levels of reduction. The A⁷ chord is inserted between two copies of an overarching Dm chord, so on the highest level, the whole piece is governed by D minor, while on the levels below it consists of a progression of three or even four chords. Even closer to the surface, the slice containing the neighbors B \flat and G (beat 1.2) can be interpreted as a Gm chord on its own.¹⁶

While the protovoice perspective addresses some of the problems of template matching, it does not solve the fundamental problem of ambiguity. A given musical surface generally affords several interpretations. Correspondingly, there can be (and generally are) multiple derivations of the same surface, each of which corresponds to a different interpretation of the tonal structure of a piece.¹⁷ This kind of ambiguity can only be resolved to a certain degree, for example by ranking the degree of plausibility of different derivations.¹⁸ There is, however, an important difference between the ambiguity of structural interpretations and the ambiguity that occurs in template matching. Structural interpretations are ambiguous because of a lack of information. Latent entities and structural relations between notes (as, e.g., intended by the composer) are not represented on the musical and (generally) cannot be recovered unambiguously from

¹⁶Not all ambiguities related to segmentation can be resolved like this, only hierarchical ones, i.e., those asking whether a segment is interpreted as a harmonic event on its own or as a non-harmonic elaboration of its surrounding harmonic context. If the point of transition between two chords is ambiguous, then usually the alternative cannot be represented in the same derivation but correspond to different derivations.

¹⁷Technically, the number of possible derivations is very large because every possible reduction of one part of a piece can be combined with (almost) every possible reduction of a different part of the piece. In the current form of the model, already the first half measure of the Allemande (including the upbeat) has 119,940 different derivations, most of which are either musically uninteresting or implausible, or closely related to a plausible analysis.

¹⁸This is often done by viewing a derivation as an instance of a probabilistic process, where each decision in the derivation has a certain probability to be chosen over its alternatives. The combined probability of all decisions is then interpreted as a measure of the derivation's plausibility. This principle is assumed to be underlying inference and perception in various cognitive domains (Abdallah, N. E. Gold, and Marsden 2016; Knill et al. 1996; Chater and Manning 2006; Chater, Tenenbaum, et al. 2006). A probabilistic version of the protovoice model is discussed in Chapter 9.

the surface. Template matching is additionally ambiguous because it assumes the relation between the prototype and the surface representation of a chord to be *noisy*, i.e., distorted by transformations and modifications (such as temporal displacement and ornamentation) that *cannot be explained* within the model. A protovoice derivation, on the other hand, provides a detailed explanation of the relationship between prototype and surface without assuming noisy modifications.¹⁹

In addition to revealing harmonic events in the form of latent chord slices, protovoice analyses can facilitate the harmonic interpretation of slices that do not directly instantiate chords. For example, the slice that corresponds to the A⁷ chord in Figure 7.14 contains an additional pedal D3. The derivation directly shows how this note is not just a non-chord tone but is derived from the surrounding Dm chord. The A⁷ slice thus can be interpreted as instantiating two harmonies: the D3 inherits its harmonic interpretation (as the root of Dm) from the parent slices by virtue of repeating the enclosing D3 while the remaining notes form the A⁷ chord as a separate harmonic entity. The A4, which is also repeated from both parents, remains harmonically ambiguous. Similar observations can be made in chords that contain suspensions. For example, measure 3 of *Hinunter ist der Sonne Schein* (Figure 7.11) contains two chords with suspensions, Dm with suspended 9 (beat 3.1) and A major with suspended 4 (beat 3.2), both resolving to their respective triads on the next sub-beat. The corresponding slices (the lowest slices in the right half of the graph) are both produced by a split between the unsuspected chords they connect, inheriting all notes from the resolved chord on the right except for the suspended tone that is held over from the left. A similar point can be made about the six-four chord, as it appears, for example, in *Träumerei* (Figure 7.12 †). The protovoice analysis is compatible with both interpretations of the six-four chord: Either the slice is considered an independent harmonic entity representing an F major chord in second inversion, or the slice is seen as a modification of the following C major chord with a 6-4 suspension (A and F) that resolve to the 5 and 3 (G and E) in the next slice while the C remains the root of the chord that is inherited from the right parent. A third alternative is to view the slice as a polyharmonic entity where the C is still the root of the following C major chord, but A and F belong to a separate F major chord, much like in the pedal example.

In all three examples the protovoice analysis itself does not provide the harmonic interpretation, but it provides the underlying structural entities and relations that harmonic interpretations may refer to and be built on. For example, slices can be interpreted as the entities that instantiate or realize abstract harmonic categories. Protovoice edges, on the other hand, provide the connections along which the harmonic interpretation of

¹⁹Despite the additional uncertainty they generate, template matching might still be cognitively relevant, e.g., to produce a first, heuristic estimate that may be refined later on by a structural interpretation.

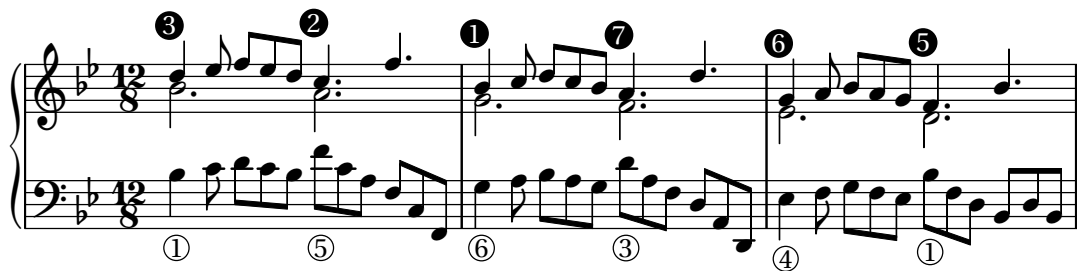
individual notes can be inherited (e.g., pedals or unsuspended notes), modified (suspensions), or explained away (non-harmonic ornaments). Thus, harmonic analysis has its own set of concepts (that are not part of the protovoice framework) but may use protovoice derivations to link these abstract concepts to concrete pieces.

7.7.2 Voice-Leading Schemata

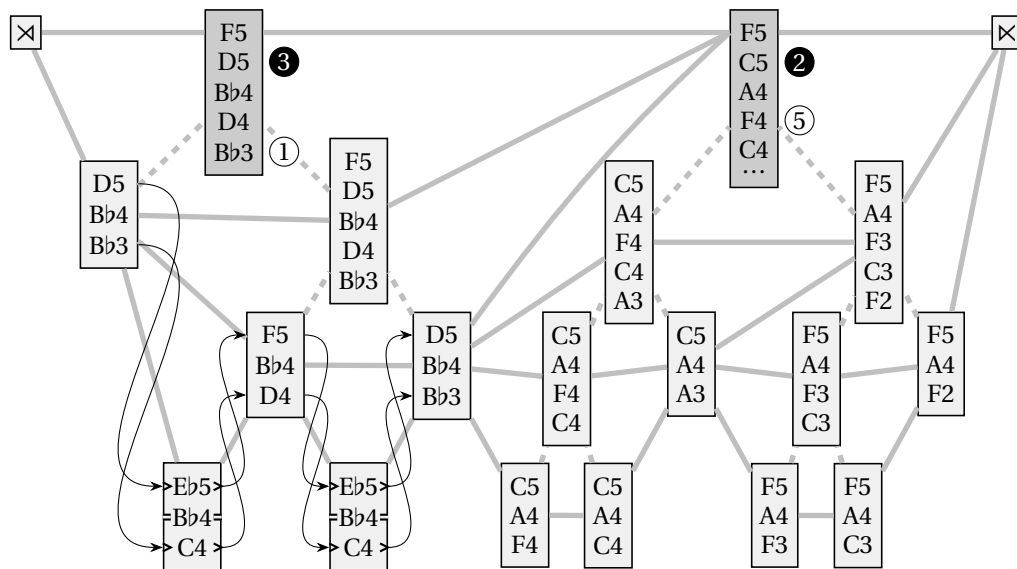
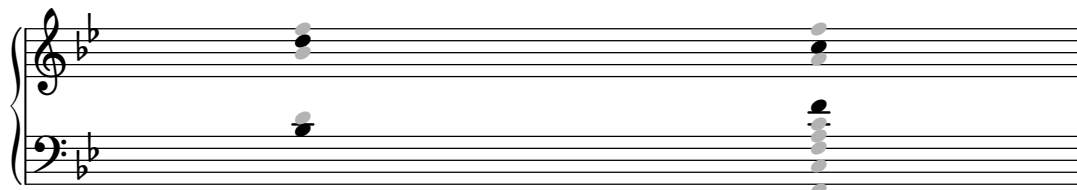
Another class of latent entities are voice-leading schemata (R. Gjerdingen 2007), recurring contrapuntal prototypes that usually consist of a fixed number of voices moving through a sequence of stages. As with harmonies, characterizing the relationship between prototypes and instances of schemata is difficult due to elaboration. Figure 7.15 shows an instance of the Romanesca schema, taken from R. Gjerdingen (2007) and a derivation of the first measure from the first two stages (the remaining measures are analogous). Just like the harmonic slices before, the stages of the schema emerge as latent slices that contain the schema notes as well as additional notes (as, for example, indicated by figured-bass annotations in the schema prototype) while the surface form is derived from these slices through horizontalization and ornamentation. In addition, the slices of the schema are directly connected by a sequence of transitions, i.e., there is a level of reduction on which the full schema is directly visible.

Figure 7.16 shows another surface instance of a schema, a Fonte (R. Gjerdingen 2007). Unlike in the previous example, the soprano and bass note of the first and third stage do not occur simultaneously, which makes this kind of schema instance very difficult to recognize for heuristic approaches that work with surface slices. In the derivation, however, the first two slices have been verticalized into a single latent slice, from which the first stage of the schema can directly be recognized. The example also shows how the schema notes need not coincide with metrically strong positions on the surface, as the soprano notes of the second and fourth stage are both delayed by an appoggiatura. Additionally, the derivation highlights the internal structure of the Fonte, which consists of two parts that both resolve a tritone into a third. In the derivation this tritone is generated as an elaboration of the third, so the stages 1 and 3 are subordinate to the stages 2 and 4, respectively. The two-part structure is thus directly visible at the top of the derivation, while the four-stage progression of the schema is visible at the level below.

An even more complex example of a schema instance can be seen when going back to the Bach's Allemande (Figure 7.13). Measures 2 and 3 realize a (non-chromatic) lamento bass or descending tetrachord that is harmonized with two additional voices according to the rule of the octave (e.g., IJzerman 2019) as shown in Figure 7.17. Because the order of the upper voices is flipped, suspensions are introduced in the middle voice to avoid



(a) The full schema instance as realized and marked by R. Gjerdingen (2007, p. 27).



(b) A derivation of the first two stages (m. 1).

Figure 7.15 – Gjerdingen's Example of a Romanesca from Handel's exercises for Princess Anne.

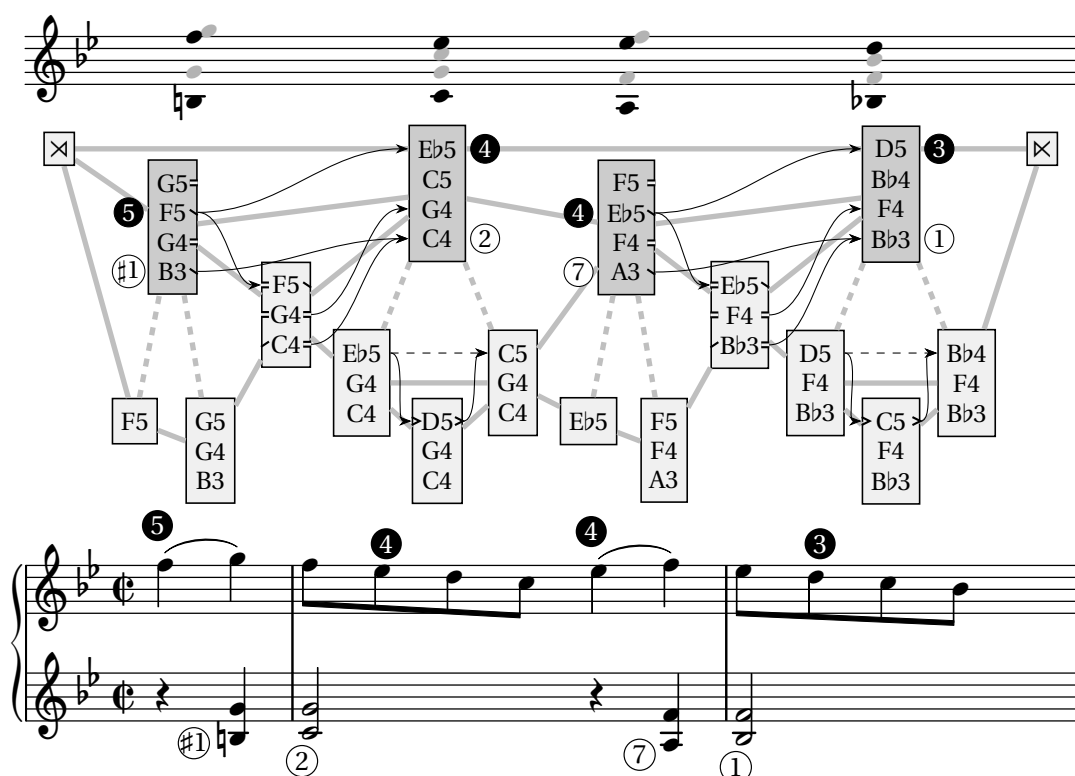


Figure 7.16 – A Fonte at the beginning of Piano Sonata No. 3 in B \flat major K. 281, III. (W. A. Mozart). Appoggiaturas have been written out for better alignment of score and graph.

parallel fifths. The analysis in Figure 7.17 shows how the highly ornamented surface can be derived from the schema prototype using the same techniques as discussed previously. In particular, the second suspension (which avoids the parallel fifths) is generated as usual by a split (*) that keeps the A from the left parent and takes D and B \flat from the right parent. The first suspensions (\dagger), on the other hand, is derived using two splits: Before the suspension slice itself can be generated, the B \flat needs to be prepared, which is achieved by another inserted slice (\ddagger) that becomes the left parent of the suspension slice. Note that Bach chooses to suspend the bass note (D $_4$) here in addition to the middle voice so that only the top voice moves at this point.

In general, we can see that schema prototypes occur in protovoice derivations as sequences of latent slices. The relationship between prototype and surface form is thus characterized by the elaborations that take place between the prototype and the surface. What is less clear is whether the prototype itself must be derived in a systematic way. One alternative is that the prototype is always produced by the same derivation or by one of a fixed number of variants. In this case, these fixed derivations would indicate the internal structure of the prototype (i.e., the relations and dependencies between the notes of

Figure 7.17 – Measures 2 and 3 of the French Suite in D minor, BWV 812, I. Allemande.

the prototype as they can be found, for example, in the Fonte in Figure 7.16), as well as possible dependencies on the context in which the schema occurs (i.e., constraints on what can happen right before the first or after the last stage, or in which kinds of situations the schema can be used). On the other hand, it is not guaranteed that all schemata have this fixed internal structure, as different instances of a schema could just be connected by the occurrence of the same pattern, i.e., by the pitch content on some level of reduction. In that case, any derivation that produces a sequence of slices containing the right notes is valid, which gives schemata a higher status as independent entities that do not “reduce” to a specific configuration of basic structure.²⁰

²⁰A more drastic step in this direction would be to deny that the latent schema slices are derived from the general protovoice formalism at all. Instead, inserting the schema prototype would be a structural primitive, a fundamental and atomic operation that constitutes an alternative to the protovoice operations. However, this extreme view would deny that the instantiations of schema prototypes in the latent slices have any internal structure or could be understood in terms of protovoice structure. Moreover, it would

A notable omission in the above characterization of schema instances as sequences of latent slices is the representation of voice structure in a schema prototype. As argued in Section 7.6, voices are not fundamental for tonal structure and (unlike slices) are therefore not directly represented by entities or relations in the protovoice model. While schema prototype as described by Gjerdingen usually contain explicit outer voices, the inner voices are represented by figured bass without concrete indication of the voice leading. That this internal structure can be more complex than a set of voices can be seen in the pattern that is underlying the beginning of Bach's A minor Invention and the Allemande from the D minor Suite (Figures 7.1 and 7.14). Both openings can be seen as a version of the Do-Re-Mi schema (R. Gjerdingen 2007) in minor. Its derivation in Figure 7.3 shows that its internal structure reflects the two outer voices described by Gjerdingen: The bass performs a neighbor motion from 1 to 7 and back to 1, while the soprano is implemented as a linear motion from 1 to 3. However, the pattern is not strictly separated into voices, as some of the lines can share notes in the first and third stage (and actually do so in the two pieces). This is at least consistent with Gjerdingen's prototype, which mentions the 3 in the third stage once in the melody and once in the figured bass and thus does not exclude that both 3s are realized by the same surface note. Explicit soprano and bass voices, on the other hand, cannot be directly represented in the protovoice model but, as argued before, may be left to a model of higher level concepts.²¹ As a consequence, deciding whether a potential schema instance respects the explicit voices of a schema prototype remains part of the additional interpretative work that comes with identifying protovoice structures as higher-level entities.

Two general points can be made about the relation between latent entities and protovoice derivations. First, latent entities do generally not emerge from protovoice analysis on its own. The plausibility of a protovoice analyses is not just determined by its ability to derive the pitch content of the surface slices (which is a simple binary criterion), but generally considers factors such as meter, phrasing, motivic aspects, or the plausibility of the latent structure that is revealed. Thus, analyses containing latent configurations that correspond to typical harmonies or schemata are generally more plausible than

place the rather high-level concept of a schemata on the same level of abstraction as the low-level relations of the protovoice model, which seems to be a mismatch of abstraction. Formally, the viewing schemata as structural primitives has no advantage over the less drastic view that the schema instances are freely generated since the protovoice formalism is likely flexible enough to generate schema instances everywhere a primitive schema operation could place them.

²¹As discussed in Section 7.6, explicit voices can but do not need to be aligned with protovoices. Both cases can be seen with schemata too: In the structure of the Do-Re-Mi instances, both bass and soprano have a corresponding path of protovoice connections. The structure of the Fonte instance in Figure 7.16, on the other hand, denies a connection in the bass from $\textcircled{\sharp 1}$ to $\textcircled{7}$. Instead, $\textcircled{\sharp 1}$ and $\textcircled{7}$ are subordinate to $\textcircled{2}$ and $\textcircled{1}$, respectively, which in turn are directly connected. The bass is thus still a connected structure, but not a single protovoice path.

analyses that use uncommon latent structures. Moreover, different assumptions about latent entities may lead to preference for different analyses: A listener familiar with 18th-century schemata might arrive at a different structural interpretation of a piece than a listener not familiar with these patterns.²² Thus, while the formal validity of a derivation is defined by the protovoice model itself, the preference for one valid derivation over another is partly determined by factors that refer to entities that are external to the model but related to the background of the listener (in a cognitive context) or the analytical goal (in an music-theoretical context). From a generative perspective, the higher-level concepts then become explicit parts of the generative process: Latent structures representing chords or schemata are generated explicitly as chord and schema instances, and potentially with a higher probability than non-harmonic or non-schematic structures.²³ Inferring a derivation then involves recovering these high-level decisions.

Second, the generative relation between latent and surface entities does not imply that the latent entities represents all the actually relevant aspects of a piece's structure while the surface only contains irrelevant or "ornamental" details.²⁴ On the contrary, a derivation captures the exact relationship between the latent structure and its surface realization, as well as the relations between the elements of the surface (even though these relations are themselves latent). Latent entities thus serve as reference points for understanding the surface, not as an excuse to ignore it. In principle, the derivation from a latent structure can even be part of the prototype of a higher-level entity. For example, some schemata exhibit typical ornaments in certain stages (R. Gjerdingen and Bourne 2015) so that derivation steps below the top-level configuration can be considered relevant for the characterization of the schema.

²²From the perspective of Bayesian cognition (Chater, Oaksford, et al. 2010), for example, those latent structures are preferred that most plausibly explain both the given piece and past musical experiences. However, the range of these experiences (which may include hearing pieces, performing pieces, learning and instrument, or learning theoretical concepts), usually differs between different listeners or analysts. Thus, while there can be objective criteria for choosing the latent entity types that explain a set of experiences best (i.e., with highest probability), taking a different set of these experiences (e.g., by including or excluding explicit teaching) will lead to different conclusions.

²³Formally speaking, the chordness of a slice is a latent variable that is determined during the generation of the slice and influences the probability of the slice's content and potentially even its children.

²⁴This concern is, for example, raised by R. Gjerdingen and Bourne (2015), regarding both reduction of a piece to long-term structure (Section 2) and the reduction of schemata to simple prototypes or lists of features rather than a collection of concrete exemplars (Section 6).

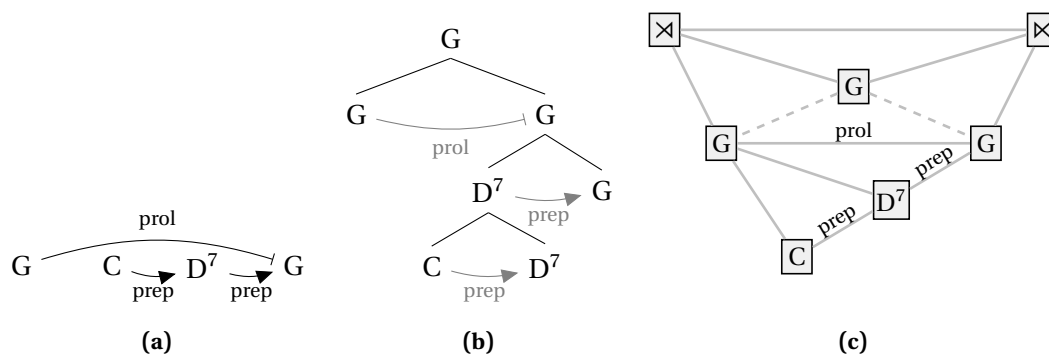


Figure 7.18 – The functional structure of the harmonic progression from the Cello Suite (Figure 7.5). The harmonic relations between the chords (a) can be expressed in a syntax tree (b), i.e., a derivation of the progression from a harmonic grammar. The same relations are reflected in a protovoice derivation (c) that derives a sequence of harmonic slices corresponding to the chords (e.g., as in Figure 7.5b).

7.8 Harmonic Syntax

Harmonic syntax describes the structural relations between chords in harmonic progressions. Like the protovoice model, harmonic relations are hierarchical and based on elaboration. For example, the harmonic grammar proposed by Rohrmeier (2020; 2011) is based on two fundamental harmonic relations: preparation and prolongation. Prolongation extends a harmony through a repetition or a functionally equivalent chord. Preparation elaborates a harmony by prepending an applied dominant (or an equivalent substitution) relative to the parent harmony. A complete harmonic progression is derived from a single overarching chord by repeatedly applying preparation or prolongation operations, just like the repeated application of protovoice operations generates a score. Since preparation and prolongation refer to a single parent, the resulting structure is a tree rather than a planar graph.

An example of such a tree is shown in Figure 7.18. The harmonic progression (taken from the Prelude of the Cello Suite, Figure 7.5) starts and ends with a G major chord (the tonic). The second tonic is prepared by a D^7 , which in turn is prepared by C major (Figure 7.18a). These relations are expressed in the syntactic derivation of the progression (Figure 7.18b): Starting with a G (representing the overarching tonic), the prolongation rule creates the initial and final G. The second G is elaborated through the preparation rule, while the resulting D^7 is itself elaborated by a preparation. Note that the syntax model does not assume a direct relation between the initial G and C. This reflects that the nested preparation of the final tonic can be continued up to an arbitrary point, so instead of a C, the first G could as well be followed by any chord that leads back to D^7 in a series of

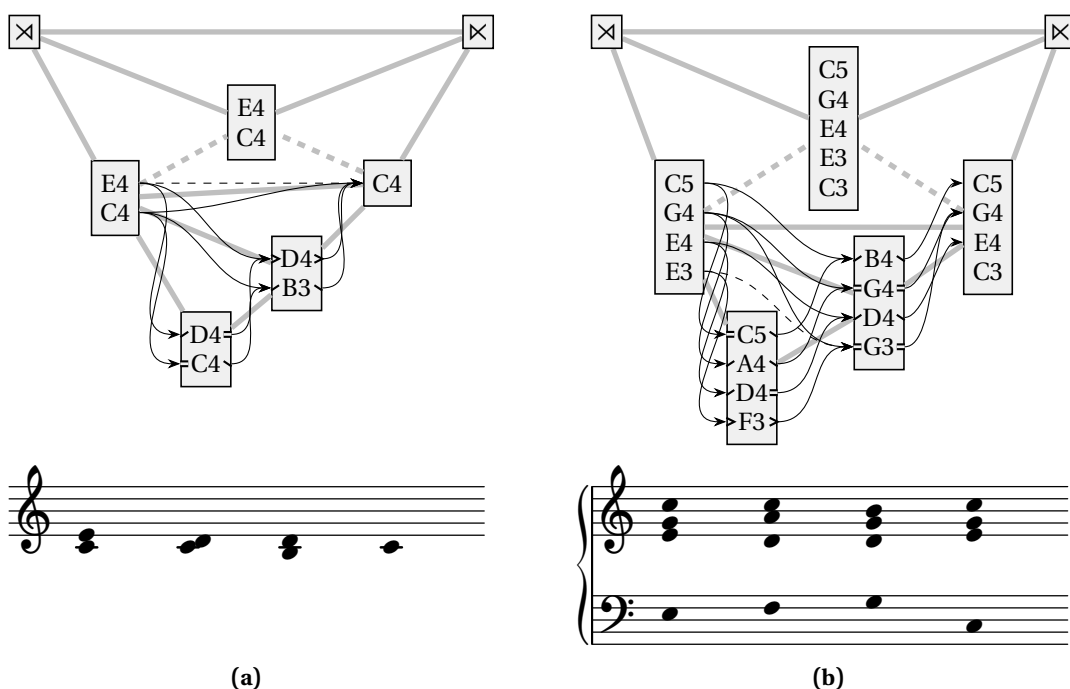


Figure 7.20 – Contrapuntal (a) and harmonic (b) progressions generated through elaboration.

G^7 C D^7 G). Functionally, the G^7 can either be explained as an applied dominant to C or as a “passing chord”, a non-functional extension of G with a passing 7. While the first interpretation (Figure 7.19a), cannot capture the voice-leading relation between G and G^7 , the “passing chord” interpretation ignores the preparatory character of G^7 . The protovoice derivation captures both relations (Figure 7.19b).²⁵ The split operation that generates the G^7 slice again represents the preparation relation between G^7 and C while the operations that generate the notes in the slice show how the G^7 chord is derived from the parent G. In particular, the F is explicitly marked as a passing note that connects the two parent chords.

Once a harmonic interpretation has been assigned to the elaboration steps of a deriva-

²⁵The notes in Figure 7.19b have been chosen to resemble the beginning of the Cello Suite Prelude and are derived analogous to Figure 7.5b. With the inserted G^7 and the pedal on G this pattern corresponds to a Quiescenza schema (R. Gjerdingen 2007). Generally, schemata can pose a challenge for functional harmony because they can produce harmonic progressions that do not follow the general syntax: the progression is then licensed through the schema instead of the grammar. One such example is the Pachelbel version of the Romanesca with the bass ① ⑤ ⑥ ③ ④ ① ④ ⑤. The harmonic progression IV-I-IV on the bass notes ④ ① ④ either must interpret the I as a preparation of the second IV (and thus not as a prolongation of the tonic), or the first IV as a plagal preparation of the I rather than a regular (authentic) preparation, which would be preferred.

tion, the note-generating operations and protovoice connections reveal the voice leading that takes place in a chord sequence. The protovoice model can thus shed light on how voice leading generates harmonic progression. Consider the simple cadential phrase shown in Figure 7.20a, consisting of a *clausula tenorizans* above a *clausula cantizans* with a suspension. From a contrapuntal perspective, the two voices start on an imperfect consonance, then create a dissonance by one voice moving while the other voice stays, resolve into another imperfect consonance, and finally close on a perfect consonance. The derivation of the progression reflects this left-to-right perspective in the protovoice paths that connect the surface slices, but it adds a top-down component to the story: The penultimate slice is generated as an ornamentation of the surrounding slices, the B as a neighbor to the two Cs, and the and D as a passing note between E and C. Similarly, the suspension is not created by *first* moving into a dissonance that is subsequently resolved, but rather by starting with the preparation (slice 1) and resolution (slice 3) and inserting the suspension as a hybrid between the two slices: one voice has already moved on (i.e., inherits the D from the resolution) while the other voice holds the previous note (inherits the C from the preparation). In a similar fashion, a full harmonic cadence can be derived (Figure 7.20b). The dominant chord is created by deviating from the chord tones of the tonic (especially the root) and repeating its fifth. Similarly, the predominant (IIIm⁷) is further elaborates the progression from the initial I to V with passing notes, neighbors, and repetitions from either side.

The top-down perspective explains why the arrival of the tonic is perceived as closure. The tonic is the reference point for the notes in the dominant slice, so moving from the penultimate to the ultimate slice means returning to this reference point. Note that in both versions of the cadence, there is no dissonance in the penultimate slice the resolution of which would evoke the feeling of closure. Instead, the dissonance is between the leading tone and the tonic that it wants to reach. The very notion of a leading tone is a top-down phenomenon, an *interpretation* of the functional role of a tone in its context as the lower neighbor of the overarching reference note.

It is important to note that this hierarchical perspective does not deny the left-to-right expectations evoked by the dissonance in a suspension or by a dominant. A listener does not necessarily need to hear the complete progression to form an interpretation. Instead, incomplete progressions can be partially interpreted, and expectations about the continuation of a progression are formed based on what is needed to complete the progression (Herff et al. 2021). Forming these expectations is generally not possible without some form of interpretation.²⁶ While in some cases there are direct local markers that indicate a specific continuation (such as the accented dissonance in a suspension or the tritone in a dominant-seventh chord), the expectation after configurations that

²⁶This is not to say that expectations are formed exclusively based on interpretation.

lack these specific markers (such as the minor third in Figure 7.20a and the G major triad in Figure 7.20b) only evoke these expectations because they are interpreted as dominants in context.²⁷ The specific markers thus do not directly evoke an expectation but rather suggest a certain interpretation which in turn evokes an expectation: The tritone in a dominant chord wants to resolve in a specific way not because this is an intrinsic property of tritones: Its inherent dissonance could be resolved in several ways, for example by resolving into a perfect fifth with one note moving by a semitone. The conventional resolution (both notes move by a semitone) is preferred because in a diatonic context tritones typically occur as scale degrees ⑦ and ④, and the simplest way to generate these notes is as ornaments of ① and ③.²⁸ While the specific markers can reduce uncertainty and thus strengthen expectations, the expectation itself is based on interpretation of the heard, and thus works even in the absence of specific markers.

7.9 Conclusion

The purpose of this chapter was to discuss the musical intuitions behind the protovoice model and to show how it addresses the phenomena that motivate it in practice. Through a series of examples, it was demonstrated how protovoice derivations capture the (generally hidden) tonal relations between surface notes in terms of simultaneity, sequentiality, and functionality, and how they relate the surface to concepts of common music-theoretical discourse. The protovoice model thus provides a precise formal language for expressing analytical intuitions within flexible yet interpretable and principled constraints. In addition, the model can support analytical judgments about higher-level entities such as harmonies or schemata by linking them explicitly to the surface notes. Since protovoice analyses are not bound to these high-level concepts, however, they are able to explain configurations in which these concepts break down, such as non-harmonic chords or free polyphony.

The protovoice model can be seen as an instantiation of the interpretation perspective outlined in Chapter 1. A derivation and the structural relations it entails provide a generative explanation of the observed surface notes using principles that are shared across many observations. In this chapter, however, the presented derivations have

²⁷A similar argument is made by Krumhansl (1990b) in response to Butler (1989) in the context of key recognition: The tritone might be a useful indicator when it is present, but key identification also works in its absence, so it cannot be the primary mechanism behind key recognition.

²⁸Expectations need not rely on the protovoice interpretation of the notes alone, but can involve higher-level structures such as schemata, harmonic implications, sequential progressions, or themes and motives. The general principle, however, remains the same: The expectation is formed by interpreting the heard notes as an incomplete expression of a latent structure that is to be completed.

been taken for granted, appealing to the musical knowledge of the reader or to the derivations' usefulness in demonstrating particular phenomena. However, in order to propose the protovoice model as a computational cognitive model of interpretation and understanding of (tonal) music, an important piece is missing: among the many possible derivations of a piece, what distinguishes plausible from implausible analyses? Bayesian perception has a principled answer to this question: Different execution paths of the generative process occur with different probabilities. The plausibility of an explanation is proportional to the overall probability of the derivation path. How this general principle can be applied to the generative process of the protovoice model will be discussed in Chapter 9.

8 An Annotation Tool for Protovoice Analyses

8.1 Introduction

Corpus and machine learning research on music relies on high-quality datasets that often involve annotations and analysis by experts. In creating these annotations, it is important to ensure formal consistency and machine readability, but also a high musical quality. While it might be possible to create simple types of annotations manually without support by dedicated tools, this quickly becomes inconvenient, if not infeasible, for complex analyses. Annotation workflows can then either rely on general music notation tools such as the Verovio Humdrum Viewer (Ricciardi 2020; Rodin and Sapp 2010) or MuseScore (Hentschel, Moss, et al. 2021), or work with dedicated tools for creating specific types of annotations (Giraud, Groult, et al. 2018; Tomašević et al. 2021; Harasim, Finkensiep, Ericson, et al. 2020). While dedicated tools come at a higher development effort, it can usually provide more convenience to the annotator and better ensure formal consistency.

This chapter presents a set of tools for working with protovoice analyses such as the ones shown in Chapter 7. In the protovoice formalism, the structure of a piece is described as a derivation, the execution trace of a generative process that produces the piece from a small number of operations. During this process, two types of generic relations are tracked: the horizontal connection of notes that belong to the same “protovoice”, and the vertical organisation of notes into “slices”, groups of notes that are considered simultaneous. At every point in the generation process, the current state of the piece is represented as a sequence of slices and transitions, where the slices contain notes and the transitions contain edges connecting these notes (Figure 8.1a).

A step in the derivation process has two phases: First, the temporal structure is expanded by either *splitting* a transition, creating a new slice between two existing slices, or *spreading* a slice into two slices. Figure 8.1b shows both operations in a *planar graph* notation similar to Yust (2006): The top row of slices and transitions shows the original state of a piece (or segment of a piece), while the elaboration produced by splitting or spreading is shown below it, similar to how trees are used to visualize derivations of context-free

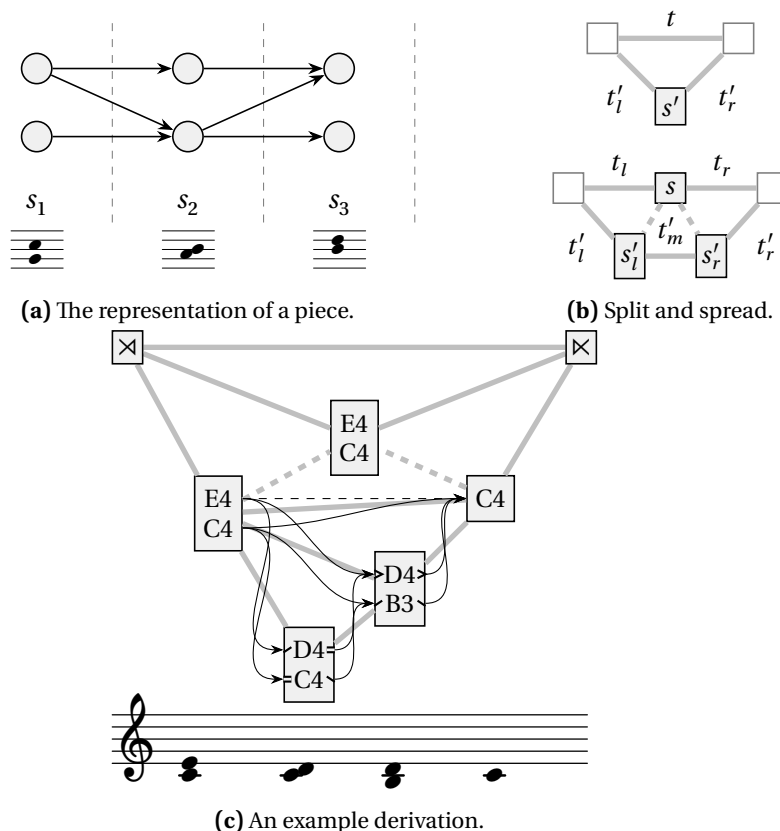


Figure 8.1 – In the protovoice formalism, a piece is represented as a sequence of slices (containing notes) and transitions (containing connections between notes). New slices and transitions are generated by splitting a transition or spreading a slice.

grammars.

After a *split*, the new slice is filled with notes that are generated as ornaments to the existing notes. Ornaments can be single-sided (neighbors or repetitions of a single parent note), double-sided (neighbors or repetitions with two parent notes, created along an existing edge), or passing notes (filling in a passing edge). Since repetitions and neighbors remain within step distance of their parents, regular protovoice edges always span step intervals. Longer intervals can be spanned by passing edges (shown as dashed lines), which must be filled in stepwise with passing notes before the derivation can terminate. After the new slice has been filled with notes, new passing edges can be created between notes from the child slice and either of the parent slices.

When a slice is *spread*, no new notes are created. Instead, the existing notes are distributed among the two child slices, moving each note either to the left or the right, or

repeating it on both sides. The transition between the child slices can either contain regular edges between repeated notes (for further elaboration in subsequent steps, or to tie the notes on the surface), as well as new passing edges.

As shown in Chapter 6, derivations can be represented as sequences of split and spread operations in left-most derivation order. A third operation (*freeze*) marks transitions as terminal, preventing them from further elaboration. Splits, spreads, freezes are then applied to the first non-frozen transition from the left (left/only split and freeze), the second non-frozen transition from the left, if present (right split), or both the first and the second transition (spread). The example derivation shown in Figure 8.1c can thus be described by the following sequence of operations, starting with a single transition between \times and \times that stands for the empty piece:

1. split the only transition (creating root notes E4 and C4)
2. spread both transitions and the slice between them)
3. freeze the left transition (and move on to next transition)
4. split the left transition (creating D4 and B3)
5. split the left transition (creating the suspension, D4 and C4)
6. freeze the left transition
7. freeze the left transition
8. freeze the left transition
9. freeze the only transition

The software presented in this paper provides tools for working with machine-readable analyses in the protovoice format. It consists of three components that will be described in Section 8.2: a web-based annotation interface for creating and editing protovoice analyses (Section 8.2.1), a viewer component to display analyses interactively on websites (Section 8.2.2), and a library that provides the underlying representations and operations and can be used by third-party tools (Section 8.2.3). The data format used to store analyses is described in Section 8.3.

Proto-Voice Annotation Tool

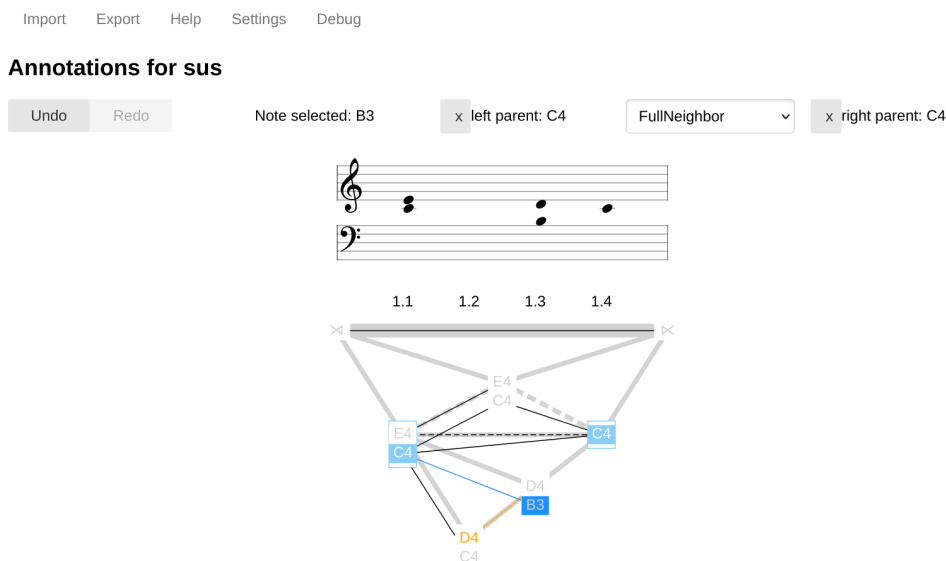


Figure 8.2 – Overview of the annotation interface.

8.2 Software Components

8.2.1 The Annotation Tool

The main purpose of the presented software is to create and edit analyses of pieces according to the protovoice formalism. The annotation component is a web-based tool¹ that facilitates the annotation process by guaranteeing a correct formal structure of the analysis and highlighting annotation inconsistencies (Figure 8.2). The tool is implemented in Purescript² using the Halogen framework³ and the VexFlow library⁴. It works entirely on the client side without requiring any server-side backend structure.⁵

The annotation workflow inverts the generative process described above. A user loads a complete piece (or a segment) they wish to analyse either in a special JSON-based notelist format or directly from MusicXML. The notelist format (described in Section 8.3)

¹<https://dcmlab.github.io/protovoice-annotation-tool/>. The source code is provided at <https://github.com/DCMLab/protovoice-annotation-tool>.

²<https://www.purescript.org/>

³<https://github.com/purescript-halogen/purescript-halogen>

⁴<https://vexflow.com/>

⁵An exception is the MusicXML-import functionality, which uses a server-side conversion service, but this is optional to the use of the tool, and the conversion functionality could be integrated into the client in the future.

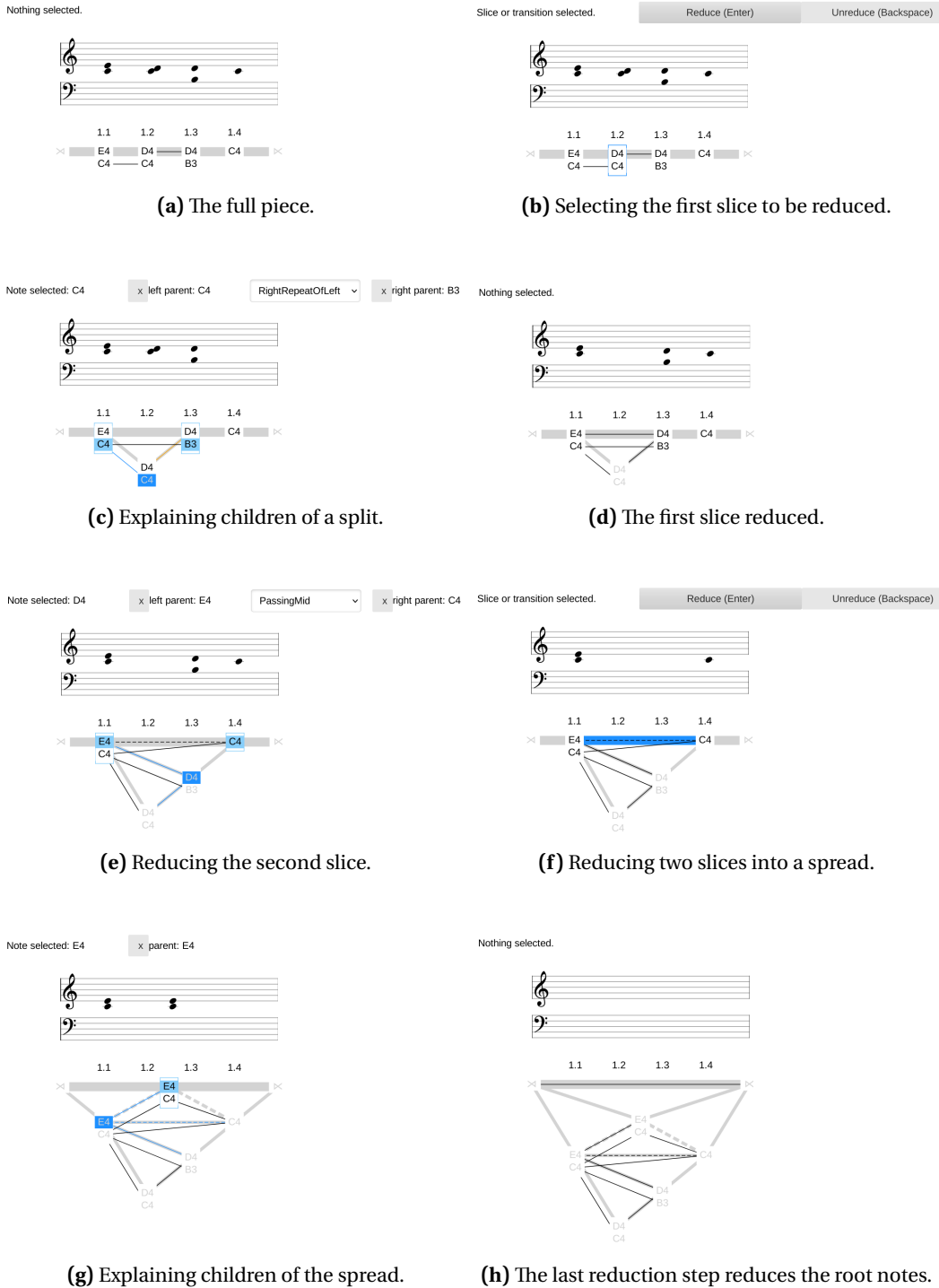


Figure 8.3 – An example of a reduction workflow in the annotation tool. To reduce the outer structure, an element of the current reduction surface (top of the graph) is selected. A slice can be reduced using a split (from (b) to (c)), a transition is reduced as the middle child of a spread (from (f) to (g)). The notes in reduced slices are then “explained” by connecting them to their parents. The score notation shows either the current reduction of the piece, or the currently selected slice and its parents.

can be created manually using a separate conversion tool.⁶ It is advisable to add `id` tags to the note elements in the source MusicXML file in order to be able to link the notes in the score and the elements of the analysis. These `id` tags will be respected both by the conversion tool and the direct MusicXML import of the annotation interface. When the piece is loaded, the annotator reduces the piece step by step until each note in the piece is explained. The resulting analysis can be exported to another JSON-based format,⁷ which can be loaded again later to edit the analysis. Saving an incomplete or inconsistent analysis is possible but will result in a warning. To prevent data loss, the current analysis is saved locally using the local storage API after each edit. This allows the user to recover an analysis after a crash or after accidentally closing the page.

In each step, they first select a temporal reduction (the inverse of split or spread), and then provide an “explanation” for each note, connecting it to its parents. The annotation tool will automatically create the correct parent structures and highlight unexplained notes, unused mandatory edges, and invalid reductions. An example of the annotation process is shown in Figure 8.3. After the piece is loaded (a), the full surface of the piece is shown as a chain of slices and transitions, with mandatory edges between tied notes (here the first C and the D, ties are not shown in the score). The user first selects the second slice (b) and reduces it, which creates a split operation with the slice as a child. The notes in this slice can now be reduced by connecting it to their parents. The C is reduced as a repetition of the C in the left parent slice, but as part of a line that continues to the B in the right parent slice (c). Selecting C and B as parents automatically sets the role of the child note to `RightRepeatOfLeft`. In a similar fashion, the D is reduced to its parents E and D (as a `LeftRepeatOfRight`), which “uses” the D-D edge and thus reverts it back from orange to black (d). The two notes of the child slice are shown in gray to indicate that they have been taken care of.

In the next step, the slice at beat 1.3 is similarly reduced as a split (e), explaining the B as a neighbor note and the D as a passing note between E and C, which creates a passing edge (dashed line). This passing edge is consumed by combining the two remaining slices (1.1 and 1.4) into a single slice using a spread operation (f). Here, the annotation tool is able to automatically select the parent of each note, since the assignment is unambiguous (g). This final “root slice” is reduced by another split into the empty edge between the beginning (⊗) and the end (⊗) of the piece, explaining the remaining notes as root notes (h).

The annotation tool includes support for generating graphics via LaTeX and TikZ⁸ such

⁶<https://github.com/DCMLab/musicxml-to-pvpiece>

⁷To make it easier to distinguish pieces and analyses, the file extensions `.piece.json` and `.analysis.json` are used, respectively.

⁸<https://github.com/pgf-tikz/pgf>

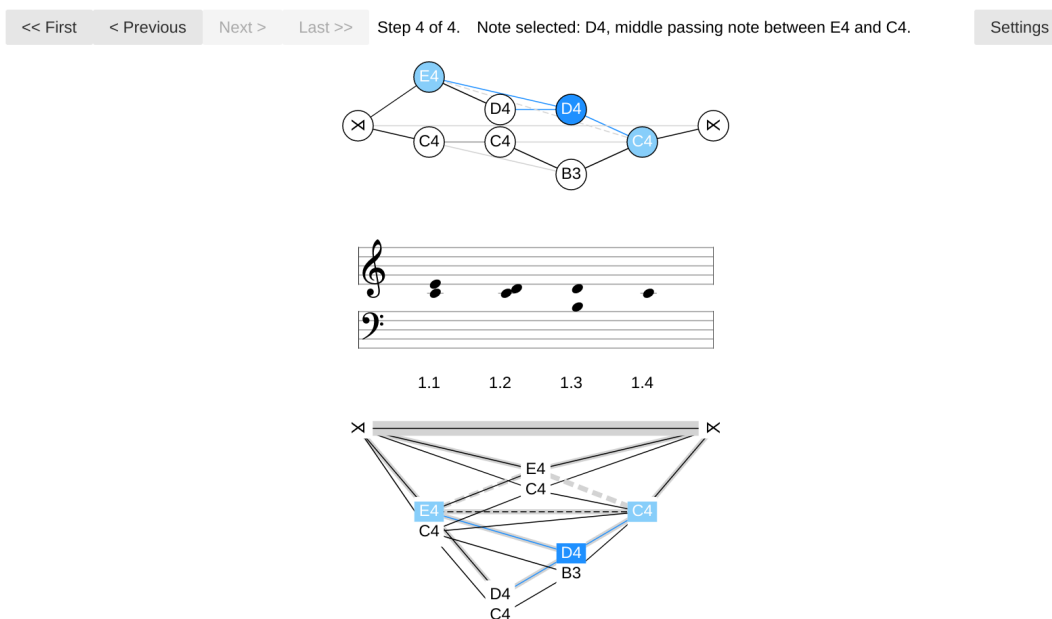


Figure 8.4 – The viewer component can also display the inner protovoice graph at the current step.

as the one shown in Figure 8.1c. The current analysis can be exported either as plain TikZ code that can be inserted into `tikzpicture` environment, or as a complete standalone document. Different types of objects (transitions, slices, notes, edges) are marked with corresponding TikZ styles that can be used to customize their general appearance. By default, markers that indicate the type of a note are not exported. However, for the currently selected note, a command for decorating this note with a type marker is provided separately, allowing the user to decorate individual notes.

8.2.2 The Viewer Widget

The viewer component provides an interactive frontend for displaying and exploring analysis. It is provided as a standalone JavaScript module that can be easily included into a webpage.⁹ The module has a single entry-point that creates a widget and binds it to a `div` in the current page. The shown analysis is provided in the analysis JSON format at creation time, e.g., by loading the analysis through the XHR or `fetch` API, or through a file upload. In addition, certain settings such as the zoom level or the shown derivation step can be pre-configured.

⁹An example page can be found at <https://musicology.epfl.ch/posters/protovoices-ismir21/>.

The widget shows the analysis in an interface that resembles the annotation tool. In addition, a graph of the internal connections between the notes at the current step can be shown. The user can step through the derivation in leftmost-derivation order and select notes to see their functional role and related notes. Since analyses of full pieces can become hard to read, the user can adjust the size of the visualization and enable or disable individual components.

8.2.3 The Internal Library

The common functionality of the annotation interface and the viewer widget is collected in a separate PureScript library. This ensures the compatibility of the two components and permits reuse of the functionality in third-party tools. Since PureScript compiles to JavaScript in a transparent and straightforward way, the library can either be used directly from JavaScript, from PureScript, or by creating a thin PureScript wrapper that exposes the relevant functionality to JavaScript.

The library covers three aspects of functionality: A set of types facilitate the representation of analyses and related data in a type-safe and consistent way. High-level editing and graph-walking operations allow the programmer to manipulate and process these data structures safely. Finally, conversion functions to and from JSON representations are implemented. The library integrates with the `purescript-pitches`¹⁰ library for representing pitches and checking the correct sizes of intervals.

The internal representation is not based on leftmost-derivations but rather on a tree structure of segments, pairs of a transition and the slice to its right. The details of an operation as well as its child segments are attached to the parent segment (to the left parent in case of a spread). This representation can be easily converted to and from a leftmost derivation but represents the graph structure of the analysis more directly and are easier to navigate. Transformation operations automatically maintain the integrity of this structure, reordering operations if necessary to preserve leftmost-derivation order.

8.3 Data Formats

The tools described in the previous section use two JSON-based formats to encode input pieces and analyses. The input piece is encoded as an array of slices, each of which contains a timestamp string (which usually has a `measure.beat.subbeat` structure)

¹⁰<https://github.com/DCMLab/purescript-pitches>

```

type PieceJSON = Array
  { time :: String
  , notes :: Array { pitch :: String, hold :: Boolean, id :: String }
  }

```

Listing 8.1 – The format of a `.piece.json` file expressed as a PureScript type.

as well as an array of notes. Notes have a pitch (a string using the notation of the `purescript-pitches` library and its siblings), an ID (optional) and a flag that indicates whether the note is continued in the next slice. 8.1 shows the schema of the format as a PureScript type, which directly translates to the JSON structure. A JSON encoding of the example in Figure 8.1c is shown in B.1

The schema of an analysis file is shown as the type `AnalysisJSON` in 8.2. Analyses are encoded as a combination of a starting structure (`start` and `topSegments`) and a list of derivation steps (`derivation`). Derivation steps (`LeftmostJSON`) are encoded as objects consisting of a tag (`"type": "<operation type>"`) that indicates the type of the operation as well as where it is applied, and a body (`"value": <operation>`) that contains the actual operation object. The operation types (`FreezeJSON`, `SplitJSON`, and `HorizJSON`) encode both the notes and edges they generate, as well objects that are left unexplained by the operation and meta information such as IDs. This ensures that the analysis can be saved and completely restored in any state, even when incomplete. The full specification of the format is shown in B.2, together with an example file (B.3) that encodes the analysis in Figure 8.1c.

8.4 Conclusion

Protovoice analyses are very detailed and thus complex, both in terms of their structure and in the consistency conditions they require. Even just displaying a complete analysis as a static two-dimensional image is problematic because of the three-dimensional character of a derivation (pitch, time, and generation order). Analysing scores and exploring analyses thus is not feasible without specialized tools that enforce consistency and allow the user to navigate a derivation in a meaningful way. Creating such a tool is always a balance between such specialization and integration into existing and familiar interfaces (e.g., music notation software or score-based annotation tools). Because of the very specific requirements of protovoice analyses, the tools presented here opt for a custom user interface that is specialized to the structure of the protovoice model and thus provides more guidance to the user. The separately usable library that is underlying

```
type AnalysisJSON =
  { derivation :: Array LeftmostJSON
  , start :: SliceJSON
  , topSegments ::
      Array
        { trans :: TransitionJSON
        , rslice :: SliceJSON
        }
  }
...

type LeftmostJSON = Variant
  ( freezeLeft :: FreezeJSON
  , freezeOnly :: FreezeJSON
  , splitLeft :: SplitJSON
  , splitRight :: SplitJSON
  , splitOnly :: SplitJSON
  , hori :: HoriJSON
  )

type ChildrenJSON = Array { child :: Note, orn :: Maybe String }
type SplitJSON =
  { regular :: Array { parent :: Edge, children :: ChildrenJSON }
  , passing :: Array { parent :: Edge, children :: ChildrenJSON }
  , fromLeft :: Array { parent :: Note, children :: ChildrenJSON }
  , fromRight :: Array { parent :: Note, children :: ChildrenJSON }
  , unexplained :: Array Note
  , keepLeft :: Array Edge
  , keepRight :: Array Edge
  , passLeft :: Array Edge
  , passRight :: Array Edge
  , ids :: { left :: TransId, slice :: SliceId, right :: TransId }
  }
...
```

Listing 8.2 – Parts of the format of a `.analysis.json` file expressed as PureScript types. The full specification is shown in Listing B.2.

the different tools, however, makes it possible to integrate the functionality of these tools into more general annotation or notation software in the future.

Ensuring consistent and unambiguous encodings is particularly relevant when the annotations are not just created for visualization purposes (as in Chapter 7) but as datasets for computational models. Chapter 9 discusses how generative process of the protovoice model can be described as a probabilistic program that makes random decisions based on a learned understanding of what constitutes a plausible derivation. A set of analysis created using the annotation tool is used to train a proof-of-concept version of such a probabilistic model, bridging the gap between the music-theoretical aspects shown in Chapter 7 and computational models of cognition.

9 Bayesian Modeling of Protovoice Structure

9.1 Introduction

The previous chapters have introduced the protovoice model, shown how pieces can be parsed into derivations, and how these can encode an analytical understanding of the tonal relations in a piece. In Chapter 6, however, we have also seen that the space of possible derivations for a given piece is usually extremely large and contains plausible analyses as well as implausible analyses. This chapter addresses the issue of judging the relative plausibility of different analyses for a given piece.

A principled way of reasoning about plausibility is offered by Bayesian probability theory (Jaynes 2003; MacKay 2003), as described in Chapter 2. In this framework, the plausibility of an analysis a given the surface of a piece s is expressed as the conditional probability

$$p(a | s), \tag{9.1}$$

which in turn is derived from a joint distribution

$$p(a, s, \dots) \tag{9.2}$$

according to the operations of probability theory, in particular marginalization and conditioning.¹ The joint distribution represents the listener's beliefs about what makes an analysis more or less plausible. The main question to be answered is thus what this joint distribution looks like, and why.

Bayesian probability serves as the foundation of Bayesian cognitive modeling (Chater, Oaksford, et al. 2010), which uses Bayesian models to describe cognitive capacities such as reasoning and decision making (Jaynes 1988; Griffiths and Tenenbaum 2006; Oaksford and Chater 2007; Pearl 2009), perception (Knill et al. 1996; Kersten et al. 2004; Chater and Manning 2006), and learning (Jacobs and Kruschke 2011; Gopnik and Bonawitz 2015;

¹The probabilistic notation in this paper corresponds to the slightly simplified notation that is sometimes used in machine learning: p does not directly refer to the probability of an event (which would be problematic for continuous random variables) but to the probability mass function (for discrete variables) or the probability density function (for continuous variables).

Ullman and Tenenbaum 2020b; N. D. Goodman, Ullman, et al. 2011). Bayesian perception and learning, in particular, are based on the idea that the brain, when confronted with a new perceptual input, tries to reason to the best explanation of this observation, adjusting its beliefs about the world if necessary. The plausibility of an explanation z for an observation x is quantified as

$$p(z | x) = \frac{p(z, x)}{p(x)} = \frac{p(x | z)p(z)}{p(x)}, \quad (9.3)$$

i.e., it depends on the prior probability of the explanation $p(z)$ (independent from the observation), as well as the likelihood $p(x | z)$, the probability that the explanation (if true) would give rise to the observation.² This conditional distribution corresponds exactly to the conditional in Equation 9.1: If an interpretation of a piece is understood as an explanation of the piece's surface, then the plausibility of a certain interpretation of a piece depends on its prior plausibility $p(a)$, as well as the likelihood of the piece given the interpretation $p(a | s)$.

The remainder of this chapter proposes and discusses a Bayesian version of the protovoice model. Section 9.2 gives an overview of the relationship between generative modeling and probabilistic programs, which serves as the foundation of the overall approach. Section 9.3 presents a simple proof-of-concept of a probabilistic program that generates protovoice derivations. Section 9.4 evaluates this proof-of-concept and demonstrates how it can be employed for (supervised) learning and inference. Finally, Section 9.5 gives an outlook of possible extensions of the model and more advanced inference methods.

9.2 Generative Models and Probabilistic Programs

A generative model is a model that describes how observations are produced through a (generally unobserved) generative process. For example, the image of an object on the retina can be understood as the result of a physical process that involves light being reflected by the object, falling into the eye, and being projected to the retina by the lens of the eye. The relation between observation and process is usually ambiguous: A sentence can be seen as a linguistic expression of a certain intention, but the same intention can be expressed in different ways, and the same sentence can be the product of different intentions. In addition, just like every model, this process can be a simplification, abstraction, or approximation: The height of a person can be modeled as the

²The third term, $p(x)$, is a constant for a fixed observation and can be thus be ignored when comparing two explanations of the same observation.

random result of a normal distribution with a certain mean and variance, although the true biological process is more complicated than that.

Descriptions of generative processes typically involve several steps as well as intermediate variables, which, if unobserved, are called *latent variables*. A simple process generates the observed outcome directly in a single step, for example:

1. Pick a set of lottery numbers.

A slightly more complex process could involve first selecting a person that then picks the lottery numbers according to their preferences:

1. Pick person A or person B.
2. If person A, pick numbers according to A's preferences,
If person B, pick numbers according to B's preferences.

If only the resulting numbers are observed, the person that picked them is a latent variable. If the preferences of each person are known, observing the selected numbers lets us draw conclusions about the person that picked them. Alternatively, the preferences of each person can be considered part of the model and are thus chosen throughout the process:

1. Pick preferences for A.
2. Pick preferences for B.
3. Pick A or B.
4. Let the chosen person pick the numbers.

Usually, when dealing with datasets, some variables are fixed across the dataset (*global*) while other variables are specific to a single datapoint (*local*). We could modify the lottery process to produce several sets of numbers, each chosen by a different person (*local*), but with fixed preferences for each person (*global*):

1. Pick preferences for A.
2. Pick preferences for B.
3. Repeat N times:
 1. Pick A or B.
 2. Let the chosen person pick the numbers.

While these kinds of process structures are very typical, (resembling, for example, the chord model used in Chapter 4), the generative process can have a much more complex structure, involving recursion or dependencies between the datapoints. The distinction between local and global variables does thus not always apply.

Generative processes usually involve some form of *random decisions*, which can reflect either actual randomness (e.g., physical randomness) or uncertainty about the underlying decision process.³ These random decisions can be modeled using conditional probability distributions that characterize the probability of each outcome given on all previous decisions in the execution of the process. Making the decision can then be modeled as drawing a sample from this conditional distribution. The lottery process from above could be described more precisely as such a probabilistic process:

- 1: Draw the preferences for A: $\vec{\theta}_A \sim \text{Dirichlet}(\vec{\alpha}_A)$.
- 2: Draw the preferences for B: $\vec{\theta}_B \sim \text{Dirichlet}(\vec{\alpha}_B)$.
- 3: **for** i from 1 to N **do**
- 4: Draw the person: $P_i \sim \text{Bernoulli}(0.5)$ as A (1) or B (0).
- 5: $n_i \leftarrow \emptyset$.
- 6: **for** j from 1 to M **do**
- 7: Choose a number: $n \sim \text{Categorical}(\text{zeroAndNormalize}(\vec{\theta}_{P_i}, n_i))$.
- 8: $n_i \leftarrow n_i \cup n$.

In this example, the function *zeroAndNormalize* returns a modified preference vector ($\vec{\theta}$) where the preference of the already drawn numbers (n_i) is set to 0 and the remaining preferences are renormalized. Probabilistic decisions (e.g., the next number) can thus be mixed with deterministic computations (updating the preferences).

The description of such a process with random decisions is called a *probabilistic program* (N. D. Goodman, Mansinghka, et al. 2008; van de Meent et al. 2018). As discussed in Chapter 2, probabilistic programs correspond to a factorization of a joint probability distribution into the conditional distributions that correspond to the steps of the program:

$$\begin{aligned}
 p(\theta_A, \theta_B, \vec{P}, \vec{n}) &= p(\theta_A) \\
 &\quad \cdot p(\theta_B | \theta_A) \\
 &\quad \cdot p(P_1 | \theta_A, \theta_B) \\
 &\quad \cdot p(n_1 | \theta_A, \theta_B, P_1) \dots
 \end{aligned} \tag{9.4}$$

Each of these conditionals has in principle access to all previously sampled variables (e.g., n_i uses P_i , and θ_A or θ_B), but they can also ignore some or all of the previous variables (e.g., $p(\theta_B | \theta_A) = p(\theta_B)$).

Probabilistic programs are not just a convenient representation of complex joint distributions, they also provide ways to query this distribution. Sampling from p is equivalent

³A generative process might be deterministic, i.e., it might not contain decisions, or all decisions are predetermined. Deterministic processes are not relevant to the current discussion.

to running the program and sampling each random variable according to its conditional distribution. The probability of a full sample (where all variables are observed) is computed by running the program with each variable assigned its observed value, recording for each variable the local probability of the observed value, and taking the product of these probabilities, according to Equation 9.4. Conditioning on variables in the beginning of the program (e.g., $p(\vec{P}, \vec{n} \mid \theta_A, \theta_B)$) amounts to fixing these variables to specific values. Conditioning on later variables (e.g., $p(\theta_A, \theta_B \mid \vec{P}, \vec{n})$) involves the application of more complex inference algorithms, which can nevertheless use the program representation of the distribution, such as sampling methods (Hastings 1970; S. Geman and D. Geman 1984; Hoffman and Gelman 2014) or variational inference (Blei et al. 2017; Ranganath et al. 2014; Kucukelbir et al. 2017). Most modern probabilistic programming languages use both types of inference or combinations thereof (N. D. Goodman and Stuhlmüller 2014; Wood et al. 2014; Carpenter et al. 2017; Cusumano-Towner et al. 2019; Bingham et al. 2019). Because of their strong ties with generative models, the probabilistic programming paradigm is a natural fit for generative Bayesian models of cognition and perception (N. D. Goodman, Tenenbaum, et al. 2016; Kulkarni et al. 2015).

Besides the general relationship between generative models and probabilistic programming, there is a more practical reason for using the latter. Previous generative models of musical structure have generally used existing classes of probabilistic models such as Markov chains (Conklin and Witten 1995; Ponsford et al. 1999; Pearce 2005; Moss, Neuwirth, Harasim, et al. 2019), hidden Markov models (HMM; Allan and Williams 2004; Temperley 2007a; C. W. White and Quinn 2018; Duane 2019), or context-free grammars (PCFG Gilbert and Conklin 2007; Abdallah, N. E. Gold, and Marsden 2016; Groves 2016; Harasim, Rohrmeier, et al. 2018; Rohrmeier 2020b). For these classes, specialized algorithms for inference, parameter estimation, and prediction are known, such as the forward-backward algorithm, the Baum-Welch algorithm, or the inside-outside algorithm (Manning and Schütze 1999). Markov chains, HMMs, and PCFGs can all be expressed as probabilistic programs (Algorithm 9.1). The generative process of the protovoice model, on the other hand, cannot be expressed in terms of such a standard model, but it can be described as a probabilistic program and thus be treated with the inference methods known for probabilistic programs. From the perspective of probabilistic programming, the protovoice model is thus much “larger” and more detailed than the programs of HMMs or PCFGs, but not fundamentally more complex. The classical models enforce strong independence assumptions between the probabilistic decisions in the process, which usually can only see a limited context (such as the current state, or the leftmost non-terminal). Probabilistic programs have generally much less strong restrictions about their dependency structure, which in principle allows each generation step to consider all previously generated structure. This flexibility comes at the cost of not being able to perform exact inference anymore. It may therefore be necessary to

Algorithm 9.1 Markov chains, HMMs, and PCFGs expressed as probabilistic programs. The parameters of each model (e.g., transition probabilities) are expressed as latent variables that are drawn from some prior distribution.

(a) A first-order Markov chain.

```
1:  $\theta \sim \text{some prior}$  ▷ sample transition probabilities
2:  $s_0 \leftarrow \text{start-state}$  ▷ initialize state
3:  $i \leftarrow 0$ 
4: while  $s_i \neq \text{end-state}$  do
5:    $s_{i+1} \sim p(s_{i+1} \mid \theta, s_i)$  ▷ sample next state
6:    $i \leftarrow i + 1$ .
```

(b) A HMM process.

```
1:  $\theta \sim \text{some prior}$  ▷ sample transition probabilities
2:  $\eta \sim \text{some prior}$  ▷ sample emission probabilities
3:  $s_0 \leftarrow \text{start-state}$  ▷ initialize hidden state
4:  $i \leftarrow 0$ 
5: while  $s_i \neq \text{end-state}$  do
6:    $o_i \sim p(o_i \mid \eta, s_i)$  ▷ sample emitted symbol
7:    $s_{i+1} \sim p(s_{i+1} \mid \theta, s_i)$  ▷ sample next state
8:    $i \leftarrow i + 1$ 
```

(c) A PCFG process.

```
1:  $\theta \sim \text{some prior}$  ▷ sample rule probabilities
2:  $s \leftarrow [\text{start-symbol}]$  ▷ initialize string
3: while  $s$  contains a non-terminal symbol do
4:    $L \leftarrow \text{leftmost non-terminal in } s$ 
5:    $r \sim p(r \mid L, \theta)$  ▷ choose rule with LHS  $L$ 
6:   apply  $r$  to  $L$  in  $s$ 
```

restrict some of this flexibility again to be able to apply efficient inference methods.

9.3 A Proof-of-Concept Model

This section presents a simple probabilistic version of the protovoice model. It follows the general scheme of the protovoice generation process: The leftmost one or two open transitions are elaborated using splits, spreads, or freezes. Once an operation type is chosen, its details are decided in a sequence of steps that is specific to that operation. These sub-programs can be shared between different contexts, e.g., the same routine

for generating a split is used for splitting the left transition, the right transition, or the final transition. Similarly, left neighbors, right neighbors, and full neighbors are all generated by the same process. This sharing is not necessary, as the parameters or even the procedures might be different in the different cases, but it simplifies the program and is sufficient for a proof-of-concept model. An overview of the program structure is given in Figure 9.1. Some parts of this model are outlined in Algorithm 9.2 and Algorithm 9.3. The full pseudocode of the model is provided in Section C.2, and a Haskell implementation of the full model is available online.⁴

The complete model describes the joint distribution

$$p(\vec{\theta}, \vec{D}) = p(\vec{\theta}) \cdot p(\vec{D} | \vec{\theta}) \quad (9.5)$$

over global parameters $\vec{\theta}$ and derivations \vec{D} (Algorithm 9.2a). In this proof-of-concept version of the model, all decisions within a derivation are made locally and independent from their context as far as possible. This way, the global parameters that influence these decisions can be inferred directly from the observed outcomes, without having to account for the context in which the decisions were made (see Section 9.4.1). In particular, each random decision is only based on one corresponding global parameter that controls its probability. This parameter is drawn from a prior distribution that is conjugate to the distribution it is used by. For example, the parameter $\theta_{\text{freezeSingle}}$ is a real number between 0 and 1 that controls the probability to freeze the last open edge in the piece (and thus terminate the derivation). The decision whether to freeze is modeled by a Bernoulli distribution (Algorithm 9.2c, line 5), so $\theta_{\text{freezeSingle}}$ is drawn from a Beta distribution, which is conjugate to the Bernoulli distributions. Choosing conjugate priors makes it easy to infer the posterior distribution of the parameters when learning from a dataset of analyses (MacKay 2003). The set of all parameters θ is chosen once in the beginning of the derivation process (Algorithm 9.2a, line 2) and remains constant across all pieces. A list of all global parameters and their prior distributions is given in Table C.1 in Appendix C. All of these priors correspond to a uniform distribution over the domain of the respective variable, although other priors within the same respective distribution family are possible.

Throughout the generation process, it is important that random decisions are only made if their result is needed. For example, the set of legal outer structure operations (spread, split, freeze) depends on the number of open transitions, so the function `SAMPLESTEP` (Algorithm 9.2b) will first check the number of open transitions and then branch on whether no open transition is left (derivation is complete), one transition is open (allowing only split and freeze), or more than one transition is open. Similarly, the function

⁴<https://github.com/DCMLab/protovoices-haskell/blob/main/src/PVGrammar/Prob/Simple.hs>

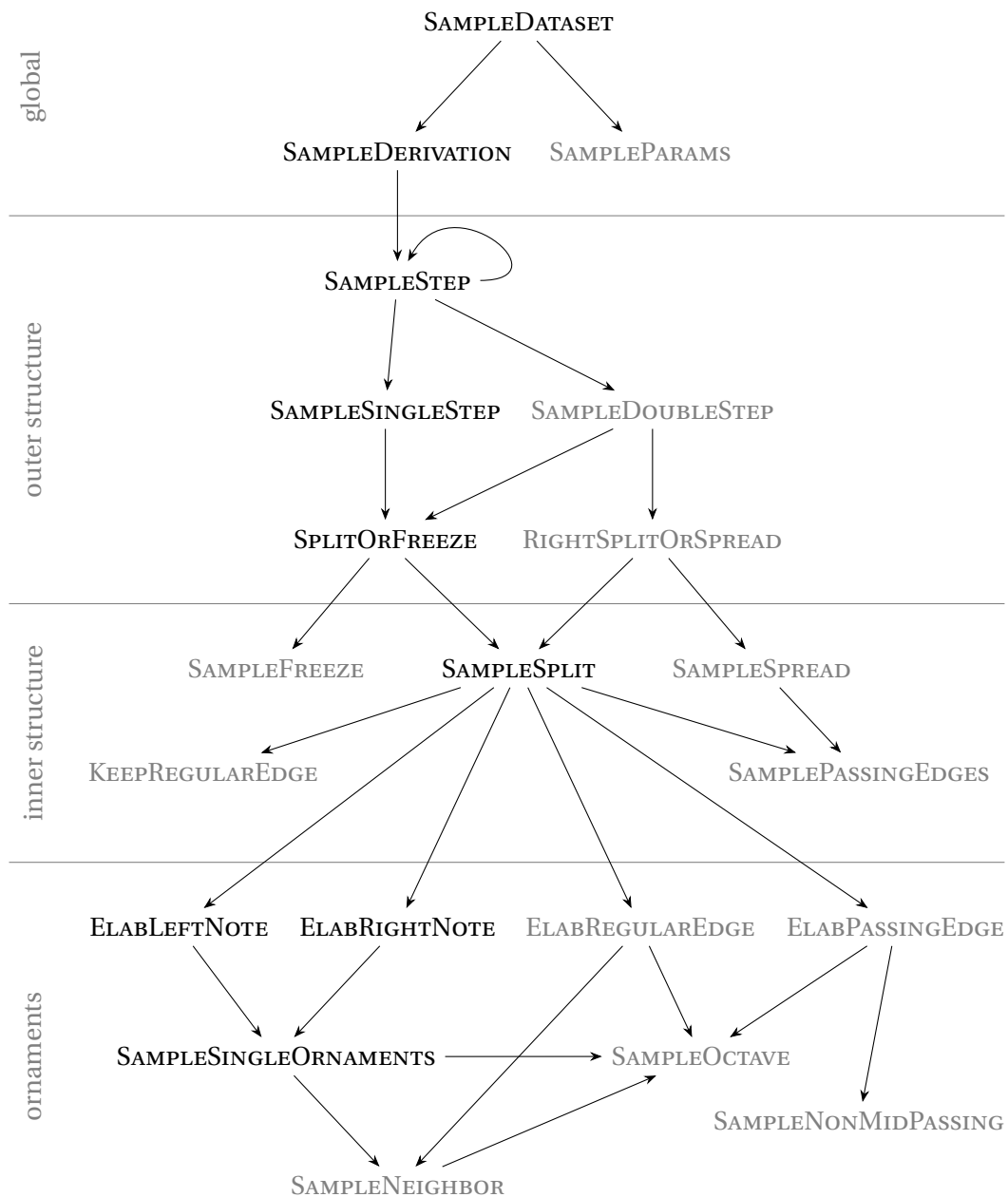


Figure 9.1 – An overview of the structure of the model program. Each node represents a function (subroutine) in the model and is connected to the functions that it calls. Black functions are shown in Algorithms 9.2 and 9.3. The full model is shown in Section C.2.

Algorithm 9.2 Generating global parameters and outer structure.

(a) Generating the full dataset.

```

1: function SAMPLEDATASET()
2:   global  $\vec{\theta} \leftarrow$  SAMPLEPARAMS()
3:   for  $i \in [1, N]$  do
4:      $D_i \leftarrow$  SAMPLEDERIVATION
5:   return  $\vec{D}$ 

```

(b) Sampling one derivation.

```

1: function SAMPLEDERIVATION()
2:   -- start with the empty piece and derivation
3:   return SAMPLESTEP( $\times - \times$ , [], false)

4: function SAMPLESTEP(surface, deriv, afterRightSplit)
5:   -- check number of open transitions
6:   if 0 open transitions in surface then
7:     return deriv
8:   else if 1 open transition then
9:      $op \leftarrow$  SAMPLESINGLESTEP(surface)
10:  else
11:     $op \leftarrow$  SAMPLEDOUBLESTEP(surface, afterRightSplit)
12:     $surface' \leftarrow$  applyLeftmost( $op$ , surface)
13:    return SAMPLESTEP( $surface'$ ,  $deriv + op$ , isRightSplit( $op$ ))

```

(c) Elaborating a single open transitions.

```

1: function SAMPLESINGLESTEP(surface)
2:   return makeSingleOp(SPLITORFREEZE(last(surface), freezeSingle))

3: function SPLITORFREEZE(transition, param)
4:   if transition is freezable then
5:      $shouldFreeze \sim$  Bernoulli( $\theta_{param}$ )
6:     if  $shouldFreeze$  then
7:       return SAMPLEFREEZE(transition)
8:     else return SAMPLESPLIT(transition)
9:   else return SAMPLESPLIT(transition)

```

SPLITORFREEZE first checks whether the current transition can be frozen, i.e., whether it contains only repetition edges (Algorithm 9.2c, l. 4). If the transition is freezable, a coin is flipped to determine whether to freeze or to split (l. 5). If it is not freezable, it is always split (l. 9). This design ensures that all random decisions can be reconstructed from an annotated derivation, which is essential for inference and learning.

There are cases in which it is not possible to restore the exact order of operations from the derivation. For example, while the order in which the parent edges and notes are elaborated can be fixed by some arbitrary ordering (e.g., Algorithm 9.3a, ll. 3-6), the children of the *same* parent could have been generated in an arbitrary order. These cases can be treated similar to the order independence in the multinomial distribution: During generation, the children are generated in an arbitrary order, which might be normalized later in the process (e.g., by sorting the children). When evaluating the probability of a derivation, a separate term for the number of possible permutations is included. During parameter inference, this permutation term is ignored because it is an irrelevant multiplicative constant. In order to make these optimizations, permutation-independent parts of the program are marked. In the pseudocode, this is represented by a **permute** block, which repeats its code a given number of times (like a **for** loop), collects the generated values (marked with **yield**), and returns them in some canonical order, irrespective of the order in which they were generated. An example can be found in SAMPLESINGLEORNAMENTS (Algorithm 9.3b, l. 9). SAMPLESINGLEORNAMENTS also shows how some decisions can use constant probabilities instead of global parameters, when estimating these probabilities from the data is not of interest (l. 15).

The particular parametrisation of the model shown in this section is generally too simplistic for analytical purposes. We cannot expect to distinguish plausible or implausible analyses only based on simple preferences such as repetitions over neighbor notes, or chromatic over non-chromatic ornaments. Keeping decisions local and independent primarily serves a simple inference scheme, which will be used in Section 9.4 to evaluate the proof-of-concept model. Despite these limitations, even this simple model demonstrates how the plausibility of a particular derivation can be linked to global stylistic properties via a probabilistic generative process. More elaborate and musically meaningful versions of such a process are discussed in Section 9.5.

Algorithm 9.3 Generating inner structure.

(a) Sampling a split operation.

```

1: function SAMPLESPLIT( $\theta$ , surface, transition)
2:   -- sample children:
3:   childrenReg  $\leftarrow$  [ELABREGULAREDGE(edge) for edge  $\in$  regularEdges(transition)]
4:   childrenPass  $\leftarrow$  [ELABPASSINGEDGE(edge) for edge  $\in$  passingEdges(transition)]
5:   childrenL  $\leftarrow$  [ELABLEFTNOTE(note) for note  $\in$  leftSlice(transition)]
6:   childrenR  $\leftarrow$  [ELABRIGHTNOTE(note) for note  $\in$  rightSlice(transition)]
7:   -- collect generated objects:
8:   middleSlice  $\leftarrow$  makeSlice(childrenReg,childrenPass,childrenL,childrenR)
9:   (edgesL, edgesR)  $\leftarrow$  makeEdges(childrenReg,childrenPass,childrenL,childrenR)
10:  -- generate passing edges and drop/keep regular edges:
11:  passL  $\leftarrow$  SAMPLEPASSINGEDGES(left(transition), middleSlice)
12:  passR  $\leftarrow$  SAMPLEPASSINGEDGES(middleSlice, right(transition))
13:  keepL  $\leftarrow$  [KEEPREGULAREDGE(edge) for edge  $\in$  edgesL]
14:  keepR  $\leftarrow$  [KEEPREGULAREDGE(edge) for edge  $\in$  edgesR]
15:  return makeSplit(childrenReg, childrenPass, ...)

```

(b) Sampling child notes from a parent on the left or right.

```

1: function ELABLEFTNOTE(parent)
2:   children  $\leftarrow$  SAMPLESINGLEORNAMENTS(parent, elaborateL)
3:   return (parent, [makeLeftOrnament(child) for child  $\in$  children])

4: function ELABRIGHTNOTE(parent)
5:   children  $\leftarrow$  SAMPLESINGLEORNAMENTS(parent, elaborateR)
6:   return (parent, [makeRightOrnament(child) for child  $\in$  children])

7: function SAMPLESINGLEORNAMENTS(parent, paramElaborate)
8:    $n \sim \text{Geometric}_0(\theta_{\text{paramElaborate}})$ 
9:   children  $\leftarrow$  permute  $n$  do
10:    repeat  $\sim \text{Bernoulli}(\theta_{\text{repeatOverNeighbor}})$ 
11:    if repeat then
12:      oct  $\leftarrow$  SAMPLEOCTAVE()
13:      yield makeRepeat(parent + oct)
14:    else
15:      up  $\sim \text{Bernoulli}(0.5)$ 
16:      child  $\leftarrow$  SAMPLENEIGHBOR(up,parent)
17:      yield makeNeighbor(child)
18:  return children

```

9.4 Evaluating the Proof-of-Concept Model

9.4.1 Parameter Inference

As shown in Section 9.3, all decisions in the probabilistic process are based on local conditional distributions of the form $p(o_i | \theta_a)$, where o_i denotes the outcome of decision i based on the parameter θ_a (the index a is used to mark a specific parameter θ_a and the corresponding decisions that use this parameter, while $\vec{\theta}$ denotes the collection of all parameters). When learning from a given set of derivations, all decisions of the process except for the choice of θ are observed. In particular, the control flow of the process is known (up to certain permutations, as discussed above). Thus, the joint probability of a derivation and a set of parameters can be expressed as

$$p(\vec{\theta}) \cdot p(\text{deriv} | \theta) = p(\vec{\theta}) \cdot p(o_1 | \vec{\theta}) \cdot p(o_2 | \vec{\theta}) \cdot \dots \quad (9.6)$$

In addition, each decision only refers to a single parameter θ_a and does not depend on any other parameters or decisions, so the probability of one parameter θ_a and all decisions \vec{o}_a referring to that parameter can be expressed as

$$p(\theta_a, \vec{o}_a) = p(\theta_a) \cdot p(o_{a1} | \theta_a) \cdot p(o_{a2} | \theta_a) \cdot \dots \quad (9.7)$$

where o_{ai} denotes the i -th decision that uses the parameter θ_a . Finally, we have required the prior $p(\theta_a)$ to be conjugate to the distributions in which it is used, which means that the posterior of θ_a after observing an outcome o_{ai}

$$p(\theta_a | o_{ai}) \propto p(\theta_a) \cdot p(o_{ai} | \theta_a) \quad (9.8)$$

can be computed analytically and is of the same family as $p(\theta_a)$. Combining this property with Equation 9.7, we observe that the posterior distribution of θ_a given all \vec{o}_a can be computed incrementally, by iterating over the observations \vec{o}_a :

$$\begin{aligned} p(\theta_a | \vec{o}_a) &\propto p(\theta_a) \cdot p(o_{a1} | \theta_a) \cdot p(o_{a2} | \theta_a) \cdot \dots \\ &\propto p(\theta_a | o_{a1}) \cdot p(o_{a2} | \theta_a) \cdot \dots \\ &\propto p(\theta_a | o_{a1}, o_{a2}) \cdot \dots \end{aligned} \quad (9.9)$$

Equation 9.9 shows that under these particular constraints, the posterior distribution of a parameter can be obtained exactly using the following inference scheme: For each parameter, the posterior is initialized with the prior distribution. Going through a derivation, each time a decision uses the parameter, the posterior is updated based on the observed outcome of the decision. For example, for a decision with outcome $o_{ai} \sim$

Algorithm 9.4 A baseline parsing algorithm.

```

1: function PARSERANDOM(surface)
2:   deriv ← []
3:   while |surface| > 1 do
4:     ops ← all possible leftmost reductions of surface
5:     if ops = ∅ then
6:       return nothing
7:     else
8:       op ← chooseRandom(ops)
9:       surface ← applyReduction(surface, op)
10:      deriv ← [op] ◦ deriv
11:  return deriv

```

Bernoulli(θ_a) and a corresponding parameter $\theta_a \mid o_{a1}, \dots, o_{a(i-1)} \sim \text{Beta}(\alpha, \beta)$ (before observing o_{ai}), the parameter’s distribution after observing o_{ai} is updated to

$$\theta_a \mid o_{a1}, \dots, o_{ai} \sim \text{Beta}(\alpha + o_{ai}, \beta + 1 - o_{ai}). \quad (9.10)$$

Since the parameters are chosen globally for all pieces, this procedure can be continued through all pieces in the training dataset. The intuition behind this inference scheme is that the outcome of each decision is evidence for the underlying parameter, so accumulating this evidence across all uses of the parameter yields its overall posterior distribution. Since different decisions do not influence each other except for changing control flow changing the outcome of one decision does not influence the outcome of another decision, only the presence or absence of the decision.⁵

9.4.2 Baseline Parsing

In order to evaluate the effect of learning (i.e., inference about the parameters), a dataset of hand-annotated analyses (the ground truth) is compared to a set of baseline derivations of the same pieces, which are chosen arbitrarily by a random parser. Due to the excessively large number of derivations for even moderately long pieces, completely uniform sampling of a derivation (which would require enumerating all derivations) is not feasible. Instead, an arbitrary derivation is selected in a bottom-up fashion. The algorithm used by the baseline parser is shown in Algorithm 9.4. It applies the generative process backwards, starting with the full piece. In each step, the set of possible

⁵Generally, probabilistic programs do not satisfy the independence conditions that enable this inference scheme to work. In that case, approximate inference methods based on sampling or optimization can be used.

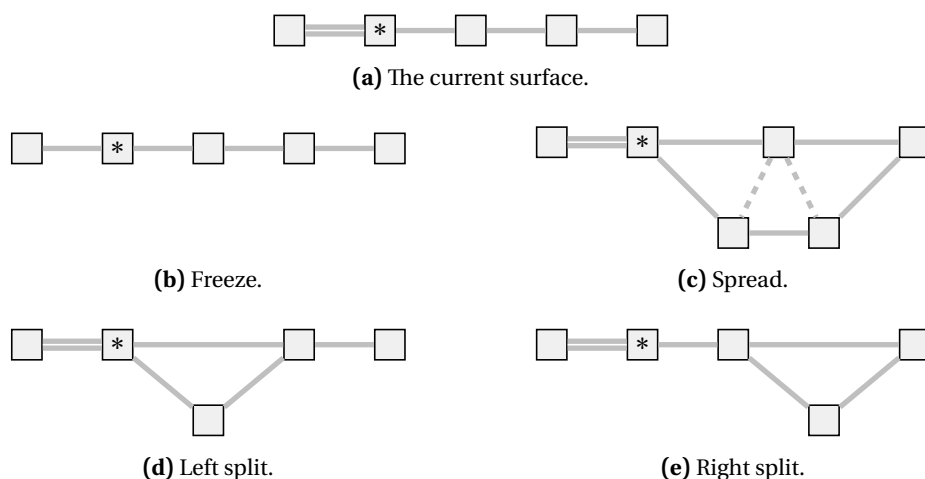


Figure 9.2 – Four possible ways to reduce the current surface (a) at the point between frozen and open transitions (*).

reductions of the current surface is enumerated (l. 4), i.e., all possible freeze, spread, or (left or right) split operations that could have produced the current surface. A random reduction is selected, applied to the current surface, and added to the derivation (ll. 8-10). The baseline parser does not backtrack, so if it encounters into an irreducible configuration, it will abort and has to be restarted, similar to a rejection sampler. Although such dead-end configurations are possible, the parser has never run into one during the experiments presented here, so they seem to be extremely rare. Thus, the non-backtracking, greedy strategy is sufficiently efficient for the present purpose. While this procedure does not sample the distributions uniformly (thus introducing a potential bias), it does not make use of any musical information.

The possible outer-structure reductions of the current surface are shown in Figure 9.2. Reductions are always applied according to leftmost-derivation order, as described in Chapter 6, i.e., at the point between frozen and open transitions (*). Depending on the number of frozen and open transitions, not all reductions are possible. For example, at the beginning of the reduction, all transitions are frozen, so the only possible operation is to unfreeze the last transition. In addition, restrictions about the derivation order (no left split or freeze after a right split) must be observed.

The parser chooses uniformly from the available reductions, i.e., from all possible ways to realize the permitted outer-structure reductions internally. This does not guarantee that the parser samples uniformly from the overall derivations since the number of available reductions at a later step can vary depending on earlier decisions. However, for the present purpose it is sufficient that the obtained derivations are not systematically biased according to musically relevant criteria.

Name	Segment	Figure	Notes
Prelude in C major (BWV 939)	full piece	C.2	309
Prelude in D minor (BWV 940)	full piece	C.4	442
Invention in A minor (BWV 784)	mm. 1-2.1	7.4	23
Cello Suite in G major, Prelude (BWV 1007)	mm. 1-4	7.5	32
Fly Me to the Moon	mm. 1-8	7.7	30
Er weckt mich alle Morgen	mm. 1-7 (melody)	7.9b	11
The Moldau	main theme	7.10b	13
rare interval example		7.10c	3
Hinunter ist der Sonne Schein	mm. 1-4	7.11	47
Träumerei	mm. 1-5	7.12	124
French Suite in D minor, Allemande	mm. 1-2.2	7.14	63
Quiescenza example		7.19	18
suspension example		7.20a	7
cadence example		7.20b	16

Table 9.1 – The set of examples used for evaluating the model.

9.4.3 Results and Discussion

The probabilistic model is evaluated on a small dataset of analyses that consists of all complete reductions from Chapter 7 and derivations of two complete preludes from J. S. Bach’s five little preludes (No. 1 in C-major, BWV 939; and No. 2 in D-minor, BWV 940), shown in Appendix C.⁶ The list of examples in the dataset is shown in Table 9.1.

Using this dataset, the posterior distribution of the parameters can be computed according to the procedure described in Section 9.4.1. Due to the locally conjugate form of the model, the parameters are independent in the posterior and each parameter’s posterior is in the same distribution family as its prior, which in this case are beta and Dirichlet distributions. While some of the inferred parameters have nearly symmetric parameters (e.g., keepL: Beta(296, 309), the preference for keeping an edge in the left child transition), most posteriors indicate a clear preference in one direction. The results also show, how the variance of the distribution (which correspond inversely to the magnitude of the parameters in beta and Dirichlet distributions) depends on the number of decisions that the model has seen for the corresponding parameter: Freezing a single

⁶Scores and analyses are available at <https://github.com/DCMLab/protovoice-annotations/tree/main/theory-article> and <https://github.com/DCMLab/protovoice-annotations/tree/main/bach/fünf-kleine-präludien>. The analyses can be viewed more conveniently and in greater detail using the protovoice viewer at <https://dcmllab.github.io/protovoice-annotation-tool/viewer/>.

transition is a rare decision, that only happens when only one open transition is left, so the corresponding parameter has a high variance (singleFreeze: Beta(15, 1)), while the parameters for frequent decisions, such as how often to elaborate a note in the right parent slice (elaborateR: Beta(677, 58)), have a lower variance. The full list of posteriors is shown in Table C.1.

The performance of the model is evaluated using leave-one-out cross-validation, inferring the posterior distribution of the model parameters from each training set. For the remaining test derivation, the predictive probability of the derivation is computed, marginalizing⁷ over the possible parameter values

$$p(d_{\text{test}} | \vec{d}_{\text{train}}) = \int_{\vec{\theta}} p(d_{\text{test}} | \vec{\theta}) p(\vec{\theta} | \vec{d}_{\text{train}}). \quad (9.11)$$

To make this quantity comparable between pieces of different sizes, it is adjusted exponentially⁸ by the number of notes in a piece N (summing over all slices) or, equivalently, its logarithm is scaled by $\frac{1}{N}$, the *log-probability per note* (logppn):

$$\text{logppn}(d) = \frac{1}{N} \log p(d), \quad (9.12)$$

which can also be interpreted as the negative cross-entropy per note in *nats* (using the natural logarithm). The negative exponential of this quantity is the *perplexity per note* (perppn):

$$\text{perppn}(d) = \exp(-\text{logppn}(d)). \quad (9.13)$$

When aggregating over all cross-validation splits, the overall logppn is first summed over all test pieces and then scaled by the total number of notes:

$$\text{logppn}(\vec{d}) = \frac{1}{\sum_i N_i} \sum_i \log p(d_i). \quad (9.14)$$

This effectively assigns a different weight to each test piece (proportional to its size) but keeps the overall influence per note (and thus roughly per decision that needs to be made) constant across pieces.

The results of the cross-validation are shown in Table 9.2. Training the model on the training sets drastically increases the (log) probability of the test analyses compared to its prior probability. The average amount of information needed to encode one surface note

⁷The integral over the parameters is approximated by drawing samples from the respective parameter distribution and averaging the resulting likelihood over the drawn samples. Since the parameters are independent, a small number of samples is sufficient to explore this otherwise very high-dimensional space.

⁸Since probabilities are multiplicative, their average should also be multiplicative (i.e., geometric), while log-probabilities are additive (i.e., arithmetic).

9.4 Evaluating the Proof-of-Concept Model

test data	annotated	annotated	random baselines
model	untrained (prior)	trained (posterior)	trained
	$\log p(d_{\text{test}})$	$\log p(d_{\text{test}} \vec{d}_{\text{train}})$	$\frac{1}{B} \sum_i \log p(b_i \vec{d}_{\text{train}})$
logppn (nats)	-6.602	-4.512	-5.357
logppn (bits)	-9.525	-6.510	-7.729
perppn	736.58	91.11	212.18

Table 9.2 – Log-probability per note (higher is better) and perplexity per note (lower is better), evaluated on test pieces (d_{test}) before and after seeing the training data (\vec{d}_{train}), and on baseline derivations of the same pieces (\vec{b}) after training.

decreases by about 2 nats (or 3 bits). Thus, the trained model assigns more probability mass to the annotated derivations and generally less mass to other derivations. However, from this alone we cannot tell how the probability mass is distributed among these non-examples. In the worst case, the model has only learned to assign more plausibility to all derivations that generate the example pieces (as opposed to derivations of other pieces), and in effect only distinguishes plausible from implausible *pieces* rather than plausible from implausible *derivations* (of the same piece).

To further investigate this issue, each test derivation is compared to 100 baseline derivations of the same piece. These derivations have been obtained by the baseline parser described in Section 9.4.2. The posterior logppn is computed on and averaged over the baseline derivations of the same piece and then aggregated across the cross-validation splits in the same way as the logppn of the test derivations. The results in Table 9.2 indicate that the difference between test and baseline derivations is not as pronounced as the one between untrained and trained performance on the test derivations, which means that the model’s plausibility judgment refers to some extent to the pieces rather than their derivations. However, there is still a pronounced difference between test pieces and baselines (about 0.8 nats or 1.2 bits), which shows that even this extremely simplistic version of the model is on average able to distinguish plausible from less plausible derivations based on previous examples.

A detailed overview of the difference between baseline and test derivations is shown in Figure 9.3. The logppn in each row is computed over the notes of the corresponding test piece. The blue boxes and diamonds show a summary of the baseline logppns, the orange points indicate the logppn of the test derivation. In all but one case (*Fly Me to the Moon*)⁹, the annotated derivation is rated higher than all of the baseline derivations.

⁹For the rare interval example, there are only five possible derivations, so the baselines include the annotated derivation. Small fluctuations in the logppn values for these identical derivations are due to the

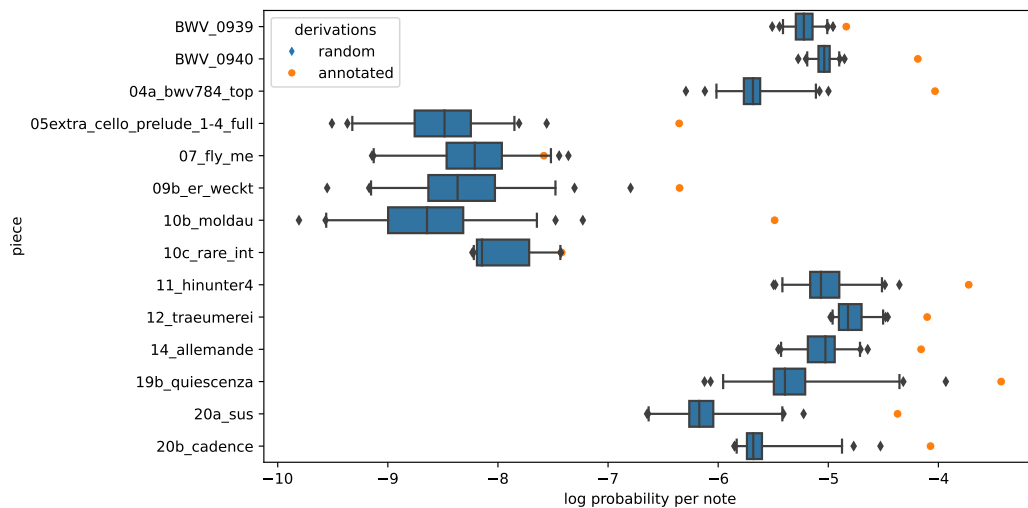


Figure 9.3 – Baseline (blue boxes and diamonds) vs. test derivation (orange points) logppn for each cross-validation split (indicated by the corresponding test piece). The box whiskers show the 2nd and 98th percentiles.

Even in the remaining case, the test derivation is rated relatively high, close to the 98th percentile.

9.5 Conclusion and Future Work

This chapter has demonstrated how the generative protovoice model can be fleshed out into a fully Bayesian model by expressing it as a probabilistic program. Even the simplistic proof-of-concept version of the model was able to learn useful parameters from a small dataset and to distinguish plausible from implausible derivations based on these observations. This constitutes the missing piece that connects the protovoice model as an analytical language (as it is used in Chapter 7) with the cognitive framework of Bayesian perception. The model is thus more than a music-theoretic tool to express musical intuitions, it becomes a computational-level model of how these intuitions could be formed by a listener.

With the current model being a proof-of-concept, there are three main directions in which the model could be improved. First, the way that decisions are made and evaluated is an oversimplification and should be based on more fine-grained musical knowledge. For example, the choice of elaborations could take into account harmonic knowledge,

approximate marginalization over the parameters.

both for generating typical ornaments *to* an existing harmony (as in Chapter 4) and for selecting ornaments that together form new harmonic entities. As argued in Chapter 7, the combination of ornamenting harmonic entities note-wise and coordinating these ornaments into new harmonic entities gives rise to harmonic syntax. Furthermore, the coordination of protovoices similar to contrapuntal coordination (e.g., preferring contrary over parallel motion) should be accounted for. Another criterion for good analyses is coherence, i.e., ensuring that similar passages are analyzed analogously as instances of a recurring pattern or motive. Implementing these aspects results in a more complex model in which decisions are made based on their context (e.g., the notes in the parent slices, or already generated concurrent notes), potentially using additional latent variables, such as harmonic information about a slice or global patterns. Both aspects require new modeling and inference methods: Latent variables pose a problem for the simple analytical inference method used here, but could be solved with sampling-based inference. Large contexts are problematic when each possible context configuration is treated independently, which leads to an explosion of model parameters (every possible context paired with every possible decision) and a high probability that a specific combination of context and outcome is never observed in the training data.¹⁰ A possible solution to this problem is to share information between different contexts, e.g., using a neural network. The parameters of the network then become parameters of the probabilistic model and their posterior distributions can be inferred using variational inference (Rezende et al. 2014; Graves 2011).

Second, the model so far does not cover the aspect of rhythm and meter. The surface it generates does not contain all information about the notes, but only their pitch and proto-temporal organization (simultaneity and succession). Generating the rhythmic and metrical structure that allows placing the notes in time would require a conceptualization of rhythmic information of latent entities. One possible approach is to integrate the model with a rhythm grammar (Yust 2018; Rohrmeier 2020b), but this is not straightforward because of the two complementary outer structure operations in the protovoice model.

Third, while parsing pieces into protovoice derivations is theoretically possible (as the two parsing algorithms in Chapter 6 and in this chapter show), there is currently no practically feasible solution that obtains an analysis with high plausibility for a given non-trivial piece. The exhaustive chart parsing approach from Chapter 6 cannot handle large pieces due to combinatorial explosion, while the random parser in Algorithm 9.4 does not take the quality of the derivation into account. A possible solution would be to modify Algorithm 9.4 to use a non-random policy (l. 8) and instead use heuristics to select a good

¹⁰This is known as the zero-frequency problem in n-gram/Markov models (Witten and Bell 1991), but it applies to all conditional distributions with large condition spaces.

next reduction step. The architecture of such a heuristic parser is a typical reinforcement learning scenario: A sequence of actions (reduction steps) is chosen one after the other, with a sparse reward (the probability of the final derivation) at the end of the process (see also Weber et al. 2015). The right policy for choosing the next reduction step could therefore be learned using reinforcement learning techniques. Since the state space in this process is extremely large (consisting of the previous reduction steps as well as the current surface), deep reinforcement learning techniques can be used to generalize over different states (Mnih et al. 2015), potentially in combination with planning methods (Silver, Huang, et al. 2016). When paired with a deep generative model, parser and generative model can be trained in a semi-supervised or unsupervised fashion, in which the generative model is trained on the results of a parser while the parser is trained on the judgment of the generative model, as in an asymmetric and cooperative version of self-play (Silver, Schrittwieser, et al. 2017). Such a training scheme aims to choose parameters in such a way that the given pieces are represented as faithfully as possible, without the need to learn from human-annotated analyses. It thus serves as a possible approach to investigate how a human can learn musical representations only from listening, without (or with limited) explicit theoretical instruction.

10 Conclusion

10.1 Looking Back

Let us briefly summarize the results of the previous chapters. The protovoice model is based on a generative process that produces the surface notes of a piece (up to their exact rhythm). This process is based on coordinated elaboration between concurrent notes, generating the structural relations of interest (simultaneity, sequentiality, and functionality) together with the notes. In Chapter 7, it was shown how these structural primitives can be used to express a number of musical phenomena, including latent voice-leading structure in monophony and (free) polyphony, and the relation between latent entities and their surface realizations. Chapters 6, 8, and 9 have demonstrated how this model can be implemented computationally and used to parse a given surface, encode analyses, and learn to judge the plausibility of competing analyses.

How do these results answer the questions raised by the preliminary studies in Chapters 5 to 4? Compared to the melodic grammar introduced in Chapter 5, the protovoice model solves the problem of polyphonic organization. Instead of a single line of notes, a network of several concurrent lines is established. One consequence of this extension is that sequentiality (which used to be implicit in the melody: one note is followed by the next in the sequence) is now modeled explicitly. This separation of surface adjacency and structural sequentiality extends the range of possible analyses even for monophonic melodies: Several concurrent streams of notes can be ascribed to a melody which captures the phenomenon of implicit polyphony. Correspondingly, notes that are non-simultaneous on the surface can become simultaneous on a higher level, which the melody grammar was unable to express. As a result, latent vertical organization is redefined: Where the melody grammar is integrated with a mode, which shapes the operations of the melody but remains constant throughout the piece, the protovoice model captures verticality purely as groups of notes, which is less specific on its own, but on the other hand can be linked with different ways of organizing pitch material.

The main issue with the schema matcher identified in Chapter 3 was that the heuristic approach to distinguishing schema instances from non-instances is not the criterion on

which such a decision is ideally based. In other words, what it *means* to say that a surface segment is an instance of a schema is not that some of its notes resemble the prototype and are sufficiently salient, but that the prototype is *underlying* the surface, i.e., has a generative and explanatory function. In particular, the surrounding non-structural notes must be accounted for by a plausible transformation of the schema prototype into the exact surface configuration. The protovoice model turns this insight around and characterizes all interpretations of a piece as inferred transformations from simpler to more complex configurations. These simpler structures do not have to correspond to schemata, but when they do, then a protovoice derivation provides exactly the transformation from schema prototype to the surface that marks a schema instance. It thus provides a very detailed account of the relation between *style structure* (common structures shared by a style) and *idiostructure* (the specific configurations of a particular piece, Narmour 1977).

Finally, Chapter 4 gave an account of how harmonic types (as another form of latent entities) relate to ornamental surface tones in a systematic way. The protovoice model complements this account by detailing the derivation from harmonic entities to precise surface configurations. Like schemata, harmonies are not fundamental entities in the protovoice model but can be modeled on top of more primitive structures. Knowledge about harmonic types and their ornaments can either be used implicitly (as in the manually obtained analyses in Chapter 7), or explicitly when judging the plausibility of a derivation using a probabilistic model (as suggested in Chapter 9).

Looking back even further: How does the protovoice model address the issues raised in the introduction? First of all, the protovoice model solves the problem of integrating three dimensions of relations among the surface notes – vertical, sequential, and dependency relations – all of which are latent and can exist on different levels of abstraction. It may seem as if this comes at the cost of strongly aligning the three dimensions, effectively reducing their respective expressiveness and independence.¹ However, coordinating the dimensions without collapsing them into one thing is precisely the contribution of the protovoice model: if the dimensions were completely independent, then we could not describe how they interact; if one (or two) dimensions take precedence (such as event hierarchy in the GTTM), then the others (harmonic structure for melodies, or voice leading between the events for homophonic events) cannot be expressed in their full complexity.

¹In particular, both sequential and dependency relations manifest largely as protovoice edges, which can make it look like the two are reduced to a single structure. This is not the case, as sequential and functional relations remain non-identical, despite being aligned in many cases. They differ, for example, in the case of a spread: The functional relation is between the parent note and the children (not between the children), while an edge between two children indicates sequentiality but not functional dependency.

Second, the protovoice model (or more specifically, the graph of notes and protovoice edges) proposes a new characterization of polyphonic structure, addressing the “voice problem”. It provides definition of horizontal, voice-like relations that is principled, interpretable, and general: *Principled*, because it ascribes a sequential connection if and only if it corresponds to a functional relation that has been introduced by an interpretable generative operation. *Interpretable*, because the set of generative operations is chosen to correspond to musically meaningful categories. This set of operations is not limited to the particular operations of the model presented here – different styles have different structural concepts and should thus be modeled through different operations that reflect these concepts. For example, when harmony starts to become independent of voice leading, the presence of a node might be justified directly as a representative of a chord or a Tonfeld (Haas 2004; Polth 2018; Rohrmeier and Moss 2021) rather than an elaboration of another note. Finally, *general*, because the protovoice network does neither make assumptions about the surface notes (e.g., the maximum number of simultaneous notes, every input can be parsed) nor specific assumptions about the underlying structures (e.g., a scaffold of 3 or 4 voices). This is due to a shift of perspective from voice relations as separate streams to a network of relations. It can therefore capture rare and complex cases and seeming exceptions while still expressing meaningful interpretations of the typical or simple cases.

Third, protovoice derivations are strong interpretations, i.e., they *explain observations* (the surface notes) from the listeners perspective. The latent structures encoded in a derivation (i.e., latent slices and protovoice edges) are used to derive each surface note, explaining its function in its context. When combined with higher-level interpretations (e.g., harmonies or schemata), a derivation even describes how the observed surface is an instantiation of an abstract category: The schema or chord gives rise to the observed notes, much like an object and a light source can give rise to an light pattern that we can observe, or a thought or intention gives rise to a verbal utterance. The protovoice model also represents a generalization over many observations, since the same operations and derivation patterns (encoded by the operation’s probabilities) can be shared across many pieces. The model thus captures both style structure and idiostructure, which shows that recursive, hierarchical analysis does not necessarily imply that idiostructure is ignored (cf. Narmour 1983). The insight provided by such an analysis is not only the structure that the piece is reduced to (on any level), but the reduction (or inversely, the derivation) process itself (Martin 1978).

Since the model derives a whole piece in a closed and coherent process, a full derivation represents a consistent interpretation of all notes at the same time, which can be seen as the *idealized goal* of the interpretation task. A human listener would not be expected to obtain such a complete interpretation while listening to a piece for the first time,

maybe not even after extensively studying a piece for a longer time. Nevertheless, even a preliminary understanding of a piece (with respect to the types of relations captured by the protovoice model) corresponds to a set of partial or underspecified derivations, and by integrating and refining this understanding, a full derivation is approached.

10.2 General Insights

A number of general observations can be made from the protovoice model and the general modeling approach, which provide insights for larger discourses in music theory and cognition. First of all, better understanding a domain can require reconceptualization. The main example of this phenomenon here is the shift from “voices as streams” to protovoices. In traditional accounts, sequential relations between notes have usually been identified with membership of the same “stream voice”, on some level of abstraction. It is easy to leave this view unquestioned when focusing on typical and prototypical cases, such as common contrapuntal patterns (R. Gjerdingen 2007; IJzerman 2019) or voice leading in highly constrained and regular styles such as strict polyphony (Jeppesen 1946). The key to testing the assumptions underlying a conceptualization is to look at the complex examples and edge cases in which the concept fails, and ask: Is this failure due to a different principle being at work (that overrides the concept in question), or should this example still be covered by the concept? In the case of voice leading, such edge cases are implied polyphony within voices, free polyphonic textures, and different types of strict voices (e.g., in fugues vs. chorales). Taken together, these examples show that there are at least three different perspectives on voice structure (cf. Cambouropoulos 2006): parts (i.e., written surface voices), auditory streams (Bregman 1990; Huron 2016), and tonal-functional relations between sequential notes. To be clear, none of these meanings of “voice” is privileged over the others, they simply correspond to different aspects of musical experience: compositional conventions, perceptual effects, and structural interpretation, respectively. However, it is important to be aware of their distinctness and to know, which concept is appropriate in which context. In particular, the fundamental type of voice in the context of tonal structure is the protovoice, not the streams or the part.

The protovoice model also demonstrates that computational models can be very expressive and meaningful. They are neither constrained to a fixed set of existing formalisms (such as Markov models or probabilistic grammars), nor do they have to be heuristic and ad-hoc when dealing with ambiguity. This is an important insight regarding the question whether formal and computational modeling is an appropriate (or even necessary) tool for music theory (Lerdahl and Jackendoff 1983; Temperley 1999a; Wiggins

2012; Rohrmeier and Pearce 2018), or whether it necessarily leads to oversimplification and decontextualization.² Formal models require abstraction and separation of different aspects (i.e., *analysis*), but that doesn't mean that they are not meaningful or that they systematically and necessarily fail to capture certain aspects. Mathematics and formal modeling are human-made tools, not given laws: they are designed (and can be changed, adapted, and invented) to make sense of reality by providing abstract descriptions of and generalizations over phenomena that are observed in reality. This insight is important for both music theorists and computational modelers: On one hand, modeling is not just a reduction of a complex object to numbers (or some fixed, preexisting set of abstract concepts). Quite contrary, the abstract concepts are designed to be meaningful and capture intuitions, and their systematic and formalistic character only serves their exactness. That something can be executed by a computer does not imply that it does not mean anything. On the other hand, it is important to note that mathematical consistency is not the only criterion for the plausibility of a model. It is important to understand the modeled domain and its phenomena well and to show that the model captures these phenomena accurately.

The complexity of the protovoice model suggests moreover that the similarities between syntactic structure in language and music might not be as deep as sometimes believed (Forte 1967; Winograd 1968; Bernstein 1976; Katz and Pesetsky 2011). In particular, Katz and Pesetsky (2011) base their claims about the identity of syntax in language and music on the GTTM, which does not take into account polyphonic structure at all. While the protovoice model is also a generative and recursive model (like grammars), the structures it generates are considerably different from (and arguably more complex than) the tree-like syntactic structures generated in language. The main differences between the two are the multidimensionality of the protovoice texture with both vertical and horizontal connections as opposed to a flat sequence of elements, as well as the coexistence of operations that elaborate slices and transitions. Taken separately, operations on only slices or only transitions would each result in the familiar tree-like dependencies (Yust 2006). In combination, however, no type of entity (slices or transitions) can take precedence over the other and become the primary object that would correspond to the elements of a syntax tree in language. For these reasons, the similarity between linguistic and musical syntax either relies on much more generic features (such as recursive dependencies) which are potentially shared with other domains (Fitch and Martins 2014), or, if music and language specifically share syntactic mechanisms, those would have to exhibit properties that go beyond what is needed for language in order to account for musical structure.

²Wiggins (2012) speaks of a “deep-seated horror of reductionism”.

10.3 Looking Onwards

There are several directions in which the work presented in this thesis can be continued. One such direction is a better model of how interpretations are obtained and how their plausibility is evaluated. While the current probabilistic version of the model works surprisingly well for a rather naive proof of concept, a more refined and informed model would take into account the context of the note to be generated, typical derivation patterns, as well as latent entities such as harmonies. Such a model could establish an explicit connection between higher-level concepts and structural primitives within the model instead of external to the model (as in Chapter 7). Similarly, more flexible parsing strategies are needed if the model should describe cognitive phenomena not just on the computational but also on the algorithmic level. One possible strategy is to learn heuristics for step-by-step reduction using reinforcement learning, which has been argued to describe aspects of cognition down to the neuronal level (Gerstner et al. 2018). The generative probabilistic model then takes the role of the reward, teaching the model how to choose interpretations (Weber et al. 2015). Another strategy is to allow incomplete, partial, and contradictory interpretations, which are then incrementally refined to obtain a more complete and consistent overall analyses, reinterpreting segments when required by their context. Such a model might more accurately describe how interpretations develop in actual human perception, rather than the idealized listener considered here.

Another important direction is to broaden the scope of the model. In its current form, the protovoice model focuses rather strictly on the pitch aspect of tonal structure. This raises the question how to integrate other musical aspects such as temporal structure (Yust 2018; Rohrmeier 2020b; Lerdahl and Jackendoff 1983). A similar point could be made about more style-specific aspects such as regularities of surface voices, form, or motivic structure.

Broadening the scope of the model also means applying its underlying ideas to styles of music that use different structural primitives. Chapter 5 has already demonstrated how an abstract model can describe phenomena across styles. While the specific details of a model (e.g., its operations and their interpretation) will generally differ between different styles, the idea of strong interpretations as a form of “making sense” of the observed surface is a general cognitive phenomenon that on its own is independent from cultural influences (although its concrete instantiation is very much dependent on individual experience and thus cultural context). It thus remains a question that we have conveniently ignored so far: How is the set of generative operations and structural primitives obtained in the first place? In other words, Chapter 9 has shown how to learn the probabilities in a probabilistic program, but where does the program itself come from? To some extent, the rules of a system can be inferred from plain observations

(Harasim 2020; Ullman and Tenenbaum 2020a; Tenenbaum et al. 2011), but even then a space of hypotheses, of possible models must be assumed. The fact that the same inference problem arises in various domains of perception and cognition (such as language, vision, and conceptual reasoning) suggests that these domains might share a common space of representable and learnable structures. In that sense, studying the rich and complex structures that we encounter in various musical styles and cultures would ultimately help us understand how humans make sense of the world in general.

Appendix

Part IV

A Chord Types and Ornamentation

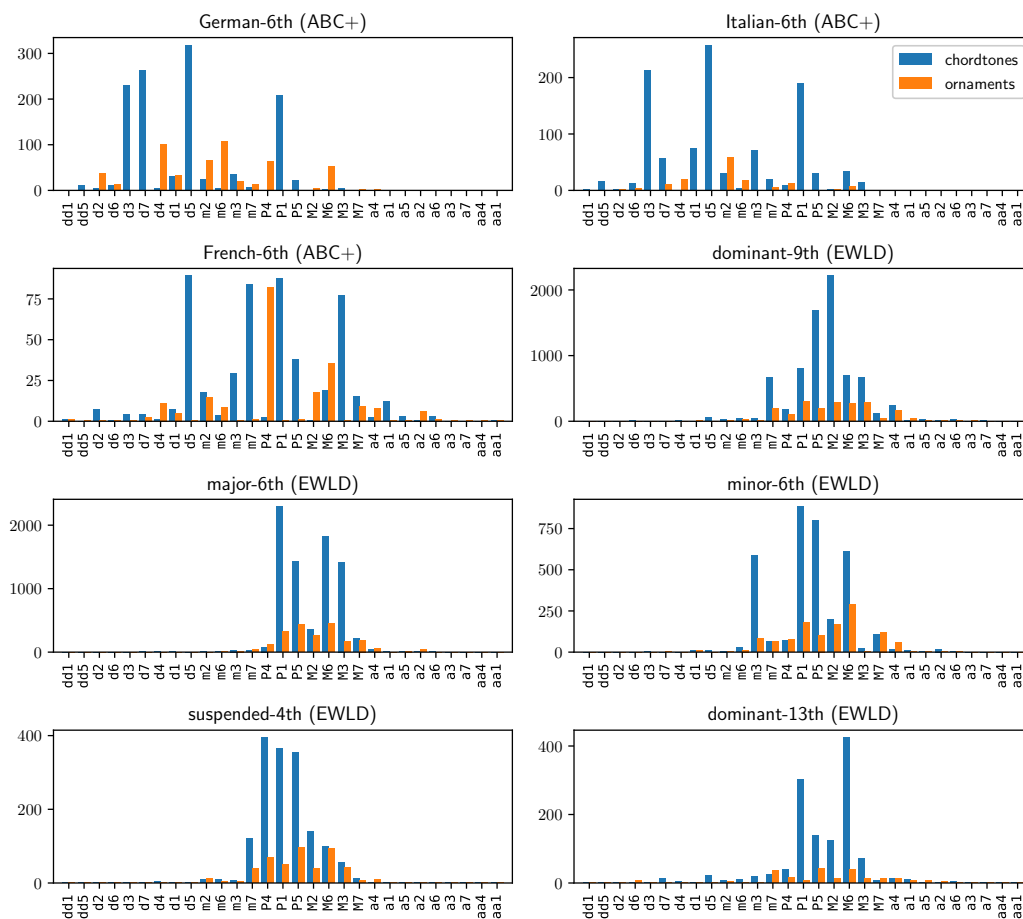


Figure A.1 – The posterior distributions of the chordtones $\phi^{(ct)}$ (blue, left-leaning bars) and ornaments $\phi^{(or)}$ (orange, right-leaning bars) of the chord types that are specific to either the ABC+ or the EWLD corpus. The chords that are common to both datasets are shown in Figure 4.5 Pitches are ordered according to the line of fifths and expressed as intervals relative to the root (P1, unison). (Continues on next page.)

Appendix A. Chord Types and Ornamentation

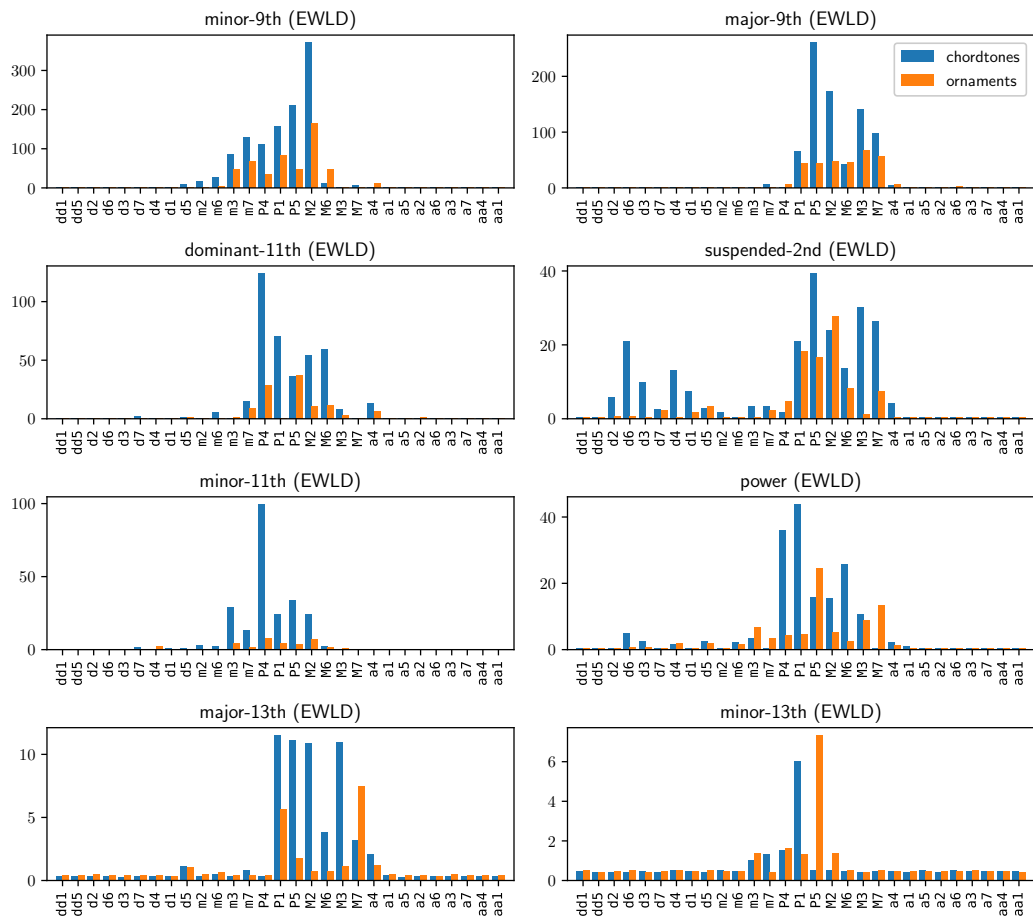


Figure A.1 – (cont.)

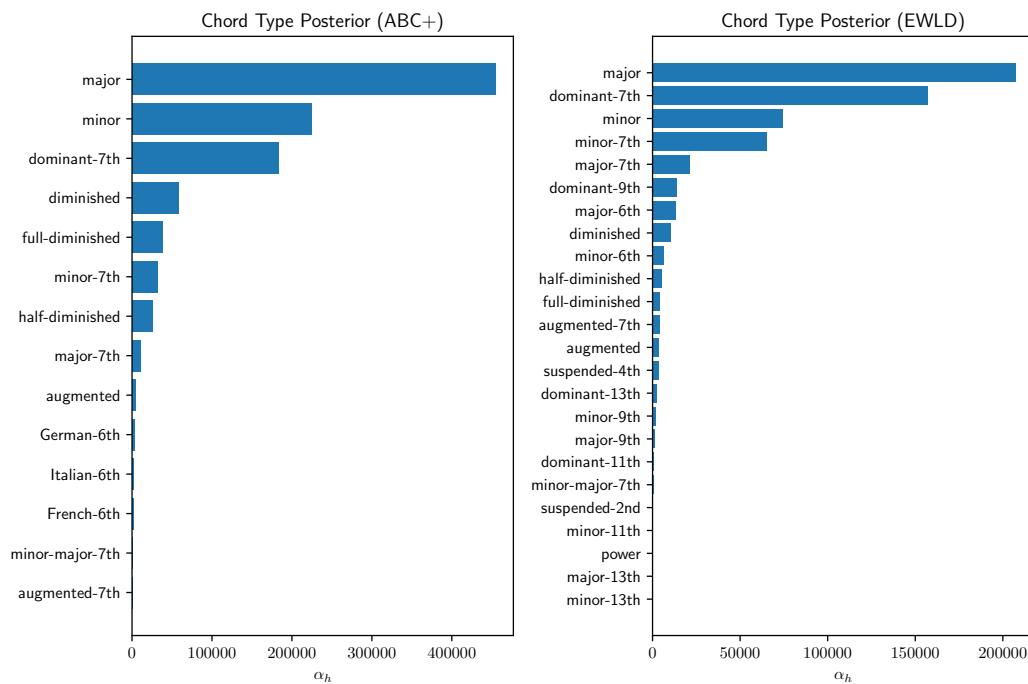


Figure A.2 – Posterior distributions of the chord type probabilities χ . The bars indicate the parameters of the posterior Dirichlet distribution, where one parameter corresponds to each chord type and indicates the prevalence of that chord type. The values essentially correspond to the number of occurrences of each chord types.

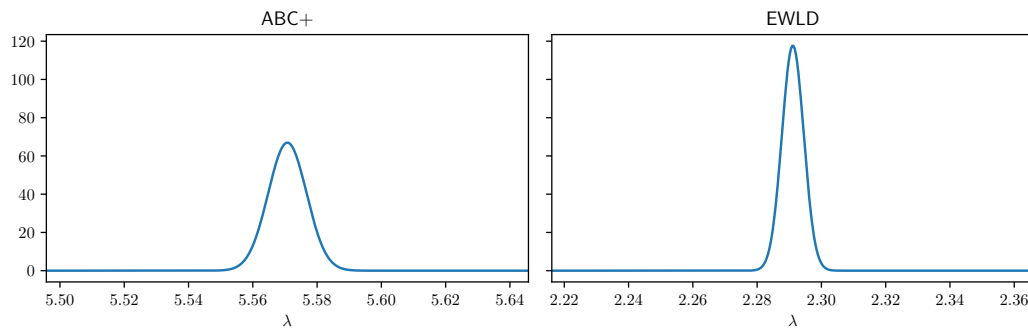


Figure A.3 – Posterior distributions of the note rate λ for the ABC+ corpus (left) and the EWLD corpus (right). Both plots share the y-axis and have the same scaling on the x-axis, so the variance of the distributions can be compared directly. The mean, however, is much smaller for the EWLD data, probably due to the fact that it consists of melodies instead of full scores.

B Examples of the Annotation Format

B.1 Example of an Input Piece

```
[
  {
    "notes": [
      { "id": "slice0.note1", "hold": false, "pitch": "E4" },
      { "id": "slice0.note0", "hold": true, "pitch": "C4" }
    ],
    "time": "1.1.0"
  },
  {
    "notes": [
      { "id": "slice1.note3", "hold": true, "pitch": "D4" },
      { "id": "slice1.note0", "hold": false, "pitch": "C4" }
    ],
    "time": "1.2.0"
  },
  {
    "notes": [
      { "id": "slice2.note4", "hold": false, "pitch": "B3" },
      { "id": "slice2.note3", "hold": false, "pitch": "D4" }
    ],
    "time": "1.3.0"
  },
  {
    "notes": [{ "id": "slice3.note6", "hold": false, "pitch": "C4" }],
    "time": "1.4.0"
  }
]
```

Listing B.1 – An example of a `.piece.json` file.

B.2 The JSON Format of an Analysis File

```
type AnalysisJSON =
  { derivation :: Array LeftmostJSON
  , start :: SliceJSON
  , topSegments ::
      Array
        { trans :: TransitionJSON
        , rslice :: SliceJSON
        }
  }

type SliceJSON =
  { id :: SliceId -- an integer
  , notes :: StartStop (Array Note) -- "start", "stop", or [notes]
  }

type Note = { pitch :: String, id :: String }
type TransitionJSON =
  { id :: TransId -- an integer
  , edges :: EdgesJSON
  , is2nd :: Boolean
  }

type EdgesJSON = { regular :: Array Edge, passing :: Array Edge }
type Edge = { left :: StartStop Note, right :: StartStop Note }
type LeftmostJSON = Variant
  ( freezeLeft :: FreezeJSON
  , freezeOnly :: FreezeJSON
  , splitLeft :: SplitJSON
  , splitRight :: SplitJSON
  , splitOnly :: SplitJSON
  , hori :: HoriJSON
  )

type FreezeJSON =
  { ties :: Array Edge
  , prevTime :: String
  }

type ChildrenJSON = Array { child :: Note, orn :: Maybe String }
type SplitJSON =
  { regular :: Array { parent :: Edge, children :: ChildrenJSON }
  , passing :: Array { parent :: Edge, children :: ChildrenJSON }
  , fromLeft :: Array { parent :: Note, children :: ChildrenJSON }
  }
```

```

, fromRight :: Array { parent :: Note, children :: ChildrenJSON }
, unexplained :: Array Note
, keepLeft :: Array Edge
, keepRight :: Array Edge
, passLeft :: Array Edge
, passRight :: Array Edge
, ids :: { left :: TransId, slice :: SliceId, right :: TransId }
}
type HoriJSON =
{ children ::
  Array
  { parent :: Note
  , child ::
    Variant
    ( leftChild :: Note
    , rightChild :: Note
    , bothChildren :: { left :: Note, right :: Note }
    , tooManyChildren ::
      { left :: Array Note
      , right :: Array Note
      }
    )
  }
, unexplained :: { left :: Array Note, right :: Array Note }
, midEdges :: EdgesJSON
, ids ::
  { left :: TransId -- an integer
  , lslice :: SliceId -- an integer
  , mid :: TransId
  , rslice :: SliceId
  , right :: TransId
  }
}

```

Listing B.2 – The full format of a `.analysis.json` file expressed as PureScript types.

B.3 Example of an Analysis File

```
{
  "topSegments": [
    {
      "trans": {
        "is2nd": false,
        "id": 9,
        "edges": {
          "regular": [{ "right": "stop", "left": "start" }],
          "passing": []
        }
      },
      "rslice": { "notes": "stop", "id": 5 }
    }
  ],
  "start": { "notes": "start", "id": 0 },
  "derivation": [
    {
      "type": "splitOnly",
      "value": {
        "unexplained": [],
        "regular": [
          {
            "parent": { "right": "stop", "left": "start" },
            "children": [
              {
                "orn": "RootNote",
                "child": { "pitch": "E4", "id": "notes6.0" }
              },
              {
                "orn": "RootNote",
                "child": { "pitch": "C4", "id": "notes6.1" }
              }
            ]
          }
        ]
      }
    }
  ],
  "passing": [],
  "passRight": [],
  "passLeft": [],
}
```

```

    "keepRight": [],
    "keepLeft": [],
    "ids": { "slice": 6, "right": 8, "left": 7 },
    "fromRight": [],
    "fromLeft": []
  }
},
{
  "type": "hori",
  "value": {
    "unexplained": { "right": [], "left": [] },
    "midEdges": {
      "regular": [
        {
          "right": { "pitch": "C4", "id": "slice3.note6" },
          "left": { "pitch": "C4", "id": "slice0.note0" }
        }
      ],
      "passing": [
        {
          "right": { "pitch": "C4", "id": "slice3.note6" },
          "left": { "pitch": "E4", "id": "slice0.note1" }
        }
      ]
    },
    "ids": { "rslice": 4, "right": 4, "mid": 6, "lslice": 1, "left": 0 },
    "children": [
      {
        "parent": { "pitch": "E4", "id": "notes6.0" },
        "child": {
          "type": "leftChild",
          "value": { "pitch": "E4", "id": "slice0.note1" }
        }
      },
      {
        "parent": { "pitch": "C4", "id": "notes6.1" },
        "child": {
          "type": "bothChildren",
          "value": {
            "right": { "pitch": "C4", "id": "slice3.note6" },

```

Appendix B. Examples of the Annotation Format

```
        "left": { "pitch": "C4", "id": "slice0.note0" }
      }
    }
  ]
}
},
{ "type": "freezeLeft", "value": { "ties": [], "prevTime": "" } },
{
  "type": "splitLeft",
  "value": {
    "unexplained": [],
    "regular": [
      {
        "parent": {
          "right": { "pitch": "C4", "id": "slice3.note6" },
          "left": { "pitch": "C4", "id": "slice0.note0" }
        },
        "children": [
          {
            "orn": "FullNeighbor",
            "child": { "pitch": "B3", "id": "slice2.note4" }
          }
        ]
      }
    ]
  },
  "passing": [
    {
      "parent": {
        "right": { "pitch": "C4", "id": "slice3.note6" },
        "left": { "pitch": "E4", "id": "slice0.note1" }
      },
      "children": [
        {
          "orn": "PassingMid",
          "child": { "pitch": "D4", "id": "slice2.note3" }
        }
      ]
    }
  ]
},
],
```



```

"passRight": [],
"passLeft": [],
"keepRight": [],
"keepLeft": [
  {
    "right": { "pitch": "B3", "id": "slice2.note4" },
    "left": { "pitch": "C4", "id": "slice0.note0" }
  },
  {
    "right": { "pitch": "D4", "id": "slice2.note3" },
    "left": { "pitch": "E4", "id": "slice0.note1" }
  }
],
"ids": { "slice": 3, "right": 3, "left": 5 },
"fromRight": [],
"fromLeft": []
}
},
{
  "type": "splitLeft",
  "value": {
    "unexplained": [],
    "regular": [
      {
        "parent": {
          "right": { "pitch": "B3", "id": "slice2.note4" },
          "left": { "pitch": "C4", "id": "slice0.note0" }
        },
        "children": [
          {
            "orn": "RightRepeatOfLeft",
            "child": { "pitch": "C4", "id": "slice1.note0" }
          }
        ]
      }
    ]
  },
  {
    "parent": {
      "right": { "pitch": "D4", "id": "slice2.note3" },
      "left": { "pitch": "E4", "id": "slice0.note1" }
    },
  },

```

Appendix B. Examples of the Annotation Format

```
    "children": [
      {
        "orn": "LeftRepeatOfRight",
        "child": { "pitch": "D4", "id": "slice1.note3" }
      }
    ]
  },
  "passing": [],
  "passRight": [],
  "passLeft": [],
  "keepRight": [
    {
      "right": { "pitch": "D4", "id": "slice2.note3" },
      "left": { "pitch": "D4", "id": "slice1.note3" }
    }
  ],
  "keepLeft": [
    {
      "right": { "pitch": "C4", "id": "slice1.note0" },
      "left": { "pitch": "C4", "id": "slice0.note0" }
    }
  ],
  "ids": { "slice": 2, "right": 2, "left": 1 },
  "fromRight": [],
  "fromLeft": []
}
},
{
  "type": "freezeLeft",
  "value": {
    "ties": [
      {
        "right": { "pitch": "C4", "id": "slice1.note0" },
        "left": { "pitch": "C4", "id": "slice0.note0" }
      }
    ]
  },
  "prevTime": "1.1.0/1"
}
},
```

```
{
  "type": "freezeLeft",
  "value": {
    "ties": [
      {
        "right": { "pitch": "D4", "id": "slice2.note3" },
        "left": { "pitch": "D4", "id": "slice1.note3" }
      }
    ],
    "prevTime": "1.2.0/1"
  }
},
{ "type": "freezeLeft", "value": { "ties": [], "prevTime": "1.3.0/1" } },
{ "type": "freezeOnly", "value": { "ties": [], "prevTime": "1.4.0/1" } }
]
```

Listing B.3 – An example of a .analysis.json file.

C The Probabilistic Model

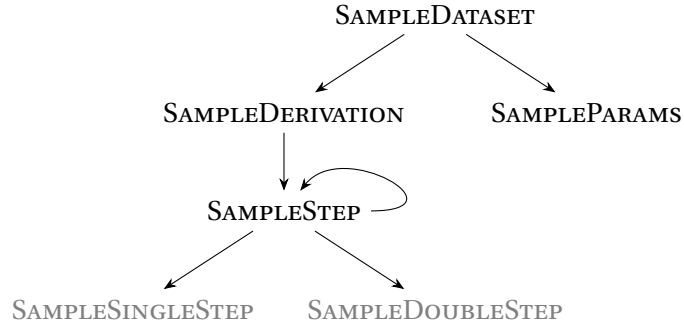
C.1 Model Parameters

	Parameter	Prior	Posterior
outer	singleFreeze	Beta(1, 1)	Beta(15, 1)
	doubleGoLeft	Beta(1, 1)	Beta(519, 233)
	doubleLeftFreeze	Beta(1, 1)	Beta(421, 31)
	doubleRightSplit	Beta(1, 1)	Beta(77, 233)
split	elaborateRegular	Beta(1, 1)	Beta(382, 82)
	elaborateL	Beta(1, 1)	Beta(621, 39)
	elaborateR	Beta(1, 1)	Beta(677, 58)
	rootFifths	Beta(1, 1)	Beta(67, 128)
	keepL	Beta(1, 1)	Beta(296, 309)
	keepR	Beta(1, 1)	Beta(246, 396)
	repeatOverNeighbor	Beta(1, 1)	Beta(239, 119)
	nbChromatic	Beta(1, 1)	Beta(8, 230)
	nbAlt	Beta(1, 1)	Beta(237, 5)
	repeatLeftOverRight	Beta(1, 1)	Beta(86, 51)
	repeatAlter	Beta(1, 1)	Beta(2, 135)
	repeatAlterUp	Beta(1, 1)	Beta(2, 1)
	repeatAlterSemis	Beta(1, 1)	Beta(2, 1)
	connect	Beta(1, 1)	Beta(101, 4)
	connectChromaticLeftOverRight	Beta(1, 1)	Beta(4, 2)
	passUp	Beta(1, 1)	Beta(6, 14)
	passLeftOverRight	Beta(1, 1)	Beta(1, 19)
	newPassingLeft	Beta(1, 1)	Beta(1821, 31)
	newPassingRight	Beta(1, 1)	Beta(1993, 22)
hori	newPassingMid	Beta(1, 1)	Beta(1638, 54)
	noteHoriDirection	Dirichlet ₃ (1)	Dirichlet ₃ (456, 264, 252)
	notesOnOtherSide	Beta(1, 1)	Beta(3, 1)
	horiRepetitionEdge	Beta(1, 1)	Beta(338, 524)

Table C.1 – An overview over all global parameters and their prior distributions.

C.2 Pseudocode of the Model

Algorithm C.1 – Top-level structure of the model: sampling parameters and derivations.



```

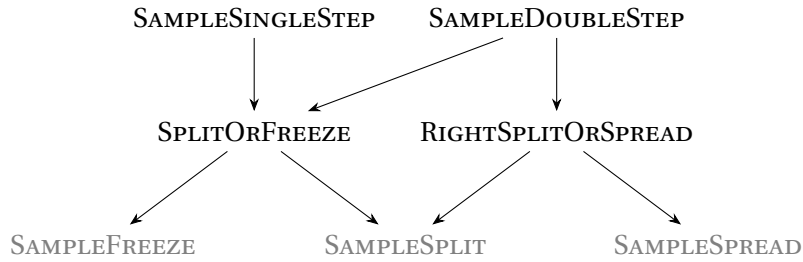
1: function SAMPLEDATASET( $N$ )
2:   global  $\vec{\theta} \leftarrow$  SAMPLEPARAMS( )
3:   for  $i \in [1, N]$  do
4:      $D_i \leftarrow$  SAMPLEDERIVATION( )
5:   return  $\vec{D}$ 

6: function SAMPLEPARAMS( )
7:   for  $a \in$  global parameters do
8:      $\theta_a \sim$  prior $_a$ 
9:   return  $\vec{\theta}$ 

10: function SAMPLEDERIVATION( )
11:   -- start with the empty piece and derivation
12:   return SAMPLESTEP( $\times - \times$ , [], false)

13: function SAMPLESTEP( $surface, deriv, afterRightSplit$ )
14:   -- check number of open transitions
15:   if 0 open transitions in  $surface$  then
16:     return  $deriv$ 
17:   else if 1 open transition then
18:      $op \leftarrow$  SAMPLESINGLESTEP( $surface$ )
19:   else
20:      $op \leftarrow$  SAMPLEDOUBLESTEP( $surface, afterRightSplit$ )
21:      $surface' \leftarrow$  applyLeftmost( $op, surface$ )
22:   return SAMPLESTEP( $surface', deriv + op, isRightSplit(op)$ )
  
```

Algorithm C.2 – Sampling outer structure (splits, spreads, freezes).



```

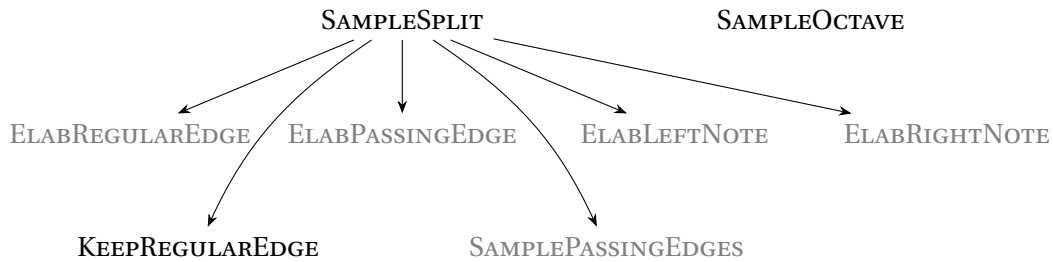
1: function SAMPLESINGLESTEP(surface)
2:   return makeSingleOp(SPLITORFREEZE(last(surface), freezeSingle))

3: function SAMPLEDOUBLESTEP( $\theta$ , surface, afterRightSplit)
4:   if afterRightSplit then
5:     return RIGHTSPLITORSREAD(fstOpen(surface), sndOpen(surface))
6:   else
7:     continueLeft  $\sim$  Bernoulli( $\theta_{\text{doubleGoLeft}}$ )
8:     if continueLeft then
9:       makeLeftOp(SPLITORFREEZE(fstOpen(surface), doubleLeftFreeze))
10:    else
11:      RIGHTSPLITORSREAD(fstOpen(surface), sndOpen(surface))

12: function SPLITORFREEZE(transition, param)
13:   if transition is freezable then
14:     shouldFreeze  $\sim$  Bernoulli( $\theta_{\text{param}}$ )
15:     if shouldFreeze then
16:       return SAMPLEFREEZE(transition)
17:     else return SAMPLESPLIT(transition)
18:   else return SAMPLESPLIT(transition)

19: function RIGHTSPLITORSREAD(leftTrans, rightTrans)
20:   shouldSplitRight  $\sim$  Bernoulli( $\theta_{\text{doubleRightSplit}}$ )
21:   if shouldSplitRight then
22:     return makeRightSplit(SAMPLESPLIT(rightTrans))
23:   else
24:     return SAMPLESPREAD(sliceBetween(leftTrans, rightTrans))
  
```


Algorithm C.4 – Sampling a split.



```

1: function SAMPLESPLIT( $\theta$ , surface, transition)
2:   -- sample children:
3:   childrenReg  $\leftarrow$  [ELABREGULAREDGE(edge) for edge  $\in$  regularEdges(transition)]
4:   childrenPass  $\leftarrow$  [ELABPASSINGEDGE(edge) for edge  $\in$  passingEdges(transition)]
5:   childrenL  $\leftarrow$  [ELABLEFTNOTE(note) for note  $\in$  leftSlice(transition)]
6:   childrenR  $\leftarrow$  [ELABRIGHTNOTE(note) for note  $\in$  rightSlice(transition)]
7:   -- collect generated objects:
8:   middleSlice  $\leftarrow$  makeSlice(childrenReg,childrenPass,childrenL,childrenR)
9:   (edgesL, edgesR)  $\leftarrow$  makeEdges(childrenReg,childrenPass,childrenL,childrenR)
10:  -- generate passing edges and drop/keep regular edges:
11:  passL  $\leftarrow$  SAMPLEPASSINGEDGES(left(transition), middleSlice)
12:  passR  $\leftarrow$  SAMPLEPASSINGEDGES(middleSlice, right(transition))
13:  keepL  $\leftarrow$  [KEEPREGULAREDGE(edge) for edge  $\in$  edgesL]
14:  keepR  $\leftarrow$  [KEEPREGULAREDGE(edge) for edge  $\in$  edgesR]
15:  return makeSplit(childrenReg, childrenPass, ...)

16: function KEEPREGULAREDGE(edge)
17:   keep  $\sim$  Bernoulli( $\theta_{\text{keep}}$ )
18:   return (edge, keep)

19: function SAMPLEOCTAVE()
20:   -- excluded from the inference process, pseudo-uniform
21:   return some octave
  
```

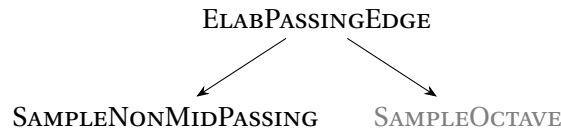
Algorithm C.5 – Elaborating regular edges.



```

1: function ELABREGULAREDGE(parentEdge → (leftParent, rightParent))
2:   n ~ Geometric1( $\theta_{\text{elaborateRegular}}$ )
3:   children ← permute n do
4:     if parentEdge = (⊗, ⊗) then
5:       -- root edge? -> generate root note
6:       fifthsDir ~ Bernoulli(0.5)
7:       fifthsN ~ Geometric0( $\theta_{\text{rootFifths}}$ )
8:       octaves ← SAMPLEOCTAVE() + 4
9:       yield makeRootNote(fifthsDir, fifthsN, octaves)
10:    else
11:      -- non-root edge -> generate normal ornament
12:      if degree(leftParent) = degree(rightParent) then
13:        -- parents equal -> full neighbor or full repeat
14:        repeat ~ Bernoulli( $\theta_{\text{repeatOverNeighbor}}$ )
15:        if repeat then
16:          yield makeFullRepeat(leftNote + SAMPLEOCTAVE())
17:        else
18:          goUp ~ Bernoulli(0.5)
19:          yield makeFullNeighbor(SAMPLENEIGHBOR(goUp, leftParent))
20:      else
21:        -- parents not equal -> repeat left or right
22:        repeatLeft ~ Bernoulli( $\theta_{\text{repeatLeftOverRight}}$ )
23:        alter ~ Bernoulli( $\theta_{\text{repeatAlter}}$ )
24:        if alter then
25:          direction ~ Bernoulli( $\theta_{\text{repeatAlterUp}}$ )
26:          semitones ~ Geometric1( $\theta_{\text{repeatAlterSemis}}$ )
27:          alteration ← makeAlteration(direction, semitones)
28:        else alteration ← perfectUnison
29:        octaves ← SAMPLEOCTAVE()
30:        if repeatLeft then
31:          makeRightRepeatOfLeft(parentLeft, alteration, octaves)
32:        else
33:          makeLeftRepeatOfRight(parentRight, alteration, octaves)
34:    return (parentEdge, children)
  
```

Algorithm C.6 – Elaborating passing edges.

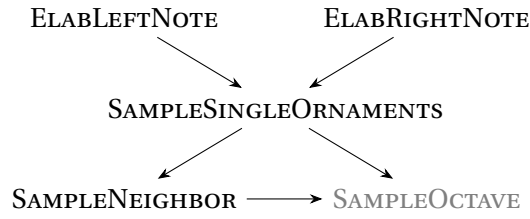


```

1: function ELABPASSINGEDGE(parentEdge → (leftParent, rightParent, n))
2:   children ← permute n do
3:     if diatonicDistance(leftParent, rightParent) = 1 then
4:       -- distance = 1 step -> chromatic passing tone
5:       onLeftSide ~ Bernoulli( $\theta_{\text{connectChromaticLeftOverRight}}$ )
6:       octaves ← SAMPLEOCTAVE()
7:       if onLeftSide then
8:         child ← parentLeft + makeAlteration(direction(parentEdge),1)
9:       else
10:        child ← parentRight - makeAlteration(direction(parentEdge),1)
11:      yield makeMidPassing(child)
12:    else if diatonicDistance(leftParent, rightParent) = 2 then
13:      -- distance = 2 steps -> can connect
14:      connect ~ Bernoulli( $\theta_{\text{connect}}$ )
15:      if connect then
16:        child ← SAMPLENEIGHBOR(direction(parentEdge), leftParent)
17:        yield makeMidPassing(child)
18:      else
19:        yield SAMPLENONMIDPASSING(leftParent, rightParent)
20:    else
21:      -- distance > 2 steps -> cannot connect
22:      yield SAMPLENONMIDPASSING(leftParent, rightParent)
23:  return (parentEdge, children)

24: function SAMPLENONMIDPASSING(leftParent, rightParent)
25:   onLeftSide ~ Bernoulli( $\theta_{\text{passLeftOverRight}}$ )
26:   goUp ~ Bernoulli( $\theta_{\text{passUp}}$ )
27:   if onLeftSide then
28:     return makeLeftPassing(SAMPLENEIGHBOR(goUp, leftParent))
29:   else
30:     return makeLeftPassing(SAMPLENEIGHBOR( $\neg$ goUp, rightParent))
  
```

Algorithm C.7 – Elaborating single notes.



```

1: function ELABLEFTNOTE(parent)
2:   children ← SAMPLESINGLEORNAMENTS(parent, elaborateL)
3:   return (parent, [makeLeftOrnament(child) for child ∈ children])

4: function ELABRIGHTNOTE(parent)
5:   children ← SAMPLESINGLEORNAMENTS(parent, elaborateR)
6:   return (parent, [makeRightOrnament(child) for child ∈ children])

7: function SAMPLESINGLEORNAMENTS(parent, paramElaborate)
8:   n ~ Geometric0( $\theta_{paramElaborate}$ )
9:   children ← permute n do
10:    repeat ~ Bernoulli( $\theta_{repeatOverNeighbor}$ )
11:    if repeat then
12:      oct ← SAMPLEOCTAVE( )
13:      yield makeRepeat(parent + oct)
14:    else
15:      up ~ Bernoulli(0.5)
16:      child ← SAMPLENEIGHBOR(up, parent)
17:      yield makeNeighbor(child)
18:   return children

19: function SAMPLENEIGHBOR(goUp, parent)
20:   chromatic ~ Bernoulli( $\theta_{nbChromatic}$ )
21:   octaves ← SAMPLEOCTAVE( )
22:   nAlteration ~ Geometric0( $\theta_{nbAlt}$ )
23:   if chromatic then
24:     return parent + octaves + makeAlteration(goUp, nAlteration)
25:   else
26:     altUp ~ Bernoulli(0.5)
27:     step ← makeStep(stepUp, altUp, nAlteration)
28:     return parent + octaves + step
  
```


C.3 Additional Data

Prelude in C major

J. S. Bach, BWV 939

The musical score for the Prelude in C major by J.S. Bach, BWV 939, is presented in a two-staff format. The piece is in C major and 3/4 time. The score is divided into four systems of four measures each. The first system (measures 1-4) begins with a treble staff containing a melodic line and a bass staff with a simple accompaniment. The second system (measures 5-8) continues the melody and accompaniment. The third system (measures 9-12) features a more complex melodic line in the treble staff. The fourth system (measures 13-16) concludes the piece with a final cadence. The score includes various musical notations such as notes, rests, accidentals, and phrasing slurs.

Figure C.1 – Prelude in C major by J. S. Bach (BWV 939).

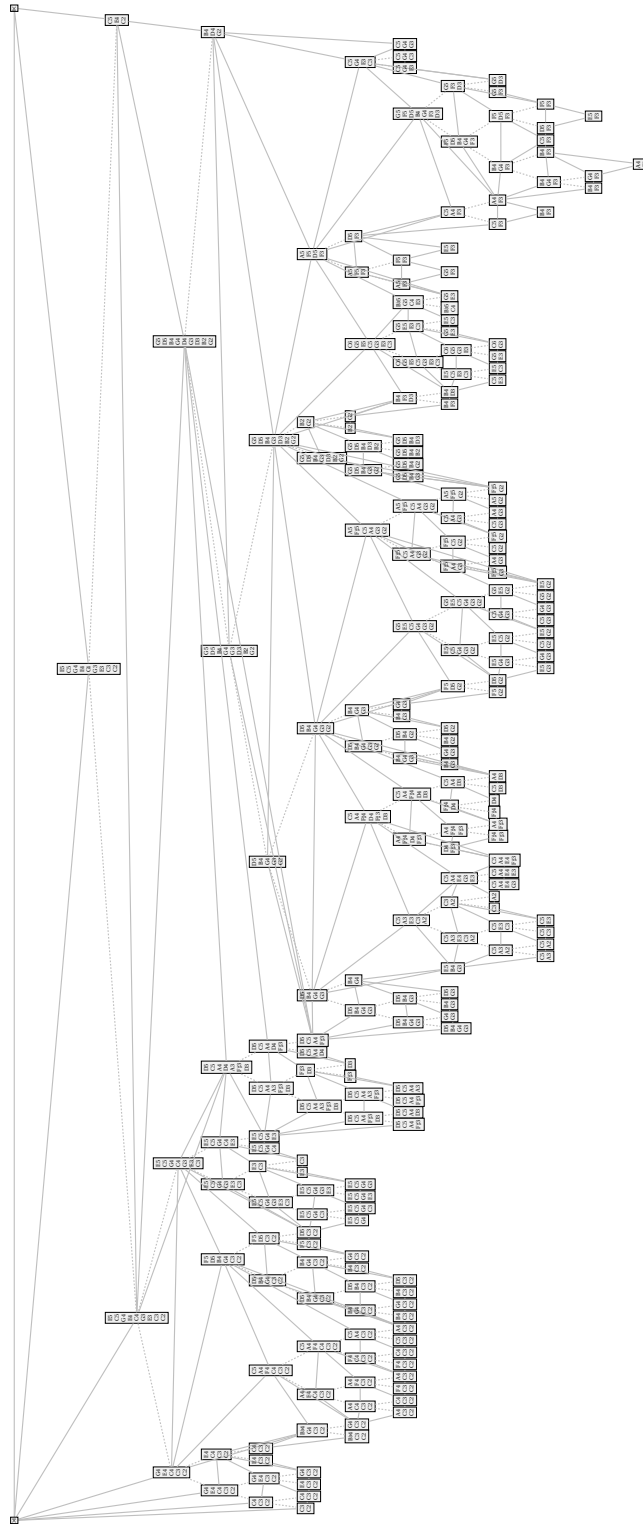


Figure C.2 – Analysis of Bach's prelude in C major (BWV 939).

Prelude in D minor

J. S. Bach, BWV 940

The musical score for the Prelude in D minor by J.S. Bach, BWV 940, is presented in four systems. Each system consists of two staves: a treble staff and a bass staff. The key signature is D minor (two flats) and the time signature is 3/4. The piece begins with a treble staff containing a melodic line and a bass staff with a steady eighth-note accompaniment. The first system (measures 1-3) shows the initial melodic phrase. The second system (measures 4-6) continues the melodic development. The third system (measures 7-9) features a more complex melodic line with some grace notes. The fourth system (measures 10-12) concludes the piece with a final cadence in the treble and a sustained bass line.

Figure C.3 – Prelude in D minor by J. S. Bach (BWV 940).

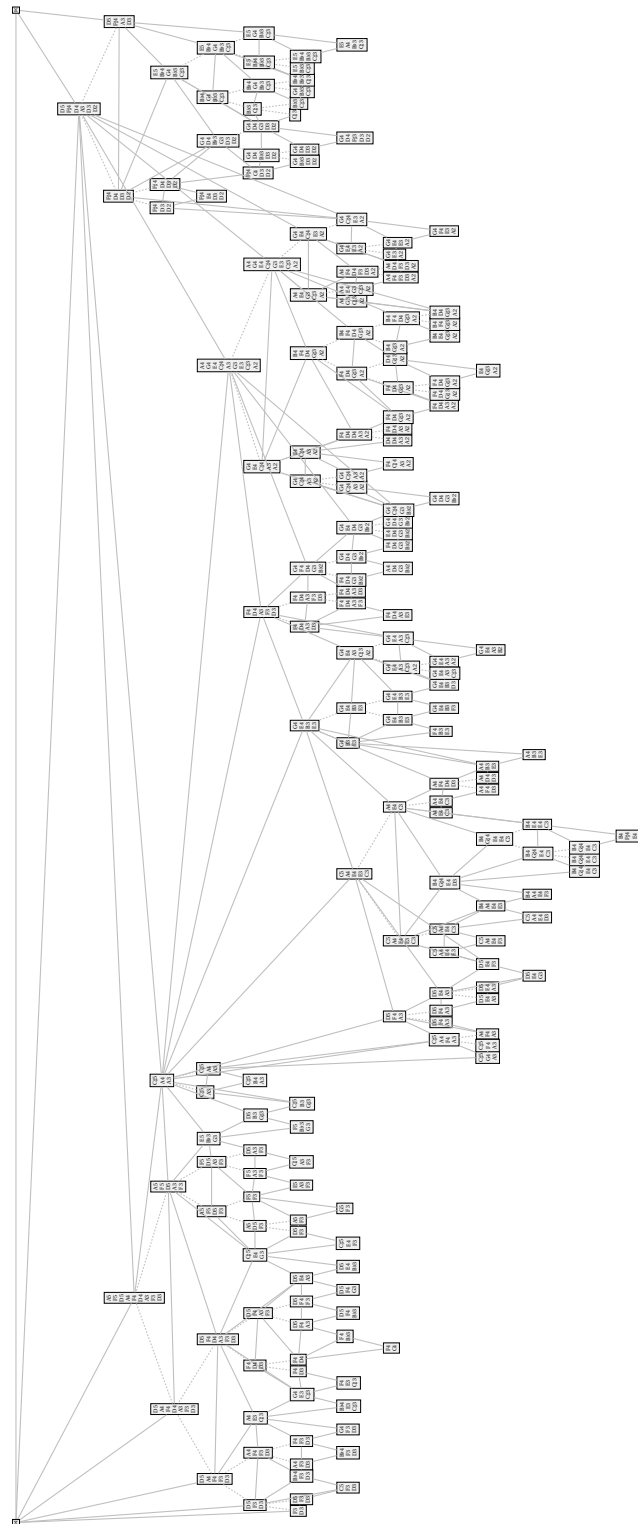


Figure C.4 – Analysis of Bach's prelude in D minor (BWV 940).

Bibliography

- Abdallah, S. A. and N. E. Gold (2014). “Comparing Models of Symbolic Music Using Probabilistic Grammars and Probabilistic Programming”. In: Joint Sound and Music Computing Conference and International Computer Music Conference. Athens, Greece.
- Abdallah, S. A., N. E. Gold, and A. Marsden (2016). “Analysing Symbolic Music with Probabilistic Grammars”. In: *Computational Music Analysis*. Ed. by D. Meredith. Cham: Springer International Publishing, pp. 157–189. DOI: 10.1007/978-3-319-25931-4_7.
- Albrecht, J. and D. Shanahan (Sept. 1, 2013). “The Use of Large Corpora to Train a New Type of Key-Finding Algorithm An Improved Treatment of the Minor Mode”. In: *Music Perception* 31.1, pp. 59–67. DOI: 10.1525/mp.2013.31.1.59.
- Albrecht, J. D. and D. Huron (Feb. 1, 2014). “A Statistical Approach to Tracing the Historical Development of Major and Minor Pitch Distributions, 1400-1750”. In: *Music Perception* 31.3, pp. 223–243. DOI: 10.1525/mp.2014.31.3.223.
- Aldwell, E. and A. Cadwallader (2018). *Harmony and Voice Leading*. Cengage Learning. 722 pp. Google Books: T69EDwAAQBAJ.
- Aldwell, E., C. Schachter, and A. Cadwallader (2011). *Harmony & Voice Leading*. 4th ed. Boston, MA: Schirmer/Cengage Learning. 720 pp.
- Allan, M. and C. K. I. Williams (2004). “Harmonising Chorales by Probabilistic Inference”. In: *Proceedings of the 17th International Conference on Neural Information Processing Systems*. NIPS’04. Cambridge, MA, USA: MIT Press, pp. 25–32.
- Arthur, C. (Apr. 1, 2017). “Taking Harmony Into Account”. In: *Music Perception* 34.4, pp. 405–423. DOI: 10.1525/mp.2017.34.4.405.
- Baroni, M., S. Maguire, and W. Drabkin (1983). “The Concept of Musical Grammar”. In: *Music Analysis* 2.2, pp. 175–208. DOI: 10.2307/854248. JSTOR: 854248.
- Bellmann, H. (2006). “About the Determination of Key of a Musical Excerpt”. In: *Computer Music Modeling and Retrieval*. Ed. by R. Kronland-Martinet, T. Voinier, and S. Ystad. Berlin, Heidelberg: Springer, pp. 76–91. DOI: 10.1007/11751069_7.
- Benjamin, W. E. (1981). “Pitch-Class Counterpoint in Tonal Music”. In: *Music Theory: Special Topics*, pp. 1–32.
- Bernstein, L. (1976). *The Unanswered Question: Six Talks at Harvard*. The Charles Eliot Norton Lectures. Cambridge, MA: Harvard University Press.

Bibliography

- Bigo, L. et al. (Sept. 23–27, 2018). “Relevance of Musical Features for Cadence Detection”. In: *Proceedings of the 19th International Society for Music Information Retrieval Conference*. 19th International Society for Music Information Retrieval Conference. Ed. by E. Gómez et al. Paris, France: ISMIR, pp. 355–361.
- Bingham, E. et al. (Jan. 1, 2019). “Pyro: Deep Universal Probabilistic Programming”. In: *The Journal of Machine Learning Research* 20.1, pp. 973–978.
- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.
- Blei, D. M., A. Kucukelbir, and J. D. McAuliffe (Apr. 3, 2017). “Variational Inference: A Review for Statisticians”. In: *Journal of the American Statistical Association* 112.518, pp. 859–877. DOI: 10.1080/01621459.2017.1285773.
- Bod, R. (2001). “Probabilistic Grammars for Music”. In: *Belgian-Dutch Conference on Artificial Intelligence (BNAIC)*. Citeseer.
- Box, G. E. P. (Dec. 1, 1976). “Science and Statistics”. In: *Journal of the American Statistical Association* 71.356, pp. 791–799. DOI: 10.1080/01621459.1976.10480949.
- Bregman, A. S. (1990). *Auditory Scene Analysis: The Perceptual Organization of Sound*. MIT press.
- Brown, M. (2005). *Explaining Tonality: Schenkerian Theory and Beyond*. Eastman Studies in Music. Rochester, NY: University of Rochester Press. 293 pp.
- Budge, H. (1943). “A Study of Chord Frequencies: Based on the Music of Representative Composers of the Eighteenth and Nineteenth Centuries”. New York: Columbia University.
- Butler, D. (Apr. 1, 1989). “Describing the Perception of Tonality in Music: A Critique of the Tonal Hierarchy Theory and a Proposal for a Theory of Intervallic Rivalry”. In: *Music Perception* 6.3, pp. 219–241. DOI: 10.2307/40285588.
- Byros, V. (2009). “Towards an “Archaeology” of Hearing: Schemata and Eighteenth-Century Consciousness”. In: *Musica Humana* 1, pp. 235–306.
- Cadwallader, A. and D. Gagné (Mar. 24, 2011). *Analysis of Tonal Music: A Schenkerian Approach*. Third Edition. Oxford, New York: Oxford University Press. 432 pp.
- Cambouropoulos, E. (2006). “‘Voice’ Separation: Theoretical, Perceptual and Computational Perspectives”. In: *Int. Conf. on Music Perception and Cognition (ICMPC)*, p. 12.
- (Sept. 1, 2008). “Voice And Stream: Perceptual And Computational Modeling Of Voice Separation”. In: *Music Perception* 26.1, pp. 75–94. DOI: 10.1525/mp.2008.26.1.75.
- (Sept. 1, 2010). “The Musical Surface: Challenging Basic Assumptions”. In: *Musicae Scientiae* 14 (2_suppl), pp. 131–147. DOI: 10.1177/10298649100140S209.
- Cambouropoulos, E., T. Crawford, and C. S. Iliopoulos (Feb. 1, 2001). “Pattern Processing in Melodic Sequences: Challenges, Caveats and Prospects”. In: *Computers and the Humanities* 35.1, pp. 9–21. DOI: 10.1023/A:1002646129893.
- Caplin, W. E. (2014). “Topics and Formal Functions - The Case of the Lament”. In: *The Oxford Handbook of Topic Theory*, p. 415.

- Carpenter, B. et al. (2017). “Stan: A Probabilistic Programming Language”. In: *Journal of statistical software* 76.1.
- Castellano, M. A., J. J. Bharucha, and C. L. Krumhansl (1984). “Tonal Hierarchies in the Music of North India”. In: *Journal of Experimental Psychology: General* 113.3, pp. 394–412. DOI: 10.1037/0096-3445.113.3.394.
- Cecchetti, G., S. A. Herff, C. Finkensiep, D. Harasim, et al. (in press). “Hearing Functional Harmony in Jazz: A Perceptual Study on Music-Theoretical Accounts of Extended Tonality.” In: *Musicae Scientiae*.
- Cecchetti, G., S. A. Herff, C. Finkensiep, and M. Rohrmeier (2020). “The Experience of Musical Structure as Computation : What Can We Learn?” In: *Rivista di analisi e teoria musicale* XXVI, 2, 2020, pp. 91–127. DOI: 10.53152/1032.
- Chater, N., M. W. Crocker, and M. Pickering (Nov. 19, 1998). “The Rational of Analysis of Inquiry: The Case of Parsing.” In: *Rational Models of Cognition*, pp. 441–469.
- Chater, N. (1996). “Reconciling Simplicity and Likelihood Principles in Perceptual Organization”. In: *Psychological Review* 103.3, pp. 566–581. DOI: 10.1037/0033-295X.103.3.566.
- Chater, N. and C. D. Manning (July 1, 2006). “Probabilistic Models of Language Processing and Acquisition”. In: *Trends in Cognitive Sciences*. Special Issue: Probabilistic Models of Cognition 10.7, pp. 335–344. DOI: 10.1016/j.tics.2006.05.006.
- Chater, N., M. Oaksford, et al. (2010). “Bayesian Models of Cognition”. In: *WIREs Cognitive Science* 1.6, pp. 811–823. DOI: 10.1002/wcs.79.
- Chater, N., J. B. Tenenbaum, and A. Yuille (July 1, 2006). “Probabilistic Models of Cognition: Conceptual Foundations”. In: *Trends in Cognitive Sciences*. Special Issue: Probabilistic Models of Cognition 10.7, pp. 287–291. DOI: 10.1016/j.tics.2006.05.007.
- Chew, E. and X. Wu (2005). “Separating Voices in Polyphonic Music: A Contig Mapping Approach”. In: *Computer Music Modeling and Retrieval*. Ed. by U. K. Wil. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 1–20. DOI: 10.1007/978-3-540-31807-1_1.
- Chi, W. et al. (Oct. 11, 2020). “Generating Music with a Self-Correcting Non-Chronological Autoregressive Model”. In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. International Society for Music Information Retrieval Conference (ISMIR 2020). Montreal, Canada: ISMIR, pp. 893–900. DOI: 10.5281/zenodo.4245578.
- Chomsky, N. (Sept. 1956). “Three Models for the Description of Language”. In: *IRE Transactions on Information Theory* 2.3, pp. 113–124. DOI: 10.1109/TIT.1956.1056813.
- Chomsky, N. (1963). “Formal Properties of Grammars”. In: *Handbook of Mathematical Psychology*. Vol. 2, pp. 328–418.
- (1965). *Aspects of the Theory of Syntax Cambridge*. Cambridge, MA, USA: MIT Press.
- Chuan, C.-H. and E. Chew (2011). “Generating and Evaluating Musical Harmonizations That Emulate Style”. In: *Computer Music Journal* 35.4, pp. 64–82. JSTOR: 41412947.

Bibliography

- Clark, A. (June 2013). "Whatever next? Predictive Brains, Situated Agents, and the Future of Cognitive Science". In: *Behavioral and Brain Sciences* 36.3, pp. 181–204. DOI: 10.1017/S0140525X12000477.
- Clarke, D. (Oct. 1, 2017). "North Indian Classical Music and Lerdahl and Jackendoff's Generative Theory – a Mutual Regard". In: *Music Theory Online* 23.3.
- Cohn, R. (Feb. 16, 2012). *Audacious Euphony: Chromatic Harmony and the Triad's Second Nature*. Oxford Studies in Music Theory. Oxford, New York: Oxford University Press. 256 pp.
- Colombo, F. and W. Gerstner (Feb. 20, 2018). "BachProp: Learning to Compose Music in Multiple Styles". arXiv: 1802.05162 [cs, eess].
- Conklin, D. and M. Bergeron (2010). "Discovery of Contrapuntal Patterns". In: *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010, Utrecht, Netherlands, August 9-13, 2010*. Ed. by J. S. Downie and R. C. Veltkamp. International Society for Music Information Retrieval, pp. 201–206.
- Conklin, D. and I. H. Witten (Mar. 1, 1995). "Multiple Viewpoint Systems for Music Prediction". In: *Journal of New Music Research* 24.1, pp. 51–73. DOI: 10.1080/09298219508570672.
- Cox, R. T. (1946). "Probability, Frequency and Reasonable Expectation". In: *American Journal of Physics* 14.1, pp. 1–13.
- Cusumano-Towner, M. F. et al. (2019). "Gen: A General-Purpose Probabilistic Programming System with Programmable Inference". In: *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 221–236.
- Davis, S. (Apr. 1, 2011). "Stream Segregation and Perceived Syncopation: Analyzing the Rhythmic Effects of Implied Polyphony in Bach's Unaccompanied String Works". In: *Music Theory Online* 17.1.
- De Clercq, T. and D. Temperley (Jan. 2011). "A Corpus Analysis of Rock Harmony". In: *Popular Music* 30.1, pp. 47–70. DOI: 10.1017/S026114301000067X.
- De Haas, W. B. et al. (Oct. 26–30, 2009). "Modeling Harmonic Similarity Using a Generative Grammar of Tonal Harmony". In: *Proceedings of the 10th International Society on Music Information Retrieval Conference*. 10th International Society on Music Information Retrieval Conference (ISMIR 2009). Kobe, Japan: ISMIR, p. 6.
- Deliege, I. (July 1, 1987). "Grouping Conditions in Listening to Music: An Approach to Lerdahl & Jackendoff's Grouping Preference Rules". In: *Music Perception* 4.4, pp. 325–359. DOI: 10.2307/40285378.
- Demirel, E., B. Bozkurt, and X. Serra (2019). "Automatic Chord-Scale Recognition Using Harmonic Pitch Class Profiles". In: *Proceedings of the 16th Sound & Music Computing Conference; 2019 May 28-31; Málaga, Spain.[Málaga]*. SMC. Málaga, p. 8.

- Deutsch, D. (Jan. 1, 1999). “9 - Grouping Mechanisms in Music”. In: *The Psychology of Music (Second Edition)*. Ed. by D. Deutsch. Cognition and Perception. San Diego: Academic Press, pp. 299–348. DOI: 10.1016/B978-012213564-4/50010-X.
- Deutsch, D. and J. Feroe (1981). “The Internal Representation of Pitch Sequences in Tonal Music”. In: *Psychological Review* 88.6, pp. 503–522. DOI: 10.1037/0033-295X.88.6.503.
- De Valk, R. and T. Weyde (May 25, 2018). “Deep Neural Networks with Voice Entry Estimation Heuristics for Voice Separation in Symbolic Music Representations”. In: *Proceedings of the 19th International Society for Music Information Retrieval Conference*. 19th International Society for Music Information Retrieval Conference (ISMIR 2018). Paris, France.
- Dharambir Singh (June 27, 2019). *Selected Alap Phrases in Raga Multani*. In collab. with D. Singh. DOI: 10.5281/zenodo.3258808.
- Dienes, Z. (Feb. 28, 2008). *Understanding Psychology as a Science: An Introduction to Scientific and Statistical Inference*. Macmillan International Higher Education. 185 pp. Google Books: qCQdBQAAQBAJ.
- Dowling, W. J. (July 1978). “Scale and Contour: Two Components of a Theory of Memory for Melodies”. In: *Psychological Review* 85.4, pp. 341–354. DOI: <http://dx.doi.org/10.1037/0033-295X.85.4.341>.
- Duane, B. (May 27, 2019). “Melodic Patterns and Tonal Cadences: Bayesian Learning of Cadential Categories from Contrapuntal Information”. In: *Journal of New Music Research* 48.3, pp. 197–216. DOI: 10.1080/09298215.2019.1607396.
- Duane, B. and B. Pardo (2009). “Streaming from MIDI Using Constraint Satisfaction Optimization and Sequence Alignment.” In: *ICMC*. Citeseer.
- Ebcioğlu, K. (1979). “Strict Counterpoint: A Case Study in Musical Composition by Computers”. In.
- (1988). “An Expert System for Harmonizing Four-Part Chorales”. In: *Computer Music Journal* 12.3, pp. 43–51. DOI: 10.2307/3680335. JSTOR: 3680335.
- Farbood, M. and B. Schoner (2001). “Analysis and Synthesis of Palestrina-style Counterpoint Using Markov Chains”. In: *Proceedings of the 2001 International Computer Music Conference*. Vol. 2. Havana, Cuba, pp. 471–474.
- Feldman, J. (2009). “Bayes and the Simplicity Principle in Perception”. In: *Psychological Review* 116.4, pp. 875–887. DOI: 10.1037/a0017144.
- Felip, J. et al. (2019). “Real-Time Approximate Inference for Scene Understanding with Generative Models”. In: *Workshop on Perceptions as Generative Reasoning (co-located with NeurIPS)*.
- Finkensiep, C., K. Déguernel, et al. (2020). “Voice-Leading Schema Recognition Using Rhythm and Pitch Features”. In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. ISMIR. Montreal, Canada (online), pp. 520–526. DOI: 10.5281/zenodo.4245482.

Bibliography

- Finkensiep, C., P. Ericson, et al. (in preparation). “Chord Types and Ornamentation – A Bayesian Model of Extended Chord Profiles”. In: *Open Research Europe*.
- Finkensiep, C., M. Neuwirth, and M. Rohrmeier (2018). “Generalized Skipgrams for Pattern Discovery in Polyphonic Streams”. In: *Proceedings of the 19th International Society for Music Information Retrieval Conference*. ISMIR. Paris, France, pp. 547–553. DOI: 10.5281/zenodo.1492473.
- (submitted). “Music Theory and Model-Driven Corpus Research”. In: *Oxford Handbook of Corpus Studies in Music*. Oxford: Oxford University Press.
- Finkensiep, C. and M. Rohrmeier (2021). “Modeling and Inferring Proto-Voice Structure in Free Polyphony”. In: *Proceedings of the 22nd International Society for Music Information Retrieval Conference*. ISMIR. Online, pp. 189–196. DOI: 10.5281/zenodo.5624431.
- Finkensiep, C., R. Widdess, and M. Rohrmeier (2019). “Modelling the Syntax of North Indian Melodies with a Generalized Graph Grammar”. In: *Proceedings of the 20th International Society for Music Information Retrieval Conference*. ISMIR (Delft, The Netherlands). Delft, The Netherlands, pp. 462–469. DOI: 10.5281/zenodo.3527844.
- Fitch, W. T. and M. D. Martins (2014). “Hierarchical Processing in Music, Language, and Action: Lashley Revisited”. In: *Annals of the New York Academy of Sciences* 1316.1, pp. 87–104. DOI: 10.1111/nyas.12406.
- Flanders, J. and F. Jannidis, eds. (May 31, 2018). *The Shape of Data in the Digital Humanities: Modeling Texts and Text-based Resources*. London: Routledge. 360 pp. DOI: 10.4324/9781315552941.
- Forte, A. (Apr. 1, 1967). “Syntax-Based Analytic Reading of Musical Scores”. In.
- (1979). *Tonal Harmony in Concept and Practice*. 3d ed. New York: Holt, Rinehart and Winston. 564 pp.
- Foscarin, E., F. Jacquemard, and P. Rigaux (May 28, 2019). “Modeling and Learning Rhythm Structure”. In: Sound and Music Computing Conference (SMC).
- Frankel, R. E., S. J. Rosenschein, and S. W. Smoliar (Mar. 1, 1978). “Schenker’s Theory of Tonal Music—Its Explication through Computational Processes”. In: *International Journal of Man-Machine Studies* 10.2, pp. 121–138. DOI: 10.1016/S0020-7373(78)80008-X.
- Fujishima, T. (1999). “Real-Time Chord Recognition of Musical Sound: A System Using Common Lisp Music”. In: *Proc. ICMC, Oct. 1999*, pp. 464–467.
- Gauldin, R. (1997). *Harmonic Practice in Tonal Music*. New York: W. W. Norton.
- Ge, H., K. Xu, and Z. Ghahramani (Mar. 31, 2018). “Turing: A Language for Flexible Probabilistic Inference”. In: *International Conference on Artificial Intelligence and Statistics*. International Conference on Artificial Intelligence and Statistics. PMLR, pp. 1682–1690.

- Gedik, A. C. and B. Bozkurt (Apr. 2010). "Pitch-Frequency Histogram-Based Music Information Retrieval for Turkish Music". In: *Signal Processing* 90.4, pp. 1049–1063. DOI: 10.1016/j.sigpro.2009.06.017.
- Geman, S. and D. Geman (Nov. 1984). "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6.6, pp. 721–741. DOI: 10.1109/TPAMI.1984.4767596.
- Gerstner, W. et al. (2018). "Eligibility Traces and Plasticity on Behavioral Time Scales: Experimental Support of NeoHebbian Three-Factor Learning Rules". In: *Frontiers in Neural Circuits* 12.
- Gilbert, É. and D. Conklin (2007). "A Probabilistic Context-Free Grammar for Melodic Reduction". In: *International Workshop on Artificial Intelligence and Music, IJCAI-07*.
- Giraud, M., K. Déguernel, and E. Cambouropoulos (2014). "Fragmentations with Pitch, Rhythm and Parallelism Constraints for Variation Matching". In: *Sound, Music, and Motion*. Ed. by M. Aramaki et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 298–312. DOI: 10.1007/978-3-319-12976-1_19.
- Giraud, M., R. Groult, and E. Leguy (May 24, 2018). "Dezrann, a Web Framework to Share Music Analysis". In: International Conference on Technologies for Music Notation and Representation (TENOR 2018), p. 104.
- Gjerdingen, R. (Oct. 11, 2007). *Music in the Galant Style*. Oxford, New York: Oxford University Press. 528 pp.
- Gjerdingen, R. and J. Bourne (June 1, 2015). "Schema Theory as a Construction Grammar". In: *Music Theory Online* 21.2.
- Gjerdingen, R. O. (1988). *A Classic Turn of Phrase: Music and the Psychology of Convention*. Studies in the Criticism and Theory of Music. Philadelphia: University of Pennsylvania Press. 299 pp.
- Gold, B. P. et al. (Nov. 20, 2019). "Predictability and Uncertainty in the Pleasure of Music: A Reward for Learning?" In: *Journal of Neuroscience* 39.47, pp. 9397–9409. DOI: 10.1523/JNEUROSCI.0428-19.2019. PMID: 31636112.
- Gómez-Corral, A. et al. (2015). "Bayesian Inference of Markov Processes". In: *Wiley StatsRef: Statistics Reference Online*. American Cancer Society, pp. 1–15. DOI: 10.1002/9781118445112.stat07837.
- Goodfellow, I., Y. Bengio, and A. Courville (Nov. 18, 2016). *Deep Learning*. Red. by F. Bach. Adaptive Computation and Machine Learning Series. Cambridge, MA, USA: MIT Press. 800 pp.
- Goodman, J. (n.d.). "Semiring Parsing". In: *Computational Linguistics* 25.4 (), p. 34.
- Goodman, N. D., J. B. Tenenbaum, and T. P. Contributors (2016). "Probabilistic Models of Cognition". In:
- Goodman, N. D., V. K. Mansinghka, et al. (2008). "Church: A Language for Generative Models". In: *UAI 2008, Proceedings of the 24th Conference in Uncertainty in Arti-*

Bibliography

- cial Intelligence, Helsinki, Finland, July 9-12, 2008*. Ed. by D. A. McAllester and P. Myllymäki. AUAI Press, pp. 220–229.
- Goodman, N. D. and A. Stuhlmüller (2014). “The Design and Implementation of Probabilistic Programming Languages”. In.
- Goodman, N. D., T. D. Ullman, and J. B. Tenenbaum (2011). “Learning a Theory of Causality”. In: *Psychological Review* 118.1, pp. 110–119. DOI: 10.1037/a0021336.
- Gopnik, A. and E. Bonawitz (2015). “Bayesian Models of Child Development”. In: *WIREs Cognitive Science* 6.2, pp. 75–86. DOI: 10.1002/wcs.1330.
- Gotham, M., D. Tymoczko, and M. S. Cuthbert (2019). “The RomanText Format: A Flexible and Standard Method for Representing Roman Numerical Analyses”. In: *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019, Delft, The Netherlands, November 4-8, 2019*. Ed. by A. Flexer et al., pp. 123–129.
- Granroth-Wilding, M. and M. Steedman (Oct. 2, 2014). “A Robust Parser-Interpreter for Jazz Chord Sequences”. In: *Journal of New Music Research* 43.4, pp. 355–374. DOI: 10.1080/09298215.2014.910532.
- Graves, A. (2011). “Practical Variational Inference for Neural Networks”. In: *Advances in Neural Information Processing Systems*. Vol. 24. Curran Associates, Inc.
- Griffiths, T. L. and J. B. Tenenbaum (Sept. 2006). “Optimal Predictions in Everyday Cognition”. In: *Psychological Science* 17.9, pp. 767–773. DOI: 10.1111/j.1467-9280.2006.01780.x.
- Groves, R. (2016). “Towards the Generation of Melodic Structure”. In: *Proceedings of the Fourth International Workshop on Musical Metacreation*, pp. 1–8.
- Guiomard-Kagan, N. et al. (Oct. 2015). “Comparing Voice and Stream Segmentation Algorithms”. In: *Proceedings of the 16th ISMIR Conference*. International Society for Music Information Retrieval Conference (ISMIR 2015). Malaga, Spain, pp. 493–499.
- Haas, B. (2004). *Die Neue Tonalität von Schubert Bis Webern: Hören Und Analysieren Nach Albert Simon*. Veröffentlichungen Zur Musikforschung 21. Wilhelmshaven: F. Noetzel, Heinrichshofen Bücher. 99 pp.
- Hadjeres, G., F. Pachet, and F. Nielsen (Dec. 3, 2016). “DeepBach: A Steerable Model for Bach Chorales Generation”. arXiv: 1612.01010 [cs].
- Haimo, E. (Nov. 9, 1995). *Haydn's Symphonic Forms: Essays in Compositional Logic*. Oxford Monographs on Music. Oxford, New York: Oxford University Press. 310 pp.
- Hamanaka, M., K. Hirata, and S. Tojo (2016). “Implementing Methods for Analysing Music Based on Lerdahl and Jackendoff's Generative Theory of Tonal Music”. In: *Computational Music Analysis*. Ed. by D. Meredith. Cham: Springer International Publishing, pp. 221–249. DOI: 10.1007/978-3-319-25931-4_9.
- Harasim, D. (2020). “The Learnability of the Grammar of Jazz: Bayesian Inference of Hierarchical Structures in Harmony”. Lausanne: Ecole Polytechnique Fédérale de Lausanne.

- Harasim, D., C. Finkensiep, L. Bigo, et al. (2021). "Music Cognition: The Complexity of Musical Structure". In: *Proceedings of the Annual Meeting of the Cognitive Science Society*. CogSci. Vol. 43. Online.
- Harasim, D., C. Finkensiep, P. Ericson, et al. (2020). "The Jazz Harmony Treebank". In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. ISMIR. Montreal, Canada (online), pp. 207–215. DOI: 10.5281/zenodo.4245406.
- Harasim, D., F. C. Moss, et al. (Jan. 4, 2021). "Exploring the Foundations of Tonality: Statistical Cognitive Modeling of Modes in the History of Western Classical Music". In: *Humanities and Social Sciences Communications* 8.1 (1), pp. 1–11. DOI: 10.1057/s41599-020-00678-6.
- Harasim, D., T. Noll, and M. Rohrmeier (2019). "Distant Neighbors and Interscalar Contiguities". In: *Mathematics and Computation in Music*. Ed. by M. Montiel, F. Gomez-Martin, and O. A. Agustín-Aquino. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 172–184. DOI: 10.1007/978-3-030-21392-3_14.
- Harasim, D., T. J. O'Donnell, and M. Rohrmeier (2019). "Harmonic Syntax in Time: Rhythm Improves Grammatical Models of Harmony". In: *Proceedings of the 20th ISMIR Conference*. 20th International Society for Music Information Retrieval Conference. CONF. Delft, The Netherlands: ISMIR, p. 335. DOI: 10.5281/zenodo.3527812.
- Harasim, D., M. Rohrmeier, and T. J. O'Donnell (Sept. 23–27, 2018). "A Generalized Parsing Framework for Generative Models of Harmonic Syntax". In: *Proceedings of the 19th International Society for Music Information Retrieval Conference*. 19th International Society for Music Information Retrieval Conference, ISMIR 2018. Ed. by E. Gómez et al. Paris, France, pp. 152–159.
- Harasim, D., S. E. Schmidt, and M. Rohrmeier (Sept. 1, 2016). "Bridging Scale Theory and Geometrical Approaches to Harmony: The Voice-Leading Duality between Complementary Chords". In: *Journal of Mathematics and Music* 10.3, pp. 193–209. DOI: 10.1080/17459737.2016.1216186.
- Harrison, P. M. C. and M. T. Pearce (Feb. 1, 2020a). "A Computational Cognitive Model for the Analysis and Generation of Voice Leadings". In: *Music Perception* 37.3, pp. 208–224. DOI: 10.1525/mp.2020.37.3.208.
- (Jan. 9, 2020b). "Representing Harmony in Computational Music Cognition". In: DOI: 10.31234/osf.io/xswp4.
- Hastings, W. K. (Apr. 1, 1970). "Monte Carlo Sampling Methods Using Markov Chains and Their Applications". In: *Biometrika* 57.1, pp. 97–109. DOI: 10.1093/biomet/57.1.97.
- Hedges, T. and M. Rohrmeier (2011). "Exploring Rameau and Beyond: A Corpus Study of Root Progression Theories". In: *Mathematics and Computation in Music*. Ed. by C. Agon et al. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 334–337. DOI: 10.1007/978-3-642-21590-2_27.

Bibliography

- Helmholtz, H. von (1860). “Handbuch Der Physiologischen Optik [English Translation]”. In: *Dover: New York* 1962.
- Hentschel, J., F. C. Moss, et al. (Nov. 7–12, 2021). “A Semi-Automated Workflow Paradigm for the Distributed Creation and Curation of Expert Annotations”. In: *Proceedings of the 22nd International Society for Music Information Retrieval Conference*. 22nd International Society for Music Information Retrieval Conference, ISMIR 2021. Ed. by J. H. Lee et al. Online: ISMIR, pp. 262–269. DOI: 10.5281/zenodo.5624417.
- Hentschel, J., M. Neuwirth, and M. Rohrmeier (2021). “The Annotated Mozart Sonatas: Score, Harmony, and Cadence”. In: *Transactions of the International Society for Music Information Retrieval* 4.1, pp. 67–80. DOI: 10.5334/tismir.63.
- Herff, S. A. et al. (2021). “Hierarchical Syntactic Structure Predicts Listeners’ Sequence Completion in Music”. In: *Proceedings of the Annual Meeting of the Cognitive Science Society* 43.43.
- Herremans, D. and K. Sørensen (Nov. 15, 2013). “Composing Fifth Species Counterpoint Music with a Variable Neighborhood Search Algorithm”. In: *Expert Systems with Applications* 40.16, pp. 6427–6437. DOI: 10.1016/j.eswa.2013.05.071.
- Hoffman, M. D. and A. Gelman (2014). “The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo.” In: *Journal of Machine Learning Research* 15.1, pp. 1593–1623.
- Hu, D. and L. K. Saul (Oct. 26, 2009). “A Probabilistic Topic Model for Unsupervised Learning of Musical Key-Profiles.” *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)* (Kobe, Japan). DOI: 10.5281/zenodo.1415160.
- Hu, T. and C. Arthur (May 11, 2021). “A Statistical Model for Melody Reduction”. arXiv: 2105.05385 [cs, eess, stat].
- Huron, D. (Apr. 14, 2006). *Sweet Anticipation: Music and the Psychology of Expectation*. Cambridge, MA, USA: A Bradford Book. 480 pp.
- (Sept. 2, 2016). *Voice Leading: The Science Behind a Musical Art*. MIT Press. 273 pp. Google Books: zcrxDAAAQBAJ.
- Ijzerman, J. (Jan. 24, 2019). *Harmony, Counterpoint, Partimento: A New Method Inspired by Old Masters*. Oxford, New York: Oxford University Press. 375 pp.
- Jackendoff, R. (1977). “Review of The Unanswered Question”. In: *Language* 53.4. In collab. with L. Bernstein, pp. 883–894. DOI: 10.2307/412916. JSTOR: 412916.
- (Feb. 1, 2009). “Parallels and Nonparallels between Language and Music”. In: *Music Perception* 26.3, pp. 195–204. DOI: 10.1525/mp.2009.26.3.195.
- Jackendoff, R. and F. Lerdahl (May 1, 2006). “The Capacity for Music: What Is It, and What’s Special about It?” In: *Cognition*. The Nature of Music 100.1, pp. 33–72. DOI: 10.1016/j.cognition.2005.11.005.
- Jacobs, R. A. and J. K. Kruschke (2011). “Bayesian Learning Theory Applied to Human Cognition”. In: *WIREs Cognitive Science* 2.1, pp. 8–21. DOI: 10.1002/wcs.80.

- Jan, S. (June 1, 2013). "Using Galant Schemata as Evidence for Universal Darwinism". In: *Interdisciplinary Science Reviews* 38.2, pp. 149–168. DOI: 10.1179/0308018813Z.00000000042.
- Janssen, B. et al. (2014). "Finding Repeated Patterns in Music: State of Knowledge, Challenges, Perspectives". In: *Sound, Music, and Motion*. Ed. by M. Aramaki et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, pp. 277–297. DOI: 10.1007/978-3-319-12976-1_18.
- Jarvinen, T. (July 1, 1995). "Tonal Hierarchies in Jazz Improvisation". In: *Music Perception* 12.4, pp. 415–437. DOI: 10.2307/40285675.
- Jaynes, E. T. (1988). "How Does the Brain Do Plausible Reasoning?" In: *Maximum-Entropy and Bayesian Methods in Science and Engineering: Foundations*. Ed. by G. J. Erickson and C. R. Smith. Fundamental Theories of Physics. Dordrecht: Springer Netherlands, pp. 1–24. DOI: 10.1007/978-94-009-3049-0_1.
- (Apr. 2003). *Probability Theory: The Logic of Science*. Ed. by G. L. Bretthorst. Cambridge University Press. DOI: 10.1017/CB09780511790423.
- Jeppesen, K. (1946). *The Style of Palestrina and the Dissonance*. 2nd Edition. Oxford: Oxford University Press.
- (1960). *Counterpoint: The Polyphonic Vocal Style of the Sixteenth Century*. Trans. by G. Haydon. Englewood Cliffs, N. J.: Prentice-Hall.
- Jones, M. R. (Nov. 1, 1987). "Dynamic Pattern Structure in Music: Recent Theory and Research". In: *Perception & Psychophysics* 41.6, pp. 621–634. DOI: 10.3758/BF03210494.
- Ju, Y. et al. (Oct. 28, 2017). "Non-Chord Tone Identification Using Deep Neural Networks". In: *Proceedings of the 4th International Workshop on Digital Libraries for Musicology*. DLfM '17. New York, NY, USA: Association for Computing Machinery, pp. 13–16. DOI: 10.1145/3144749.3144753.
- Kass, R. E. and A. E. Raftery (June 1, 1995). "Bayes Factors". In: *Journal of the American Statistical Association* 90.430, pp. 773–795. DOI: 10.1080/01621459.1995.10476572.
- Katsivalos, A., T. Collins, and B. Battey (Nov. 4, 2019). "An Initial Computational Model for Musical Schemata Theory". In: *Proceedings of the 20th International Society for Music Information Retrieval Conference*. International Society for Music Information Retrieval Conference (ISMIR 2019). Delft, The Netherlands: ISMIR, pp. 166–172. DOI: 10.5281/zenodo.3527768.
- Katz, J. and D. Pesetsky (2011). "The Identity Thesis for Language and Music". In: *Manuscript*.
- Kersten, D., P. Mamassian, and A. Yuille (Feb. 1, 2004). "Object Perception as Bayesian Inference". In: *Annual Review of Psychology* 55.1, pp. 271–304. DOI: 10.1146/annurev.psych.55.090902.142005.
- Kilian, J. and H. H. Hoos (2002). "Voice Separation-A Local Optimization Approach." In: *International Conference on Music Information Retrieval*.

Bibliography

- King, G. and L. Zeng (2001). “Logistic Regression in Rare Events Data”. In: *Political Analysis* 9.2, pp. 137–163. DOI: 10.1093/oxfordjournals.pan.a004868.
- Kirlin, P. B. and P. E. Utgoff (2008). “A Framework for Automated Schenkerian Analysis”. In: p. 6.
- Kirlin, P. B. and D. D. Jensen (2011). “Probabilistic Modeling of Hierarchical Music Analysis.” In: *Proceedings of the 12th International Society for Music Information Retrieval Conference*. 12th International Society for Music Information Retrieval Conference (ISMIR 2011), pp. 393–398.
- Kirlin, P. B. and D. L. Thomas (2015). “Extending a Model of Monophonic Hierarchical Music Analysis to Homophony”. In: *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015, Málaga, Spain, October 26-30, 2015*. Ed. by M. Müller and F. Wiering, pp. 715–721.
- Kirlin, P. B. and P. E. Utgoff (2005). “VOISE: Learning to Segregate Voices in Explicit and Implicit Polyphony.” In: *Proceedings of the Sixth International Conference on Music Information Retrieval*. ISMIR. London, pp. 552–557.
- Kirlin, P. B. and J. Yust (May 3, 2016). “Analysis of Analysis: Using Machine Learning to Evaluate the Importance of Music Parameters for Schenkerian Analysis”. In: *Journal of Mathematics and Music* 10.2, pp. 127–148. DOI: 10.1080/17459737.2016.1209588.
- Knill, D. C., D. Kersten, and A. Yuille (1996). “Introduction: A Bayesian Formulation of Visual Perception”. In: *Perception as Bayesian Inference*. Cambridge: Cambridge University Press, pp. 1–21.
- Koch, H. C. (1782, 1787, 1793). *Versuch einer Anleitung zur Composition*. 3 vols. Leipzig: Böhme. Google Books: X0VDAAAAcAAJ.
- Koelsch, S., P. Vuust, and K. Friston (Jan. 1, 2019). “Predictive Processes and the Peculiar Case of Music”. In: *Trends in Cognitive Sciences* 23.1, pp. 63–77. DOI: 10.1016/j.tics.2018.10.006.
- Koops, H. V. et al. (Feb. 1, 2020). “Automatic Chord Label Personalization through Deep Learning of Shared Harmonic Interval Profiles”. In: *Neural Computing and Applications* 32.4, pp. 929–939. DOI: 10.1007/s00521-018-3703-y.
- Krumhansl, C. L. (July 1, 1979). “The Psychological Representation of Musical Pitch in a Tonal Context”. In: *Cognitive Psychology* 11.3, pp. 346–374. DOI: 10.1016/0010-0285(79)90016-1.
- (1990a). *Cognitive Foundations of Musical Pitch*. Oxford University Press. 318 pp. Google Books: J4dJCAAQAQBAJ.
- (Apr. 1, 1990b). “Tonal Hierarchies and Rare Intervals in Music Cognition”. In: *Music Perception* 7.3, pp. 309–324. DOI: 10.2307/40285467.
- Krumhansl, C. L. and E. J. Kessler (Jan. 1, 1982). “Tracing the Dynamic Changes in Perceived Tonal Organization in a Spatial Representation of Musical Keys.” In: *Psychological Review* 89.4, p. 334. DOI: 10.1037/0033-295X.89.4.334.

- Kucukelbir, A. et al. (Jan. 1, 2017). "Automatic Differentiation Variational Inference". In: *The Journal of Machine Learning Research* 18.1, pp. 430–474.
- Kulkarni, T. D. et al. (2015). "Picture: A Probabilistic Programming Language for Scene Perception". In: *Kulkarni, Tejas Dattatraya*.
- Laitz, S. G. (2016). *The Complete Musician: An Integrated Approach to Theory, Analysis, and Listening*. Fourth edition. New York: Oxford University Press. 875 pp.
- Lartillot, O. (2008). "Motivic Pattern Extraction in Symbolic Domain:" in: *Intelligent Music Information Systems*. Ed. by J. Shen et al. IGI Global, pp. 236–260. DOI: 10.4018/978-1-59904-663-1.ch011.
- (2016). "Automated Motivic Analysis: An Exhaustive Approach Based on Closed and Cyclic Pattern Mining in Multidimensional Parametric Spaces". In: *Computational Music Analysis*. Springer, Cham, pp. 273–302. DOI: 10.1007/978-3-319-25931-4_11.
- Lee, K. (2006). "Automatic Chord Recognition from Audio Using Enhanced Pitch Class Profile". In: *ICMC*.
- Lee, T. S. and D. Mumford (July 1, 2003). "Hierarchical Bayesian Inference in the Visual Cortex". In: *JOSA A* 20.7, pp. 1434–1448. DOI: 10.1364/JOSAA.20.001434.
- Lerdahl, F. and R. Jackendoff (1983). *A Generative Theory of Tonal Music*. MIT press.
- Levine, M. (Jan. 12, 2011). *The Jazz Theory Book*. "O'Reilly Media, Inc." 786 pp. Google Books: iyNQpJ4oaMcC.
- Longuet-Higgins, H. C. (Oct. 1976). "Perception of Melodies". In: *Nature* 263.5579 (5579), pp. 646–653. DOI: 10.1038/263646a0.
- Longuet-Higgins, H. C. (July 1983). "All in Theory — the Analysis of Music". In: *Nature* 304.5921 (5921), pp. 93–93. DOI: 10.1038/304093a0.
- Longuet-Higgins, H. C. and M. Steedman (1971). "On Interpreting Bach". In: *Machine Intelligence* 6, pp. 221–241.
- MacKay, D. J. C. (Sept. 25, 2003). *Information Theory, Inference and Learning Algorithms*. Cambridge University Press. 694 pp. Google Books: AKuMj4PN_EMC.
- Makris, D., I. Karydis, and E. Cambouropoulos (2016). "VISA3: Refining the Voice Integration/Segregation Algorithm". In: *Proceedings of the Sound and Music Computing Conference 2016*. Hamburg, Germany.
- Manning, C. D. and H. Schütze (May 28, 1999). *Foundations of Statistical Natural Language Processing*. Cambridge, MA, USA: MIT Press. 718 pp.
- Marr, D. (1982). *Vision*. San Francisco: W. H. Freeman.
- Marsden, A. (Feb. 1, 2001). "Representing Melodic Patterns as Networks of Elaborations". In: *Computers and the Humanities* 35.1, pp. 37–54. DOI: 10.1023/A:1002705506386.
- (Dec. 1, 2005). "Generative Structural Representation of Tonal Music". In: *Journal of New Music Research* 34.4, pp. 409–428. DOI: 10.1080/09298210600578295.

Bibliography

- Marsden, A. (Sept. 23, 2007). "Automatic Derivation of Musical Structure: A Tool for Research on Schenkerian Analysis." In: *International Conference on Music Information Retrieval*. International Conference on Music Information Retrieval. Vienna, pp. 55–58.
- (Sept. 1, 2010). "Schenkerian Analysis by Computer: A Proof of Concept". In: *Journal of New Music Research* 39.3, pp. 269–289. DOI: 10.1080/09298215.2010.503898.
- Martin, H. (1978). "Review of Beyond Schenkerism". In: *Perspectives of New Music* 17.1. In collab. with E. Narmour, pp. 196–210. DOI: 10.2307/832664. JSTOR: 832664.
- Mavromatis, P. (Nov. 1, 2009). "Minimum Description Length Modelling of Musical Structure". In: *Journal of Mathematics and Music* 3.3, pp. 117–136. DOI: 10.1080/17459730903313122.
- Mavromatis, P. and M. Brown (2004). "Parsing Context-Free Grammars for Music: A Computational Model of Schenkerian Analysis". In: *Proceedings of the 8th International Conference on Music Perception & Cognition*. Citeseer, pp. 414–415.
- McAdams, S. (June 1, 2004). "Musical Forces and Melodic Expectations: Comparing Computer Models and Experimental Results". In: *Music Perception: An Interdisciplinary Journal* 21.4, pp. 457–498. DOI: 10.1525/mp.2004.21.4.457.
- McAdams, S. and A. S. Bregman (1979). "Hearing Musical Streams". In: *Computer Music Journal* 3.4, pp. 26–60. JSTOR: 4617866.
- McHose, A. I. (1947). *The Contrapuntal Harmonic Technique of the 18th Century*, New York: F.S. Crofts & Co.
- McLeod, A. and M. Rohrmeier (2021). "A Modular System for the Harmonic Analysis of Musical Scores Using a Large Vocabulary". In: *Proceedings of the 22nd International Society for Music Information Retrieval Conference*. ISMIR. Ed. by J. H. Lee et al. Online, pp. 435–442.
- McLeod, A. and M. Steedman (Jan. 2, 2016). "HMM-Based Voice Separation of MIDI Performance". In: *Journal of New Music Research* 45.1, pp. 17–26. DOI: 10.1080/09298215.2015.1136650.
- Mearns, L. (July 2013). "The Computational Analysis of Harmony in Western Art Music." Thesis. Queen Mary University of London.
- Melkonian, O. (Aug. 23, 2019). "Music as Language: Putting Probabilistic Temporal Graph Grammars to Good Use". In: *Proceedings of the 7th ACM SIGPLAN International Workshop on Functional Art, Music, Modeling, and Design*. FARM 2019. Berlin, Germany: Association for Computing Machinery, pp. 1–10. DOI: 10.1145/3331543.3342576.
- Meredith, D., K. Lemström, and G. A. Wiggins (Dec. 1, 2002). "Algorithms for Discovering Repeated Patterns in Multidimensional Representations of Polyphonic Music". In: *Journal of New Music Research* 31.4, pp. 321–345. DOI: 10.1076/jnmr.31.4.321.14162.
- Meyer, L. B. (1956). *Emotion and Meaning in Music*. Chicago, IL: University of Chicago Press. 315 pp.

- Mnih, V. et al. (Feb. 2015). “Human-Level Control through Deep Reinforcement Learning”. In: *Nature* 518.7540 (7540), pp. 529–533. DOI: 10.1038/nature14236.
- Morgan, E. et al. (Aug. 1, 2019). “Statistical Learning and Gestalt-like Principles Predict Melodic Expectations”. In: *Cognition* 189, pp. 23–34. DOI: 10.1016/j.cognition.2018.12.015.
- Moss, F. C., M. Neuwirth, D. Harasim, et al. (2019). “Statistical Characteristics of Tonal Harmony: A Corpus Study of Beethoven’s String Quartets”. In: *PLoS ONE* 14.6, e0217242. DOI: 10.1371/journal.pone.0217242.
- Moss, F. C., M. Neuwirth, and M. Rohrmeier (2022). “The Line of Fifths and the Co-Evolution of Tonal Pitch-Classes”. In: *Journal of Mathematics and Music*. DOI: 10.1080/17459737.2022.2044927.
- Moss, F. C. and M. Rohrmeier (Dec. 13, 2021). “Discovering Tonal Profiles with Latent Dirichlet Allocation.” in: *Music & Science*. DOI: 10.1177/20592043211048827.
- Moss, F. C., W. F. Souza, and M. Rohrmeier (Oct. 19, 2020). “Harmony and Form in Brazilian Choro: A Corpus-Driven Approach to Musical Style Analysis”. In: *Journal of New Music Research* 49.5, pp. 416–437. DOI: 10.1080/09298215.2020.1797109.
- Mukherji, S. (2014). “Generative Musical Grammar-A Minimalist Approach”. Princeton University.
- Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT press.
- Nakamura, E. et al. (Mar. 2016). “Tree-Structured Probabilistic Model of Monophonic Written Music Based on the Generative Theory of Tonal Music”. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 276–280. DOI: 10.1109/ICASSP.2016.7471680.
- Narayanan, S. and D. Jurafsky (2001). “A Bayesian Model Predicts Human Parse Preference and Reading Times in Sentence Processing”. In: *Advances in Neural Information Processing Systems*. Vol. 14. MIT Press.
- Narmour, E. (1992). *The Analysis and Cognition of Melodic Complexity: The Implication-Realization Model*. Chicago: University of Chicago Press.
- Narmour, E. (1977). *Beyond Schenkerism: The Need for Alternatives in Music Analysis*. Chicago: University of Chicago Press. 238 pp.
- (Dec. 1, 1983). “Some Major Theoretical Problems Concerning the Concept of Hierarchy in the Analysis of Tonal Music”. In: *Music Perception* 1.2, pp. 129–199. DOI: 10.2307/40285255.
- (1990). *The Analysis and Cognition of Basic Melodic Structures: The Implication-Realization Model*. The Analysis and Cognition of Basic Melodic Structures: The Implication-Realization Model. Chicago, IL, US: University of Chicago Press, pp. xiv, 485. xiv, 485.
- Nettles, B. and R. Graf (1997). *The Chord Scale Theory & Jazz Harmony*. Advance Music.

Bibliography

- Neuwirth, M., D. Harasim, et al. (2018). “The Annotated Beethoven Corpus (ABC): A Dataset of Harmonic Analyses of All Beethoven String Quartets”. In: *Frontiers in Digital Humanities* 5, p. 16. DOI: 10.3389/fgdigh.2018.00016.
- Neuwirth, M. and M. Rohrmeier (2016). “Wie wissenschaftlich muss Musiktheorie sein? Chancen und Herausforderungen musikalischer Korpusforschung”. In: *Zeitschrift der Gesellschaft für Musiktheorie [Journal of the German-speaking Society of Music Theory]* 13.2, pp. 171–193. DOI: 10.31751/915.
- Norvig, P. (2017). “On Chomsky and the Two Cultures of Statistical Learning”. In: *Berechenbarkeit der Welt? Philosophie und Wissenschaft im Zeitalter von Big Data*. Ed. by W. Pietsch, J. Wernecke, and M. Ott. Wiesbaden: Springer Fachmedien, pp. 61–83. DOI: 10.1007/978-3-658-12153-2_3.
- Oaksford, M. and N. Chater (Feb. 22, 2007). *Bayesian Rationality: The Probabilistic Approach to Human Reasoning*. OUP Oxford. 342 pp. Google Books: sLetNgiU7ugC.
- (2018). “Probabilities and Bayesian Rationality”. In: *The Routledge International Handbook of Thinking and Reasoning*. The Routledge International Handbook Series. New York, NY, US: Routledge/Taylor & Francis Group, pp. 415–433.
- Osborn, B. (2017). *Everything in Its Right Place: Analyzing Radiohead*. New York: Oxford University Press. 248 pp. DOI: 10.1093/acprof:oso/9780190629229.001.0001.
- Oudre, L., Y. Grenier, and C. Févotte (2009). “Template-Based Chord Recognition: Influence of the Chord Types.” In: *ISMIR*, pp. 153–158.
- Page, S. E. (Mar. 6, 2018). *The Model Thinker*.
- Parncutt, R. (1989). *Harmony: A Psychoacoustical Approach*. Red. by T. S. Huang et al. Vol. 19. Springer Series in Information Sciences. Berlin, Heidelberg: Springer. DOI: 10.1007/978-3-642-74831-8.
- Parncutt, R. et al. (Apr. 1, 2019). “Tone Profiles of Isolated Musical Chords: Psychoacoustic Versus Cognitive Models”. In: *Music Perception* 36.4, pp. 406–430. DOI: 10.1525/mp.2019.36.4.406.
- Pearce, M. T. (Dec. 2005). “The Construction and Evaluation of Statistical Models of Melodic Structure in Music Perception and Composition”. doctoral. City University London.
- (2018). “Statistical Learning and Probabilistic Prediction in Music Cognition: Mechanisms of Stylistic Enculturation”. In: *Annals of the New York Academy of Sciences* 1423.1, pp. 378–395. DOI: 10.1111/nyas.13654.
- Pearce, M. T. and M. Rohrmeier (Oct. 1, 2012). “Music Cognition and the Cognitive Sciences”. In: *Topics in Cognitive Science* 4.4, pp. 468–484. DOI: 10.1111/j.1756-8765.2012.01226.x.
- (2018). “Musical Syntax II: Empirical Perspectives”. In: *Springer Handbook of Systematic Musicology*. Ed. by R. Bader. Springer Handbooks. Berlin, Heidelberg: Springer, pp. 487–505. DOI: 10.1007/978-3-662-55004-5_26.

- Pearce, M. T. and G. A. Wiggins (July 1, 2006). “Expectation in Melody: The Influence of Context and Learning”. In: *Music Perception: An Interdisciplinary Journal* 23.5, pp. 377–405. DOI: 10.1525/mp.2006.23.5.377.
- (2012). “Auditory Expectation: The Information Dynamics of Music Perception and Cognition”. In: *Topics in Cognitive Science* 4.4, pp. 625–652. DOI: 10.1111/j.1756-8765.2012.01214.x.
- Pearl, J. (2009). *Causality - Models, Reasoning, and Inference*. 2nd Edition. Cambridge University Press.
- Phon-Amnuaisuk, S., A. Smaill, and G. Wiggins (Dec. 1, 2006). “Chorale Harmonization: A View from a Search Control Perspective”. In: *Journal of New Music Research* 35.4, pp. 279–305. DOI: 10.1080/09298210701458835.
- Piston, W. (1947). *Counterpoint*. New York: W. W. Norton.
- (1987). *Harmony*. Ed. by M. DeVoto. Fifth edition. New York: Norton. 575 pp.
- Polth, M. (Dec. 1, 2018). “The Individual Tone and Musical Context in Albert Simon’s Tonfeldtheorie”. In: *Music Theory Online* 24.4.
- Ponsford, D., G. Wiggins, and C. Mellish (June 1, 1999). “Statistical Learning of Harmonic Movement”. In: *Journal of New Music Research* 28.2, pp. 150–177. DOI: 10.1076/jnmr.28.2.150.3115.
- Powers, H. S. (1980). “Language Models and Musical Analysis”. In: *Ethnomusicology* 24.1, pp. 1–60. DOI: 10.2307/851308. JSTOR: 851308.
- Prince, J. B. and M. A. Schmuckler (Feb. 1, 2014). “The Tonal-Metric HierarchyA Corpus Analysis”. In: *Music Perception* 31.3, pp. 254–270. DOI: 10.1525/mp.2014.31.3.254.
- Pugin, L., R. Zitellini, and P. Roland (Oct. 27–31, 2014). “Verovio: A Library for Engraving MEI Music Notation into SVG”. In: *Proceedings of the 15th International Society for Music Information Retrieval Conference*. 15th International Society for Music Information Retrieval Conference (ISMIR 2014). In collab. with L. X. Pugin, R. Zitellini, and P. Roland. Taipei: ISMIR, pp. 107–112.
- Quick, D. and P. Hudak (Sept. 28, 2013). “Grammar-Based Automated Music Composition in Haskell”. In: *Proceedings of the First ACM SIGPLAN Workshop on Functional Art, Music, Modeling & Design*. FARM ’13. New York, NY, USA: Association for Computing Machinery, pp. 59–70. DOI: 10.1145/2505341.2505345.
- Quinn, I. and C. W. White (June 1, 2017). “Corpus-Derived Key Profiles Are Not Transpositionally Equivalent”. In: *Music Perception* 34.5, pp. 531–540. DOI: 10.1525/mp.2017.34.5.531.
- Quiroga-Martinez, D. R. et al. (2021). “Musicianship and Melodic Predictability Enhance Neural Gain in Auditory Cortex during Pitch Deviance Detection”. In: *Human Brain Mapping* 42.17, pp. 5595–5608. DOI: 10.1002/hbm.25638.
- Rabinovitch, G. (Apr. 1, 2018). “Gjerdingen’s Schemata Reexamined”. In: *Journal of Music Theory* 62.1, pp. 41–84. DOI: 10.1215/00222909-4450636.

Bibliography

- Rahn, J. (1979). "Logic, Set Theory, Music Theory". In: *College Music Symposium* 19.1, pp. 114–127. JSTOR: 40351760.
- Rameau, J.-P. (1722). *Traité de l'harmonie reduite à ses principes naturels*. Paris: J.-B.-C. Ballard.
- Ranganath, R., S. Gerrish, and D. Blei (Apr. 2, 2014). "Black Box Variational Inference". In: *Artificial Intelligence and Statistics*. Artificial Intelligence and Statistics. PMLR, pp. 814–822.
- Raphael, C. and J. Stoddard (Sept. 1, 2004). "Functional Harmonic Analysis Using Probabilistic Models". In: *Computer Music Journal* 28.3, pp. 45–52. DOI: 10.1162/0148926041790676.
- Rezende, D. J., S. Mohamed, and D. Wierstra (June 18, 2014). "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *Proceedings of the 31st International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, pp. 1278–1286.
- Rhodes, C., D. Lewis, and D. Müllensiefen (2009). "Bayesian Model Selection for Harmonic Labelling". In: *Mathematics and Computation in Music*. Ed. by T. Klouche and T. Noll. Communications in Computer and Information Science. Berlin, Heidelberg: Springer, pp. 107–116. DOI: 10.1007/978-3-642-04579-0_11.
- Ricciardi, E. (May 18, 2020). "Editing Italian Madrigals in the Digital World: The Tasso in Music Project". In: Music Encoding Conference 2020. DOI: 10.17613/17a5-2b65.
- Rice, J. A. (2014). "Adding to the Galant Schematicon: The Lully". In: *Forthcoming in Mozart-Jahrbuch*.
- (Sept. 2015). "The Morte: A Galant Voice-Leading Schema as Emblem of Lament and Compositional Building Block". In: *Eighteenth-Century Music* 12.2, pp. 157–181. DOI: 10.1017/S1478570615000287.
- Rodin, J. and C. S. Sapp (2010). *The Josquin Research Project*. URL: <https://josquin.stanford.edu/>.
- Rohrmeier, M. (2007). "A Generative Grammar Approach to Diatonic Harmonic Structure". In: *Proceedings of the 4th Sound and Music Computing Conference*, pp. 97–100.
- (Mar. 1, 2011). "Towards a Generative Syntax of Tonal Harmony". In: *Journal of Mathematics and Music* 5.1, pp. 35–53. DOI: 10.1080/17459737.2011.573676.
- (Apr. 30, 2020a). "The Syntax of Jazz Harmony: Diatonic Tonality, Phrase Structure, and Form". In: *Music Theory and Analysis (MTA)* 7.1, pp. 1–63. DOI: 10.11116/MTA.7.1.1.
- (2020b). "Towards a Formalization of Musical Rhythm". In: *Proceedings of the 21th International Society for Music Information Retrieval Conference, ISMIR 2020, Montreal, Canada, October 11-16, 2020*. Ed. by J. Cumming et al., pp. 621–629.
- Rohrmeier, M. and I. Cross (2008). "Statistical Properties of Tonal Harmony in Bach's Chorales". In.

- Rohrmeier, M. and T. Graepel (2012). “Comparing Feature-Based Models of Harmony”. In: *Proceedings of the 9th International Symposium on Computer Music Modelling and Retrieval*. Springer London, pp. 357–370.
- Rohrmeier, M. and S. Koelsch (Feb. 1, 2012). “Predictive Information Processing in Music Cognition. A Critical Review”. In: *International Journal of Psychophysiology*. Predictive Information Processing in the Brain: Principles, Neural Mechanisms and Models 83.2, pp. 164–175. DOI: 10.1016/j.ijpsycho.2011.12.010.
- Rohrmeier, M. and F. C. Moss (Nov. 7, 2021). “A Formal Model of Extended Tonal Harmony”. In: *Proceedings of the 22nd International Society for Music Information Retrieval Conference*. International Society for Music Information Retrieval Conference (ISMIR 2021). Online: ISMIR, pp. 569–578. DOI: 10.5281/zenodo.5624635.
- Rohrmeier, M. and M. Neuwirth (2015). “Towards a Syntax of the Classical Cadence”. In: Rohrmeier, M. and M. T. Pearce (2018). “Musical Syntax I: Theoretical Perspectives”. In: *Springer Handbook of Systematic Musicology*. Ed. by R. Bader. Springer Handbooks. Berlin, Heidelberg: Springer, pp. 473–486. DOI: 10.1007/978-3-662-55004-5_25.
- Rohrmeier, M. and P. Rebuschat (2012). “Implicit Learning and Acquisition of Music”. In: *Topics in Cognitive Science* 4.4, pp. 525–553. DOI: 10.1111/j.1756-8765.2012.01223.x.
- Rougier, J. (Mar. 29, 2019). “P-Values, Bayes Factors, and Sufficiency”. In: *The American Statistician* 73 (sup1), pp. 148–151. DOI: 10.1080/00031305.2018.1502684.
- Sapp, C. S. (2007). “Computational Chord-Root Identification in Symbolic Musical Data: Rationale, Methods, and Applications.” In: *Tonal Theory for the Digital Age*. Computing in Musicology 15, pp. 99–119.
- Schenker, H. (1979). *Free Composition:(Der Freie Satz): Heinrich Schenker; Translated and Edited by Ernst Oster*. Longman.
- Ścibior, A., Z. Ghahramani, and A. D. Gordon (Aug. 30, 2015). “Practical Probabilistic Programming with Monads”. In: *ACM SIGPLAN Notices* 50.12, pp. 165–176. DOI: 10.1145/2887747.2804317.
- Sears, D. R. W., A. Arzt, et al. (2017). “Modeling Harmony with Skip-Grams”. In: *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China, October 23-27, 2017*. Ed. by S. J. Cunningham et al., pp. 332–338.
- Sears, D. R. W., M. T. Pearce, et al. (Jan. 1, 2018). “Simulating Melodic and Harmonic Expectations for Tonal Cadences Using Probabilistic Models”. In: *Journal of New Music Research* 47.1, pp. 29–52. DOI: 10.1080/09298215.2017.1367010.
- Sears, D. R. W., J. Spitzer, et al. (Oct. 22, 2018). “Expecting the End: Continuous Expectancy Ratings for Tonal Cadences”. In: *Psychology of Music*, p. 0305735618803676. DOI: 10.1177/0305735618803676.
- Sears, D. R. W. and G. Widmer (July 14, 2020). “Beneath (or beyond) the Surface: Discovering Voice-Leading Patterns with Skip-Grams”. In: *Journal of Mathematics and Music* 0.0, pp. 1–26. DOI: 10.1080/17459737.2020.1785568.

Bibliography

- Shepard, R. N. (July 1982). "Geometrical Approximations to the Structure of Musical Pitch". In: *Psychological Review* 89.4, pp. 305–333. DOI: 10.1037/0033-295X.89.4.305.
- Silver, D., A. Huang, et al. (Jan. 2016). "Mastering the Game of Go with Deep Neural Networks and Tree Search". In: *Nature* 529.7587 (7587), pp. 484–489. DOI: 10.1038/nature16961.
- Silver, D., J. Schrittwieser, et al. (Oct. 2017). "Mastering the Game of Go without Human Knowledge". In: *Nature* 550.7676 (7676), pp. 354–359. DOI: 10.1038/nature24270.
- Simonetta, F. (Nov. 1, 2018). *Enhanced Wikifonia Leadsheet Dataset*. Zenodo. DOI: 10.5281/zenodo.1476555.
- Simonetta, F. et al. (Sept. 12, 2018). "Symbolic Music Similarity through a Graph-Based Representation". In: *Proceedings of the Audio Mostly 2018 on Sound in Immersion and Emotion*. AM'18. New York, NY, USA: Association for Computing Machinery, pp. 1–7. DOI: 10.1145/3243274.3243301.
- Smoliar, S. W. (1979). "A Computer Aid for Schenkerian Analysis". In: *Proceedings of the 1979 Annual Conference*. ACM '79. New York, NY, USA: ACM, pp. 110–115. DOI: 10.1145/800177.810043.
- Steedman, M. (Oct. 1, 1984). "A Generative Grammar for Jazz Chord Sequences". In: *Music Perception* 2.1, pp. 52–77. DOI: 10.2307/40285282.
- (1996). "The Blues and the Abstract Truth: Music and Mental Models". In: *Mental Models in Cognitive Science*. Erlbaum, pp. 305–318.
- Stock, J. (1993). "The Application of Schenkerian Analysis to Ethnomusicology: Problems and Possibilities". In: *Music Analysis* 12.2, pp. 215–240. DOI: 10.2307/854273. JSTOR: 854273.
- Symons, J. (2017). "A Cognitively Inspired Method for the Statistical Analysis of Eighteenth-Century Music, as Applied in Two Corpus Studies". Evanston, Illinois: Northwestern University.
- Temperley, D. (Oct. 1, 1997). "An Algorithm for Harmonic Analysis". In: *Music Perception* 15.1, pp. 31–68. DOI: 10.2307/40285738.
- (1999a). "The Question of Purpose in Music Theory: Description, Suggestion, and Explan..." In: *Current Musicology*, p. 21.
- (Oct. 1, 1999b). "What's Key for Key? The Krumhansl-Schmuckler Key-Finding Algorithm Reconsidered". In: *Music Perception* 17.1, pp. 65–100. DOI: 10.2307/40285812.
- (2002a). "A Bayesian Approach to Key-Finding". In: *Music and Artificial Intelligence*. Ed. by C. Anagnostopoulou, M. Ferrand, and A. Smaill. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 195–206. DOI: 10.1007/3-540-45722-4_18.
- (2002b). "A Bayesian Approach to Key-Finding". In: *Music and Artificial Intelligence*. Ed. by C. Anagnostopoulou, M. Ferrand, and A. Smaill. Lecture Notes in Computer

- Science. Berlin, Heidelberg: Springer, pp. 195–206. DOI: 10.1007/3-540-45722-4_18.
- (2007a). *Music and Probability*. MIT Press.
- (May 2007b). “The Melodic-Harmonic ‘Divorce’ in Rock”. In: *Popular Music* 26.2, pp. 323–342. DOI: 10.1017/S0261143007001249.
- (Mar. 1, 2009). “A Unified Probabilistic Model for Polyphonic Music Analysis”. In: *Journal of New Music Research* 38.1, pp. 3–18. DOI: 10.1080/09298210902928495.
- (2011). “Composition, Perception, and Schenkerian Theory”. In: *Music Theory Spectrum* 33.2, pp. 146–168.
- Temperley, D. and E. W. Marvin (Feb. 1, 2008). “Pitch-Class Distribution and the Identification of Key”. In: *Music Perception* 25.3, pp. 193–212. DOI: 10.1525/mp.2008.25.3.193.
- Tenenbaum, J. B. et al. (Mar. 11, 2011). “How to Grow a Mind: Statistics, Structure, and Abstraction”. In: *Science* 331.6022, pp. 1279–1285. DOI: 10.1126/science.1192788.
- Terhardt, E. (May 1974). “Pitch, Consonance, and Harmony”. In: *The Journal of the Acoustical Society of America* 55.5, pp. 1061–1069. DOI: 10.1121/1.1914648.
- (1987). “Gestalt Principles and Music Perception”. In: *Auditory Processing of Complex Sounds*. Hillsdale, NJ, US: Lawrence Erlbaum Associates, Inc, pp. 157–166.
- Tillmann, B., J. J. Bharucha, and E. Bigand (2000). “Implicit Learning of Tonality: A Self-Organizing Approach”. In: *Psychological Review* 107.4, pp. 885–913. DOI: 10.1037/0033-295X.107.4.885.
- Tomašević, D. et al. (May 4, 2021). “Exploring Annotations for Musical Pattern Discovery Gathered with Digital Annotation Tools”. In: *Journal of Mathematics and Music* 15.2, pp. 194–207. DOI: 10.1080/17459737.2021.1943026.
- Tran, D. et al. (2018). “Simple, Distributed, and Accelerated Probabilistic Programming”. In: *Neural Information Processing Systems*.
- Tymoczko, D. (July 7, 2006). “The Geometry of Musical Chords”. In: *Science* 313.5783, pp. 72–74. DOI: 10.1126/science.1126287. PMID: 16825563.
- (May 5, 2011). *A Geometry of Music: Harmony and Counterpoint in the Extended Common Practice*. Oxford Studies in Music Theory. Oxford, New York: Oxford University Press. 480 pp.
- Ullman, T. D. and J. B. Tenenbaum (Dec. 15, 2020a). “Bayesian Models of Conceptual Development: Learning as Building Models of the World”. In: *Annual Review of Developmental Psychology* 2.1, pp. 533–558. DOI: 10.1146/annurev-devpsych-121318-084833.
- (Dec. 15, 2020b). “Bayesian Models of Conceptual Development: Learning as Building Models of the World”. In: *Annual Review of Developmental Psychology* 2.1, pp. 533–558. DOI: 10.1146/annurev-devpsych-121318-084833.
- Van de Meent, J.-W. et al. (Sept. 27, 2018). “An Introduction to Probabilistic Programming”. arXiv: 1809.10756 [cs, stat].

Bibliography

- Van der Helm, P.A. (Nov. 1, 2017). “On Bayesian Simplicity in Human Visual Perceptual Organization”. In: *Perception* 46.11, pp. 1269–1282. DOI: 10.1177/0301006617719604.
- Vilares, I. and K. Kording (Apr. 2011). “Bayesian Models: The Structure of the World, Uncertainty, Behavior, and the Brain”. In: *Annals of the New York Academy of Sciences* 1224.1, pp. 22–39. DOI: 10.1111/j.1749-6632.2011.05965.x. PMID: 21486294.
- Vuust, P., O. A. Heggli, et al. (May 2022). “Music in the Brain”. In: *Nature Reviews Neuroscience* 23.5 (5), pp. 287–305. DOI: 10.1038/s41583-022-00578-5.
- Vuust, P. and M. A. G. Witek (2014). “Rhythmic Complexity and Predictive Coding: A Novel Approach to Modeling Rhythm and Meter Perception in Music”. In: *Frontiers in Psychology* 5. DOI: 10.3389/fpsyg.2014.01111.
- Vuvan, D. T. and B. Hughes (June 1, 2021). “Probe Tone Paradigm Reveals Less Differentiated Tonal Hierarchy in Rock Music”. In: *Music Perception* 38.5, pp. 425–434. DOI: 10.1525/mp.2021.38.5.425.
- Wang, Z., D. Wang, et al. (Oct. 11, 2020). “Learning Interpretable Representation for Controllable Polyphonic Music Generation”. In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. International Society for Music Information Retrieval Conference (ISMIR 2020). Montreal, Canada: ISMIR, pp. 662–669. DOI: 10.5281/zenodo.4245518.
- Wang, Z., Y. Zhang, et al. (Oct. 11, 2020). “PianoTree VAE: Structured Representation Learning for Polyphonic Music”. In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. International Society for Music Information Retrieval Conference (ISMIR 2020). Montreal, Canada: ISMIR, pp. 368–375. DOI: 10.5281/zenodo.4245446.
- Wassermann, G. and M. Glickman (June 1, 2020). “Automated Harmonization of Bass Lines from Bach Chorales: A Hybrid Approach”. In: *Computer Music Journal* 43.2-3, pp. 142–157. DOI: 10.1162/comj_a_00523.
- Weber, T. et al. (2015). “Reinforced Variational Inference”. In: *Advances in Neural Information Processing Systems (NIPS) Workshops*.
- Weiss, Y. (1996). “Interpreting Images by Propagating Bayesian Beliefs”. In: *Advances in Neural Information Processing Systems*. Vol. 9. MIT Press.
- Wertheimer, M. (1923). “Laws of Organization in Perceptual Forms”. In: *A source book of Gestalt Psychology* 1.
- Westergaard, P. (1975). *An Introduction to Tonal Theory*. New York: Norton. 435 pp.
- White, B. W. (1960). “Recognition of Distorted Melodies”. In: *The American Journal of Psychology* 73.1, pp. 100–107. DOI: 10.2307/1419120. JSTOR: 1419120.
- White, C. W. and I. Quinn (Nov. 1, 2018). “Chord Context and Harmonic Function in Tonal Music”. In: *Music Theory Spectrum* 40.2, 314–335. DOI: 10.1093/mts/mty021.
- Whorley, R. P. and D. Conklin (Apr. 2, 2016). “Music Generation from Statistical Models of Harmony”. In: *Journal of New Music Research* 45.2, pp. 160–183. DOI: 10.1080/09298215.2016.1173708.

- Widdess, R. (1981). "Aspects of Form in North Indian Ālāp and Dhrupad". In: *Music and Tradition: Essays on Asian and Other Musics Presented to Laurence Picken*. Cambridge: Cambridge University Press, pp. 143–182.
- Wiggins, G. A. (July 1, 2012). "Music, Mind and Mathematics: Theory, Reality and Formality". In: *Journal of Mathematics and Music* 6.2, pp. 111–123. DOI: 10.1080/17459737.2012.694710.
- Wilk, C. M. and S. Sagayama (2019). "Automatic Music Completion Based on Joint Optimization of Harmony Progression and Voicing". In: *Journal of Information Processing* 27.0, pp. 693–700. DOI: 10.2197/ipsjjip.27.693.
- Winn, J. (2020). *Model-Based Machine Learning*. Online.
- Winograd, T. (1968). "Linguistics and the Computer Analysis of Tonal Harmony". In: *Journal of Music Theory* 12.1, pp. 2–49. DOI: 10.2307/842885. JSTOR: 842885.
- Witten, I. H. and T. Bell (July 1991). "The Zero-Frequency Problem: Estimating the Probabilities of Novel Events in Adaptive Text Compression". In: *IEEE Transactions on Information Theory* 37.4, pp. 1085–1094. DOI: 10.1109/18.87000.
- Wood, E. J.-W. van de Meent, and V. Mansinghka (2014). "A New Approach to Probabilistic Programming Inference". In: *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, pp. 1024–1032.
- Yeary, M. (Aug. 30, 2011). "Perception, Pitch, and Musical Chords". UMI/ProQuest.
- Yust, J. (2006). "Formal Models of Prolongation". University of Washington.
- (June 1, 2015a). "Distorted Continuity: Chromatic Harmony, Uniform Sequences, and Quantized Voice Leadings". In: *Music Theory Spectrum* 37.1, pp. 120–143. DOI: 10.1093/mts/mtu020.
- (Dec. 1, 2015b). "Voice-Leading Transformation and Generative Theories of Tonal Structure". In: *Music Theory Online* 21.4.
- (July 12, 2018). *Organized Time: Rhythm, Tonality, and Form*. Oxford Studies in Music Theory. Oxford, New York: Oxford University Press. 440 pp.

Christoph Finkensiep

Digital and Cognitive Musicology Lab (DCML)
École Polytechnique Fédérale de Lausanne
INN 115, Station 14
CH-1015, Lausanne
Switzerland

✉ chfin@chfin.de
✉ christoph.finkensiep@epfl.ch
🆔 orcid.org/0000-0002-7002-6586
🐙 github.com/chfin
👤 Christoph Finkensiep

Education

- 2017-2022 **PhD**
Digital and Cognitive Musicology Lab (DCML)
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE, Switzerland
- 2014-2017 **M. Sc.** in Cognitive Science
UNIVERSITY OF OSNABRÜCK, Germany
- 2010-2014 **B. Sc.** in Computer Science
UNIVERSITY OF PADERBORN, Germany
- Fall 2012 Exchange Semester
UNIVERSITY OF HELSINKI, Finland
- 2010 **Abitur**
GYMNASIUM HORN-BAD MEINBERG, Germany

Theses

- 2022 “The Structure of Free Polyphony”
PhD Thesis
Supervisor: Martin Rohrmeier (EPFL)
- 2017 “A Formal Model of Voice Leading”
Master Thesis
Supervisors: Kai-Uwe Kühnberger (University of Osnabrück), Martin Rohrmeier (TU Dresden)
- 2014 “Fast and Flexible Automatic Composition of Semantic Web Services”
Bachelor Thesis
Supervisor: Hans Kleine Büning (University of Paderborn)

Employment

since 2017 ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Bibliography

Doctoral Assistant
Digital and Cognitive Musicology Lab

2017 TECHNICAL UNIVERSITY DRESDEN
Student Assistant

Dresden Music Cognition Lab

2015-2017 PROTESTANT CHURCH OF LADBERGEN
Director of Brass Ensemble

2013-2014 UNIVERSITY OF PADERBORN
Teaching Assistant *Introduction to Programming for Engineers*

2011-12 UNIVERSITY OF PADERBORN
Student Assistant
Decision Support & Operations Research Lab


Publications


Articles, Conference Papers, and Book Chapters

forthcoming **C. Finkensiep**, P. Ericson, S. Klassmann, and M. Rohrmeier (in preparation). “Chord Types and Ornamentation – A Bayesian Model of Extended Chord Profiles”. In: *Open Research Europe*

C. Finkensiep, M. Neuwirth, and M. Rohrmeier (submitted). “Music Theory and Model-Driven Corpus Research”. In: *Oxford Handbook of Corpus Studies in Music*. Oxford: Oxford University Press

G. Cecchetti, S. A. Herff, **C. Finkensiep**, D. Harasim, and M. Rohrmeier (in press). “Hearing Functional Harmony in Jazz: A Perceptual Study on Music-Theoretical Accounts of Extended Tonality.” In: *Musicae Scientiae*

2021 **C. Finkensiep** and M. Rohrmeier (2021). “Modeling and Inferring Proto-Voice Structure in Free Polyphony”. In: *Proceedings of the 22nd International Society for Music Information Retrieval Conference*. ISMIR. Online, pp. 189–196.  [10.5281/zenodo.5624431](https://doi.org/10.5281/zenodo.5624431)

S. A. Herff, D. Harasim, G. Cecchetti, **C. Finkensiep**, and M. Rohrmeier (2021). “Hierarchical Syntactic Structure Predicts Listeners’ Sequence Completion in Music”. In: *Proceedings of the Annual Meeting of the Cognitive Science Society*. CogSci. Vol. 43. Online.  <https://escholarship.org/uc/item/9w44g4x1>

D. Harasim, **C. Finkensiep**, L. Bigo, M. Giraud, F. Levé, D. R. W. Sears, D. Shanahan, and M. Rohrmeier (2021). “Music Cognition: The Complexity of Musical Structure”. In: *Proceedings of the Annual Meeting of the Cognitive Science Society*. CogSci. Vol. 43. Online.  <https://escholarship.org/uc/item/9qh5m4dc>

2020 **C. Finkensiep**, K. Déguernel, M. Neuwirth, and M. Rohrmeier (2020). “Voice-Leading Schema

Recognition Using Rhythm and Pitch Features”. In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. ISMIR. Montreal, Canada (online), pp. 520–526. doi 10.5281/zenodo.4245482

D. Harasim, **C. Finkensiep**, P. Ericson, T. J. O’Donnell, and M. Rohrmeier (2020). “The Jazz Harmony Treebank”. In: *Proceedings of the 21st International Society for Music Information Retrieval Conference*. ISMIR. Montreal, Canada (online), pp. 207–215. doi 10.5281/zenodo.4245406

G. Cecchetti, S. A. Herff, **C. Finkensiep**, and M. Rohrmeier (2020). “The Experience of Musical Structure as Computation : What Can We Learn?” In: *Rivista di analisi e teoria musicale* XXVI, 2, 2020, pp. 91–127. doi 10.53152/1032. https://www.torrossa.com/en/resources/an/4943892

2019 **C. Finkensiep**, R. Widdess, and M. Rohrmeier (2019). “Modelling the Syntax of North Indian Melodies with a Generalized Graph Grammar”. In: *Proceedings of the 20th International Society for Music Information Retrieval Conference*. ISMIR. Delft, The Netherlands, pp. 462–469. doi 10.5281/zenodo.3527844

2018 **C. Finkensiep**, M. Neuwirth, and M. Rohrmeier (2018). “Generalized Skipgrams for Pattern Discovery in Polyphonic Streams”. In: *Proceedings of the 19th International Society for Music Information Retrieval Conference*. ISMIR. Paris, France, pp. 547–553. doi 10.5281/zenodo.1492473. http://ismir2018.ircam.fr/doc/pdfs/202_Paper.pdf

Talks and Presentations

(☞ = invited talk, ☞ = conference talk, ☐ = poster)

- 2022 ☞ “Protovoices – A Model of Tonal Structure at the Surface” Sep. 30, 2022
in symposium: “Abstraction, Theory Building, Models”
☞ “Models in Practice”
workshop, with Johannes Hentschel
GMTH Kongress 2022
Salzburg, Austria
- 2021 ☞ ☐ “Modeling and Inferring Proto-Voice Structure in Free Polyphony” Nov. 8-12, 2021
ISMIR 2021, online
☞ “Modeling Hierarchical Structure at the Note Level” July 29, 2021
in symposium: “Music Cognition: The Complexity of Musical Structure”
CogSci 2021, online
- 2020 ☞ ☐ “Voice-Leading Schema Recognition” Oct. 12-16, 2020
ISMIR 2020, online
- 2019 ☞ ☐ “A Graph Grammar for North Indian Melodies” Nov. 6, 2019
ISMIR 2019
Delft, The Netherlands

Bibliography

- 🗨️ “A Graph Representation of Hierarchical Voice-Leading” July 2, 2019
Workshop: “Corpus Research as a Means of Unlocking Musical Grammar”
UNIVERSITY OF TEL-AVIV & BAR-ILAN UNIVERSITY
Tel Aviv, Israel
- 🗨️ “Representing Hierarchical Voice-Leading Structure with Graph Transformations” Mar. 25, 2019
SEMPRE Graduate Conference 2019
Cambridge, UK
- 🗨️ “From Bach to the Beatles” Feb. 21, 2019
with Markus Neuwirth & Martin Rohrmeier
“Mixed Methods” in the Humanities mid-term meeting, Volkswagen Foundation
Hildesheim, Germany
- 🗨️ “Schema Matching mit Skipgrams” Feb. 20, 2019
HOCHSCHULE FÜR MUSIK FREIBURG
Freiburg, Germany
- 2018 🗨️🗨️ “Generalized Skipgrams” Sep. 26, 2018
ISMIR 2018
Paris, France
- 🗨️ “Schema Matching with Skipgrams” Sep. 19, 2018
DCML Workshop 2018
ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE
Lausanne, Switzerland
- 🗨️ “Digitale Musikforschung: Projekte und Fallstudien” June 14-15, 2018
with Markus Neuwirth
UNIVERSITY OF MUSIC AND PERFORMING ARTS MANNHEIM
Mannheim, Germany
- 2017 🗨️ “Formalizing Four-Part Voice Leading Rules” Jan. 17, 2017
TECHNICAL UNIVERSITY DRESDEN
Dresden, Germany

Open Source Code

Implementation of the Protovoice Model (Haskell)

🔗 [DCMLab/protovoices-haskell](#)

Annotation Tool for Protovoice Analyses (PureScript)

🔗 [DCMLab/protovoice-annotation-tool](#)

Annotation Tool for Voice-Leading Schemata (ClojureScript)

🔗 [DCMLab/schema_annotation_app](#)

Annotation Tool for Tree Analyses (ClojureScript)

🔗 [DCMLab/tree-annotation-code](#)

Matcher for Voice-Leading Schemata (Julia)

[DCMLab/schemata_code/tree/ismir2020](https://github.com/DCMLab/schemata_code/tree/ismir2020)

Libraries for Musical Pitches and Intervals

- `musicology-pitch` (Haskell)
[DCMLab/haskell-musicology/tree/master/musicology-pitch](https://github.com/DCMLab/haskell-musicology/tree/master/musicology-pitch)
- `pitchtypes` (Python)
[DCMLab/pitchtypes](https://github.com/DCMLab/pitchtypes)
- `Pitches.jl` (Julia)
[DCMLab/Pitches.jl](https://github.com/DCMLab/Pitches.jl)
- `purescript-pitches` (PureScript)
[DCMLab/purescript-pitches](https://github.com/DCMLab/purescript-pitches)
- `pitches` (Rust)
[DCMLab/rust-pitches](https://github.com/DCMLab/rust-pitches)

Teaching

<i>2018-2022</i>	Supervision of Student Projects
<i>Spring 2019 & 2020</i>	Teaching Assistant: “Digital Musicology” Master Course, EPFL
<i>Spring 2019 & 2020</i>	Co-Lecturer: “Musical Improvisation, Invention and Creativity” Bachelor Course, EPFL
<i>2019</i>	Teaching Assistant: “Music, Musical Structure, Artificial Neural Networks, and the Mind” Summer School, Magliaso, Switzerland
<i>Winter 2013/14</i>	Teaching Assistant: “Introduction to Programming for Engineers” Bachelor Course, UNIVERSITY OF PADERBORN

Reviewer Activity

<i>Conferences</i>	ISMIR
<i>Journals</i>	Music Theory and Analysis (MTA), TISMIR

Scholarships and Awards

<i>2021</i>	Best Reviewer Award, <i>ISMIR 2021</i>
<i>2010-2017</i>	Student Scholarship, <i>German Academic Scholarship Foundation</i>
<i>2012</i>	ERASMUS Scholarship (exchange semester), <i>European Union</i>

Bibliography

2011 3rd Place for Task 2, *Data Mining Cup*

Memberships and Responsibilities

2019-2021 Student Representative
Doctoral School of Digital Humanities, EPFL

2013-2014 Student Represtenative
German Academic Scholarship Foundation, UNIVERSITY OF PADERBORN

since 2021 Member of the *Cognitive Science Society*

since 2019 Member of the *UNIL-EPFL dhCenter*

since 2018 Member of the *International Society for Music Information Retrieval (ISMIR)*

Skills

Languages

German native speaker
English fluent
French basic skills

Computing

Programming Languages Haskell, Python, Julia, PureScript, Clojure/ClojureScript, Common Lisp, JavaScript, Rust, Java, C/C++, PHP

Frameworks / Libraries numpy, pandas, PyTorch, halogen, monad transformers, probabilistic programming (pyro, Haskell)

Markup Languages \LaTeX , HTML/CSS, LilyPond

Music

since 1998 Trombone
10 years of lessons, including university level
Experience in various jazz and classical ensembles, solo literature

- Minor during Bachelor (2011-2014)
- Musikakademie, German Academic Scholarship Foundation (bass trombone, 2012-14, 2016-17)
- Landesjugendposaunenchor Westfalen Lippe (lead trombone, 2013-2019)

- Big Band Hochschule Osnabrück, Jazz Department (bass trombone, 2014-2015)

since 2001 Piano

since 2007 Conducting (brass ensemble)

since 2013 Choir Singing