THE VIS FRAMEWORK: ANALYZING COUNTERPOINT IN LARGE DATASETS

Christopher Antila and Julie Cumming
McGill University
christopher@antila.ca; julie.cumming@mcgill.ca

ABSTRACT

The *VIS Framework for Music Analysis* is a modular Python library designed for "big data" queries in symbolic musical data. Initially created as a tool for studying musical style change in counterpoint, we have built on the `music21` and *pandas* libraries to provide the foundation for much more.

We describe the musicological needs that inspired the creation and growth of the VIS Framework, along with a survey of similar previous research. To demonstrate the effectiveness of our analytic approach and software, we present a sample query showing that the most commonly repeated contrapuntal patterns vary between three related style periods. We also emphasize our adaptation of typical n-gram-based research in music, our implementation strategy in VIS, and the flexibility of this approach for future researchers.

1. INTRODUCTION

1.1 Counterpoint

"The evolution of Western music can be characterized in terms of a dialectic between acceptable vertical sonorities on the one hand... and acceptable melodic motions on the other." [12] A full understanding of polyphonic music (with more than one voice or part) requires description in terms of this dialectic, which is called counterpoint. Whereas music information retrieval research (such as [6]) typically describes polyphonic music only in terms of vertical (simultaneous or harmonic) intervals, musicologists interested in contrapuntal patterns also want to know the horizontal (sequential or melodic) intervals in each voice that connect the vertical intervals. Since counterpoint describes how pitches in independent voices are combined in polyphonic music, a computerized approach to counterpoint analysis of symbolic music can provide a wealth of information to musicologists, who have previously relied primarily on prose descriptions of musical style. [1]

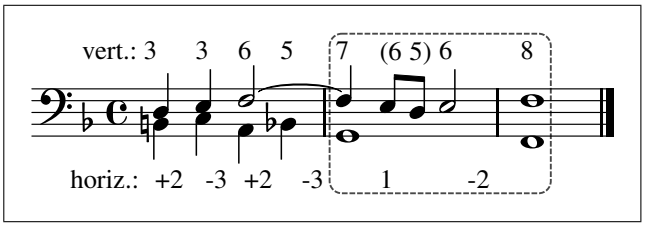

Figure 1. Symbolic score annotated with vertical and horizontal intervals. A common contrapuntal module appears in the box.

Figure 1 shows the counterpoint between two voices in a fragment of music. We annotated the vertical intervals above the score, and the lower voice's horizontal intervals below. Note that we show intervals by diatonic step size, counting number of lines and spaces between two notes, rather than semitones. We describe this contrapuntal module further in Section 2.1. By using intervals rather than note names, we can generalize patterns across pitch levels, so the same pattern may start on any pitch. For this article, we ignore interval quality (e.g., major or minor third) by using diatonic intervals (e.g., third), allowing generalization across mode and key. We do use interval quality for other queries—this is a choice available in VIS at runtime.

To allow computerized processing of contrapuntal patterns, we encode the counterpoint between two voices with alternating vertical and horizontal intervals. In Figure 1, the first two beats are "3 +2 3." We call these patterns interval n-grams, where n is the number of vertical intervals. Our n-gram notation system is easily intelligible to music theorists and musicologists, and allows us to stay close to musicology.

1.2 Research Questions

Until recently, musicologists' ability to accurately describe polyphonic textures was severely limited: any one person can learn only a limited amount of music in a lifetime, and the computer-based tools for describing or analyzing polyphonic music in detail are insufficiently precise for many repertoires. Descriptions of musical style and style change are often vague, derived from intuitive impressions and personal knowledge of repertoire rather than quantifiable

[1] We wish to thank the following people for their contributions: Natasha Dillabough, Ichiro Fujinaga, Jane Hatter, Jamie Klassen, Alexander Morgan, Catherine Motuz, Peter Schubert. The ELVIS Project was supported by the Digging into Data Challenge; the Canadian team responsible for the work described in this paper was additionally funded by the Social Sciences and Humanities Research Council of Canada.

evidence. Our project attempts the opposite by quantitatively describing musical style change using counterpoint.

We chose counterpoint not only because musicologists are already aware of its importance, but because it allows us to consider structure in all polyphonic music, which includes the majority of surviving Western music created after 1300. Our project's initial goal is to find the most frequently-repeated contrapuntal patterns for different periods, genres, and composers, to help form detailed, evidence-based descriptions of style periods and style change by knowing which features change over time and when. In addition, statistical models will allow a fresh approach to attribution problems (determining the composer of a piece where it is not otherwise known), by enabling us to describe some of the factors that distinguish a composer's style.

1.3 The VIS Framework

Our project's most important accomplishment is the VIS Framework—the software we developed to answer the research questions described above. (VIS stands for "vertical interval successions," which is a way to describe counterpoint). Currently VIS's primary function is to find contrapuntal patterns in symbolic music, recording them with the notation described above in Figure 1 so they may be counted. However, we designed the framework to allow a much broader set of queries, and we intend to add support for additional musical dimensions (like meter and harmony) as well as more complicated statistical experiments (like Markov-chain modeling).

We used the *Counterpoint Web App*, a Web-based user interface for VIS's counterpoint functionality, to run the analyses presented in this article. [2] Such Web-based software encourages musicologists to participate in data-driven analysis even if they are otherwise unable to program. The Web App's visual design, the use of musicologist-friendly terms and user workflows, and the ability to output analysis results on musical scores are significant advantages. At the same time, programmers are encouraged to download and extend the VIS Framework using its well-documented Python API. While our Framework provides a guide for structuring analysis workflows, each analytic step benefits from our integration of the `music21` and *pandas* libraries. Together, these allow analytic approaches more amenable to musicians and statisticians, respectively. [3]

2. BACKGROUND

2.1 Contrapuntal Modules

A *contrapuntal module* is a repeated contrapuntal pattern made from a series of vertical (harmonic) and horizontal (melodic) intervals—a repeated interval n-gram. [11] We are primarily interested in the frequency and nature of two-voice contrapuntal modules. VIS allows us to computerize tedious score analysis previously done by hand, as when Peter Schubert identified modules in Palestrina. [13] While two-voice contrapuntal modules are the primary structural element of much Renaissance music, we can find contrapuntal modules in nearly all polyphonic music, so our software and research strategies will be useful for a wide range of music. [3]

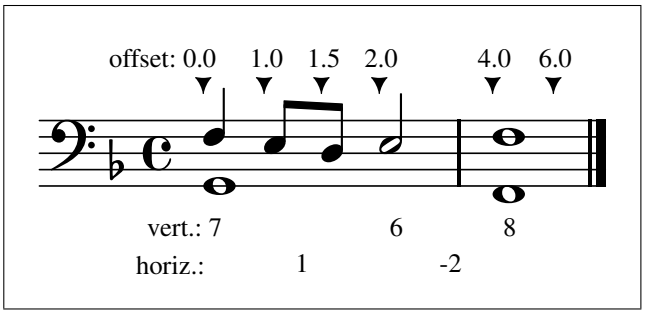

Figure 2. "Cadence" contrapuntal module from Figure 1, with `music21` offset values.

Figure 2 shows a representation of the "7 1 6 -2 8" interval 3-gram (a 3-gram because there are three vertical intervals). Using a half-note rhythmic offset, the first vertical interval is a seventh, the horizontal motion of the lower part is a unison (1), there is a vertical sixth, the lower part moves down by a second (-2), and the final vertical interval is an octave. In modal counterpoint, this is a highly conventionalized figure used to signal a cadence—a closing or concluding gesture for a phrase or piece. This is the same 3-gram as in the box in Figure 1.

Importantly, our analysis method requires that voicing information is encoded in our files. MIDI files where all parts are given in the same channel cannot be analyzed usefully with our software.

2.2 Previous Uses of n-Grams in MIR

We have chosen to map musical patterns with n-grams partly because of their previous use in natural language processing. [4] Some previous uses of n-grams in music analysis, and computerized counterpoint analysis, are described below.

J. Stephen Downie's dissertation presents a method for indexing melodic n-grams in a large set of folk melodies that will be searched using "Query by Humming" (QBH). [7] Downie's system is optimized for what he calls "lookup," rather than "analysis," and he admits that it lacks the detail required by musicologists. Importantly, Downie only indexes horizontal intervals: melody rather than counterpoint.

Another QBH lookup system, proposed by Shyamala Doraisamy, adapts n-grams for polyphonic music. [5, 6] While this system does account for polyphony, it does not record horizontal intervals so it lacks the detailed contrapuntal information we seek. Furthermore, Doraisamy's intervals are based on MIDI note numbers rather than the diatonic steps preferred by musicologists. Finally, the largest interval allowed by Doraisamy's tokenization strategy is 26

[2] Visit `counterpoint.elvisproject.ca`.

[3] Refer to `pandas.pydata.org` and `mit.edu/music21`.

[4] As in the *Google Ngram Viewer*; refer to `books.google.com/ngrams`.

semitones—just over two octaves, and therefore narrower than the normal distance between outer voices even in Renaissance polyphony. Considering the gradual expansion of range in polyphonic music, to the extremes of the late 19th-century orchestra, the gradual appearance of large intervals may be an important style indicator.

Meredith has proposed geometric transformation systems for encoding multi-dimensional musical information in a computer-friendly way. [9, 10] We especially appreciate the multi-dimensional emphasis and the mathematical properties of these systems, and the software's ability to work even when voicing information is not available in the symbolic file.

Finally, Jürgensen has studied accidentals in a large fifteenth-century manuscript of organ intabulations with an approach very similar to ours, but carried out using the Humdrum Toolkit. [8] She locates cadences by identifying a contrapuntal model and then records the use of accidentals at the cadence. While she searches only for specific contrapuntal modules, we identify all of the n-grams in a test set in order to determine the most frequently recurring contrapuntal patterns.

2.3 Multi-Dimensional n-Grams in VIS

Considering these previous uses of n-grams and counterpoint in MIR, we designed our software with the flexibility to accommodate our requirements, as well as those of future analysis strategies. By tokenizing n-grams with strings that minimally transform the input, musicologists can readily understand the information presented in an n-gram. This strategy offers a further benefit to programmers, who can easily create n-grams that include different musical dimensions without necessarily developing a new token transformation system. Users may choose to implement any tokenization strategy on top of our existing n-gram-indexing module.

The example 3-gram shown in Figure 2 is tokenized internally as "7 1," "6 -2," "8 END." Although there appear to be $2n - 1$ tokens, we consider a vertical interval and its following horizontal interval as a combined unit—as though it were a letter in an n-gram as used in computational linguistics. The simplicity afforded by using strings as tokens, each of which may contain an arbitrary array of musical information, has been advantageous.

Indeed, the difficulty of determining a musical analogue to the letter, word, and phrase divisions used in computational linguistics may be one of the reasons that computer-driven research has yet to gain much traction in mainstream musicology. That music lacks an equivalent for space characters poses an even greater problem in this regard: while some music does use clear breaks between phrases, their exact placement can often be disputed among experts. Musicologists also wish to account for the multiple simultaneous melody lines of polyphonic music, which has no equivalent in natural language. These are the primary motivating factors behind our multi-dimensional interval n-gram tokens that encode both vertical and horizontal intervals. As our research continues, context models and multiple viewpoint systems, in the style of Conklin and Witten, will partially obviate the questions of which n value to use, and of how best to incorporate varied musical elements. [1]

The popularity of Python within scientific computing communities allows us to benefit from any software that accepts *pandas* data objects. The easy-to-learn, object-oriented API of `music21`, along with the relatively high number of supported file formats, are also significant advantages. In the 1980s, a music analysis toolkit consisting of a collection of *awk* scripts was sensible, but Humdrum's limitation to UNIX systems and a single symbolic file format pose undesirable limitations for a big data project.

3. EXPERIMENT

3.1 Data Sets

We present an experiment to quantitatively describe style change in the Renaissance period, providing a partial answer for our primary research question. [5] We assembled test sets for three similar style periods, named after a representative composer from the period: Ockeghem (1440–85), Josquin (1485–1521), and Palestrina (1540–85). The pieces in the test set were chosen to represent the style period as accurately as our project's database allowed. [6] The twenty-year gap between the later periods is a result of less symbolic music being available from those decades. Each set consists of a mixture of sacred and secular vocal music, most with four parts, in a variety of genres, from a variety of composers. Though we analyzed n-grams between two and twenty-eight vertical intervals long, we report our results only for 3-grams because they are the shortest contrapuntal unit that holds meaning. Note that we include results from all possible two-part combinations, reflecting Renaissance contrapuntal thinking, where many-part textures are composed from a series of two-part structures. [3, 13]

The Ockeghem test set consists of 50 files: 28 in the MIDI format and 22 in **kern. For the composers, 8 pieces were written by Busnoys, 32 by Ockeghem, and 10 are late works by Dufay. The longest repeated n-gram was a 25-gram.

The Josquin test set consists of 56 files: 18 MIDI, 23 **kern, 9 MusicXML, and 6 NoteWorthy Composer. For the composers, 3 pieces were written by Agricola, 7 by Brumel, 6 by Compre, 2 by Fvin, 12 by Isaac, 19 by Josquin, 3 by Mouton, 2 by Obrecht, and 2 by la Rue. The longest repeated n-gram was a 28-gram.

Finally, the Palestrina test set consists of 53 files: 30 MIDI, 15 **kern, 6 MusicXML, and 2 NoteWorthy Composer. For the composers, 15 pieces were written by Palestrina, 9 by Rore, 28 by Victoria, and 1 by Wert. The longest repeated n-gram was a 26-gram.

3.2 Methodology

The *VIS Framework* uses a modular approach to query design, dividing analysis tasks into a series of well-defined

[5] You may download our test sets from `elvisproject.ca/ismir2014`.

[6] Visit `database.elvisproject.ca`.

steps. [7] We intend the module break-down to be helpful for musicologists who wish to reason about and design their own queries. Thus, musicological concerns drove the creation of many of the analysis steps, such as the filtering modules described below. The interval n-gram frequency experiment in this article uses the following modules: *NoteRestIndexer*, *IntervalIndexer*, *HorizontalIntervalIndexer*, *FilterByOffsetIndexer*, *FilterByRepeatIndexer*, *NGramIndexer*, *ColumnAggregator*, and finally the *FrequencyExperimenter*. [8]

The *NoteRestIndexer* finds note names and rests from a `music21` *Score*. The *IntervalIndexer* and *HorizontalIntervalIndexer* calculate vertical and horizontal intervals, respectively.

The *FilterByOffsetIndexer* uses a basic algorithm to filter weak-beat embellishing tones that otherwise obscure structural counterpoint. We regularize observations to a given rhythmic offset time interval using the `music21` offset, measured in quarter lengths. Refer to Figure 2 as an example, where vertical intervals are filtered with a 2.0 offset. Events beginning on a multiple of that duration will be retained (like the notes at 0.0, 2.0, and 4.0). Events lasting for multiples of that duration will appear to be repeated (like the note at 4.0, which is also recorded at 6.0). Events not beginning on a multiple of the duration will be removed (like the notes at 1.0 and 1.5) or shifted to the following offset, if no new event occurs. For this study, we chose a half-note (2.0) offset interval in accordance with Renaissance notation practices, but this can be changed in VIS at runtime.

The *FilterByRepeatIndexer* removes events that are identical to the immediately preceding event. Because of its placement in our workflow for this experiment, subsequent vertical intervals will not be counted if they use the same pitches. Our interval n-grams therefore necessarily involve contrapuntal *motion*, which is required for proper pattern recognition. Such repeated events arise in musical scores, for example, when singers recite many words on the same pitch. The *FilterByOffsetIndexer* may also create repeated events, as at offset 6.0 in Figure 2. Users may choose not to run this module.

In this article, our *NGramIndexer* includes results from all pairs of part combination. Users may exclude some combinations at runtime, choosing to limit their query to the highest and lowest parts, for example. On receiving intervals from the *FilterByRepeatIndexer*, the *NGramIndexer* uses the gliding window technique to capture all possible overlapping interval n-grams. The indexer also accepts a list of tokens that prevent an n-gram from being counted. We use this feature to avoid counting contrapuntal patterns that include rests. Finally, the *NGramIndexer* may add grouping characters, surrounding "vertical" events in brackets and "horizontal" events in parentheses to enhance legibility of long n-grams. The 3-grams in this article are short enough that grouping characters are unnecessary; on the other hand, the legibility of the "[10] (+2) [9] (1) [8] (+2) [7] (1) [6] (-2) [8]" 6-gram found 27 times in the Palestrina test set greatly benefits from grouping characters.

The *FrequencyExperimenter* counts the number of occurrences of each n-gram. These results, still specific to part combinations within pieces, are then combined with the *ColumnAggregator*.

On receiving a spreadsheet of results from VIS, we calculated the number of n-grams as the percentage total of all n-grams in each of the test sets. For each set, we also counted the total number of 3-grams observed (including all repetitions of all 3-grams), the number of distinct 3-gram types (whether repeated or not), and the number of 3-gram types that occur more than once; these are shown below in Table 1.

3.3 Results

Due to the limited time span represented in this study, we wish to suggest avenues for future exploration, rather than offer conclusive findings. We present a visualization of the experimental results in Figure 3, a hybrid between a Venn diagram, word cloud (i.e., a 3-gram cloud), and a timeline. The diagram includes interval 3-grams that constitute greater than 0.2% of the 3-grams in at least one of the test sets. When a 3-gram appears in an intersection of style periods, that 3-gram constitutes greater than 0.2% of the 3-grams in those sets. As in a world cloud, the font size is scaled proportionately to a 3-gram's frequency in the test sets in which it is common. Most visually striking is the confirmation of musicologists' existing experiential knowledge: certain contrapuntal patterns are common to all three style periods, including the cadence module ("7 1 6 -2 8") and two other 3-grams that end with the "7 1 6" cadential suspension. These results make sense because cadences are an essential feature of musical syntax.

Test Set	Total	Types	Repeated Types
Ockeghem	30,640	10,644	4,509 (42%)
Josquin	31,233	9,268	4,323 (47%)
Palestrina	33,339	10,773	5,023 (47%)

Table 1. Summary of 3-gram repetitions in our query.

In addition to the common cadential patterns noted above, both Figure 3 and Table 1 show evidence of stylistic change over time. Most notably, the Josquin and Palestrina test sets show a higher level of repetition than the Ockeghem set. The number of 3-grams included in Figure 3 is higher in the Josquin test set (with seventeen 3-grams) than either the Ockeghem or Palestrina sets (both with eleven 3-grams). Yet Table 1 indicates the Josquin and Palestrina sets both have a higher percentage of 3-gram types that are repeated at least once (47% in both sets, compared to 42% in the Ockeghem set). These data suggest an increase in repetition of contrapuntal modules from the Ockeghem to the Josquin generations, which was retained in the Palestrina generation. Figure 3 only partially reinforces this suggestion: while five 3-grams are unique to the Ockeghem set, six are unique to the Josquin set, but only one is unique

[7] This section refers to the 2.x release series.

[8] For more information about the VIS Framework's analysis modules and overall architecture, please refer to our Python API at `vis.elvisproject.ca`.

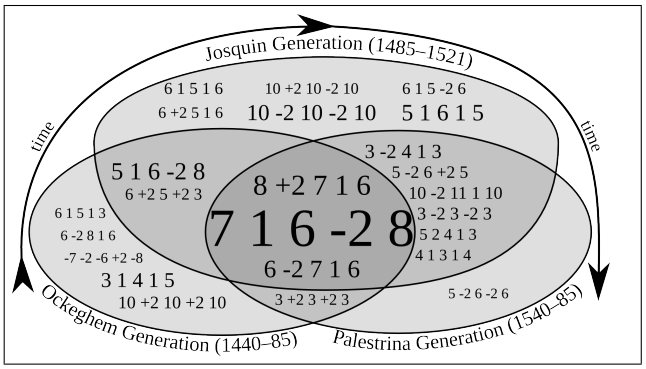

Figure 3. Frequency of contrapuntal modules is different between temporally-adjacent style periods.

to the Palestrina set. Moreover, the "5 -2 6 -2 6" module, unique to the Palestrina set, is the least common 3-gram in Figure 3—how did contrapuntal repetition both decrease in Palestrina's generation *and* remain the same?

Previous research by Cumming and Schubert may help us explain the data. In 2008, Cumming noted that exact repetition became much more common in Josquin's lifetime than in Ockeghem's. [2] Schubert showed that composers tended to repeat contrapuntal patterns in inversion during Palestrina's lifetime, so that the lower voice is moved above the original upper voice. [13] Inversion changes a contrapuntal pattern's vertical intervals in a consistent way that preserves, but switches, the horizontal intervals of the two parts. For example, "7 1 6 -2 8" inverts at the octave to "2 -2 3 +2 1." While humans can recognize both forms as two versions of the same pattern, VIS currently shows only exact repetition; future enhancements will permit us to equate the original and the inversion. This decision may explain why our data show lower rates of repetition for the Palestrina test set.

We find further evidence of stylistic change in Figure 3: certain patterns that musicologists consider to be common across all Renaissance music are in fact not equally common in our three test sets. For example, motion by parallel thirds and tenths appears to be more common in certain style periods than others, and in a way that does not yet make sense. The Palestrina set shares ascending parallel thirds ("3 +2 3 +2 3") with the Ockeghem and descending parallel thirds ("3 -2 3 -2 3") with the Josquin set. Ascending parallel tenths ("10 +2 10 +2 10") are more common in the Ockeghem set, and descending parallel tenths ("10 -2 10 -2 10") in the Josquin set. In particular, descending parallel thirds are an order of magnitude less common in the Ockeghem test set than the Josquin or Palestrina (constituting 0.013%, 0.272%, and 0.225% of 3-grams in their test set, respectively). Conventional musicological wisdom suggests these 3-grams will be equally common in all three test sets, and that parallel tenths will be more common than parallel thirds in later style periods, as the range between voices expands. Since the reasons for such a deviation are not yet known, we require further investigation to study the changing nature of contrapuntal repetition during the Renaissance period. Yet even with these preliminary findings it is clear that evidence-based research has much to offer musicology.

4. FUTURE WORK

Our research will continue by extending VIS to add the option of equivalence classes that can group, for example, inversionally-related interval n-grams. We will also build on previous work with melody- and harmony-focussed multiple viewpoint systems to create an all-voice contrapuntal prediction model. [1, 14]

Our experiments will continue with larger test sets for increased confidence in our findings, also adding style periods earlier than the Ockeghem and later than the Palestrina sets, and subdividing our current style periods. This will help us reassess boundaries between style periods, and exactly what such a boundary entails. We will also compare results of single pieces with test sets of various sizes.

Finally, we will implement additional multi-dimensional n-gram tokens, for example by adding the note name of the lowest voice. This approach would encode Figure 2 as "7 F 1 6 F -2 8 E." In Renaissance music, this type of n-gram will clarify the relationships between contrapuntal modules and a piece's mode.

5. CONCLUSION

The *VIS Framework for Music Analysis* is a musicologist-friendly Python library designed to analyze large amounts of symbolic musical data. Thus far, our work has concentrated on counterpoint—successions of vertical intervals and the horizontal intervals connecting them—which some scholars view as composers' primary concern throughout the development of Western music. Our software uses multi-dimensional n-grams to find and count the frequency of repeated contrapuntal patterns, or modules. In particular, by retaining all inputted dimensions and using strings as tokens (rather than integers or characters), we simultaneously allow musicologists to quickly understand the content of an n-gram while also avoiding the challenge of developing a new tokenization strategy for every musical dimension added to the n-gram. We hope this flexibility and ease-of-use encourages musicologists and non-expert programmers, who would otherwise be discouraged from computer-based music analysis, to experiment more freely.

The results of our query presented in this article, which compares the most commonly-repeated contrapuntal modules in three Renaissance style periods, show the type of insight possible from computerized music research. The time-consuming effort required for previous work on contrapuntal modules is greatly reduced when analysts have access to specialized computer software. We analyzed more than 150 polyphonic compositions for interval n-grams between two and twenty-eight vertical intervals in length, which would have taken months or years for a human. Even with simple mathematical strategies like counting the frequency of interval n-grams to know which are most common, we can confirm existing intuitive knowledge about

the foundations of counterpoint while also suggesting avenues for future research on the nature of musical repetition.

6. REFERENCES

[1] D. Conklin and I. Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1):51–73, 1995.

[2] J. Cumming. From variety to repetition: The birth of imitative polyphony. In Bruno Bouckaert, Eugeen Schreurs, and Ivan Asselman, editors, *Yearbook of the Alamire Foundation*, number 6, pages 21–44. Alamire, 2008.

[3] J. Cumming. From two-part framework to movable module. In Judith Peraino, editor, *Medieval music in practice: Studies in honor of Richard Crocker*, pages 177–215. American Institute of Musicology, 2013.

[4] M. S. Cuthbert and C. Ariza. music21: A toolkit for computer-aided musicology and symbolic music data. In *Proceedings of the International Symposium on Music Information Retrieval*, pages 637–42, 2010.

[5] S. Doraisamy. *Polyphonic Music Retrieval: The n-gram approach*. PhD thesis, University of London, 2004.

[6] S. Doraisamy and S. Rger. Robust polyphonic music retrieval with n-grams. *Journal of Intelligent Information Systems*, 21(1):53–70, 2003.

[7] J. S. Downie. *Evaluating a Simple Approach to Music Information Retrieval: Conceiving melodic n-grams as text*. PhD thesis, University of Western Ontario, 1999.

[8] F. Jürgensen. Cadential accidentals in the Buxheim organ book and its concordances: A midfifteenth-century context for musica ficta practice. *Acta Musicologica*, 83(1):39–68, 2011.

[9] D. Meredith. A geometric language for representing structure in polyphonic music. In *Proceedings of the International Society for Music Information Retrieval*, pages 133–8, 2012.

[10] D. Meredith, K. Lemström, and G. Wiggins. Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *Journal of New Music Research*, 41(4):321–45.

[11] J. A. Owens. *Composers at Work: The Craft of Musical Composition 1450–1600*. Oxford University Press, 1997.

[12] P. Schubert. Counterpoint pedagogy in the renaissance. In T. Christensen, editor, *The Cambridge History of Western Music Theory*, pages 503–33. Cambridge University Press, 2002.

[13] P. Schubert. Hidden forms in Palestrina's first book of four-voice motets. *Journal of the American Musicological Society*, 60(3):483–556, 2007.

[14] R. Whorley, G. Wiggins, C. Rhodes, and M. Pearce. Multiple viewpoint systems: Time complexity and the construction of domains for complex musical viewpoints in the harmonisation problem. *Journal of New Music Research*, 42(3):237–66, 2013.