ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

The slice-group-analyze pipeline
A flexible visualization framework for comparative corpus studies

Johannes Hentschel
Martin Rohrmeier

The recent decades have given rise to *computational musicology* as a distinct field of research (Huron, 1999; Meredith, 2016; Volk et al., 2011). Corpus studies have played a vital role in this multi-disciplinary endeavour (Shanahan et al., 2022) and resulted in an uncountable plethora of music processing software for the creation, analysis, and visualization of music corpora. The specificity of research questions and approaches, as well as the heterogeneity of the employed music data (e.g., audio files, score encodings, annotations), so far seem to have hindered the emergence of a unifying corpus analysis framework that would allow researchers to execute well-known or custom algorithms and transformations on corpora of diverse origins and formats. In this paper we propose to make heterogeneous datasets interoperable by converting them into the same tabular data structure, and we introduce the slice-group-analyze pipeline as a flexible and powerful conceptual tool for devising, performing and communicating corpus-based research. Using the Python library `DiMCAT` (Hentschel et al., submitted for review), we demonstrate the intuitive correspondence between particular pipeline configurations and the visualizations they afford.

Our framework draws on the observation of three commonplace operations that computational musicologists frequently need to perform on corpora. First of all, by *analyzing* we understand the computation of one result per piece or per slice, depending on whether the analysis is preceded by a slicing operation (see below), or not. The result type (e.g., a number, a distribution, a matrix) determines which types of plots lend themselves for presenting and inspecting the analysis results. Furthermore, some plot types may be better suited than others for visualizing results in grouped fashion, be it within the same figure or in sub-plots with shared axes. In other words, the exact composition of slicing and grouping operations in an analysis pipeline, in our framework, determines the most useful ways of visualizing its outcomes.

By *slicing* we understand the segmentation of a piece of music either by equal-sized time intervals (i.e., contiguous or overlapping windows) or by a set of arbitrarily spaced time points derived from a given feature, such as inferred beats or unique note onset positions ("chordify"). Presence of a slicing operation in a pipeline reduces the unit of analysis from the piece to the slice level. In other words, any subsequent analysis will yield one result per segment instead of per piece (the default).

By *grouping* we mean binning the pieces or slices of a dataset based on a criterion, such as composition/recording dates, chord changes, or outputs of a key finding or clustering algorithm. Grouping *per se* does not change the unit of analysis (piece or slice); instead, it is the principal operation allowing us to make scientific claims based on the statistical comparison of groups that are homogeneous in some regard. Moreover, grouped visualization of the individual analysis results—potentially including aggregated values such as means, quartiles, or centroids—represents a meaningful way of inspecting and making sense of them.

As the data structure underlying these operations we suggest the dataframe (Petersohn, 2021) in combination with an addressing scheme for musical timelines. This approach enables representing the information from diverse datasets (chord analyses, audio-score alignments, MIDI data) in a unified way and creating new alignments wherever necessary. One way of using dataframes efficiently is by including only the musical objects that are relevant for a particular analysis. For example, a pipeline whose purpose is the analysis of (absolute) guitar chords needs to perform all previous slicing and grouping operations only on a single dataframe containing exactly one row per chord, with an index keeping apart rows pertaining to different slices and/or pieces. The demonstration of the above principles focuses on the manifold ways by which pitch profiles can be derived, visualized, and evaluated, based on a large dataset of annotated score encodings.

REFERENCES

Hentschel, J., McLeod, A., Rammos, Y., & Rohrmeier, M. (submitted for review). Introducing DiMCAT to utilize the dataframe for processing and analyzing notated music on a very large scale.

Huron, D. (1999). *The New Empiricism: Systematic Musicology in a Postmodern Age* (Music and Mind: Foundations of Cognitive Musicology No. 3). The 1999 Ernest Bloch Lecture. University of California, Berkely.

Meredith, D. (Ed.). (2016). *Computational Music Analysis*. Springer. https://doi.org/10.1007/978-3-319-25931-4

Petersohn, D. (2021). *Dataframe Systems: Theory, Architecture, and Implementation* (Doctoral dissertation). University of California. Berkeley.

Shanahan, D., Burgoyne, J. A., & Quinn, I. (Eds.). (2022). *The Oxford Handbook of Music and Corpus Studies* (First). Oxford University Press. https://doi.org/10.1093/oxfordhb/9780190945442.001.0001

Volk, A., Wiering, F., & van Kranenburg, P. (2011). Unfolding the Potential of Computational Musicology. *Proceedings of the 13th International Conference on Informatics and Semiotics in Organisations*, 137–144.