Key-Finding Based on a Hidden Markov Model and Key Profiles

Néstor Nápoles López
McGill University, CIRMMT
Montréal, QC, Canada
nestor.napoleslopez@mail.mcgill.ca

Claire Arthur
Georgia Institute of Technology
Atlanta, GA, USA
claire.arthur@gatech.edu

Ichiro Fujinaga
McGill University, CIRMMT
Montréal, QC, Canada
ichiro.fujinaga@mcgill.ca

ABSTRACT

Musicologists and musicians often would like to search by keys in a digital music library. In this paper, we introduce a new key-finding algorithm that can be applied to music in both symbolic and audio formats. The algorithm, which is based on a Hidden Markov Model (HMM), provides two stages of key-finding output; the first one referring to local keys and the second one to the *global* key.

We describe the input, the two output stages, and the parameters of the model. In particular, we describe two configurable parameters, the *transition probability distributions*, which are based on a matrix of neighbouring keys, and the *emission probability distributions*, which make use of established key profiles.

We discuss the *local* key-finding capabilities of the algorithm, presenting an example analysis of the Prelude Op. 28 No. 20 in C minor by Chopin, showing the local key regions obtained using different key profiles. We evaluate the *global* key-finding capabilities of the model, using an existing dataset and six well-known key profiles as different model parameters.

Since different key profiles will tend to err or misclassify in different ways and across different pieces, we train an ensemble method with the predictions from all the key profiles (6) through our model. We show that the ensemble method achieves state-of-the-art performance for major and overall keys, however, it still underperforms the state-of-the-art for minor keys.

CCS CONCEPTS

• **Information systems** → **Music retrieval**; • **Applied computing** → **Sound and music computing**; • **Computing methodologies** → *Machine learning approaches*; *Markov decision processes.*

KEYWORDS

symbolic key-finding, audio key-finding. music information retrieval, symbolic music analysis, key profiles, hidden markov model

ACM Reference Format:
Néstor Nápoles López, Claire Arthur, and Ichiro Fujinaga. 2019. Key-Finding Based on a Hidden Markov Model and Key Profiles. In *6th International Conference on Digital Libraries for Musicology (DLfM '19), November 9, 2019, The Hague, Netherlands.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3358664.3358675

1 INTRODUCTION

Within musicology and music information retrieval (MIR), it is often desirable to automatically extract the key of a piece of music. Numerous key-finding algorithms have been developed over the past decades for exactly this purpose. Typically, however, these key-finding algorithms are domain-specific, working exclusively on either representations of digital audio *or* symbolic music.[1] As the fields of computational musicology and MIR continue to grow, it is becoming more common for numerous tasks to require datasets that include both symbolic and audio representations of music (e.g., music performance analysis, automatic chord analysis, etc.). In this scenario, applying the same algorithm to the audio and symbolic data could prove to be convenient for accessing larger amounts of annotated data and to simultaneously increase the performance and consistency of the model across domains. In this paper, we propose a novel key-finding algorithm that is capable of processing symbolic data with an accuracy level that matches or exceeds the state-of-the-art [2] while maintaining a similar performance in the audio domain. Moreover, our algorithm automatically segments the music and provides key labels for each segment—what we call the *local* key, as well as an overall key estimate for the entire piece—what we call the *global* key. To our knowledge our algorithm is the first to comprehensively provide local and global keys in the same model while delivering similar performance in the symbolic and audio domains. Given these characteristics of the model, we believe this is the first of its kind.

2 OVERVIEW OF THE MODEL

In this section we introduce the design of the model: its inputs, parameters, and outputs. For practical applications, a full implementation of the model (supporting raw MIDI and WAV files) is made available.[2]

2.1 Inputs

Our model requires a sequence of pitch classes as input (i.e., the letter-name or chroma representation of each note). In order to automatically obtain the sequence in both domains, the model pre-processes symbolic and audio inputs in a fairly different way.

2.1.1 Symbolic. For symbolic inputs, the only pre-processing required is the removal of all information accompanying each note aside from its pitch class. Our model is implemented to accept MIDI files, however, working with other symbolic music representations (e.g., Humdrum, Lilypond, MEI, and MusicXML) is trivial if an external parser for that format is available.

[1]For a brief overview of recent symbolic approaches, see [13]; for an overview of audio approaches—including common methods and limitations—see [7].
[2]https://github.com/napulen/justkeydding

2.1.2 Audio. For audio inputs, an external algorithm is used to compute chromagram[3] features of the raw audio. The individual chroma *bins* (12) in each chromagram-frame are converted into discrete pitch-class events when their energy surpasses a threshold value. The chromagram algorithm used in the implementation of the model is the *NNLS Chroma* [9] with its default parameters.

After the corresponding pre-processing steps, the pitch-class sequences are ready to be consumed by the model. When several notes sound simultaneously, a bottom-to-top approach is preferred for dispatching the notes into the algorithm. Figure 1 shows an example of a pitch-class sequence.[4]

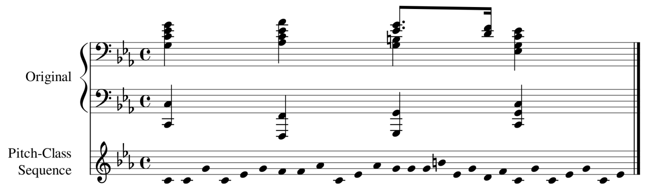

Figure 1: Pitch-class sequence from the first measure of Chopin's Op. 28 No. 20

Although additional information beyond pitch class (e.g., pitch spelling, pitch height, time signature, duration, and measures) is potentially available in symbolic music files, we omit this information from the model, by design, in order to facilitate the use of the model on audio data, where this information is not easily obtainable (e.g., polyphonic pitch estimation, pitch spelling, duration, etc.). Consequently, this also simplifies the parameters of the model.

2.2 Parameters

Our model is based on a Hidden Markov Model (HMM) [14]. An HMM typically consists of *observation symbols*, *states*, a *transition probability distribution*, an *emission probability distribution*, and an *initial state* distribution. We describe each of these parameters.

2.2.1 Observation symbols. Each observation is one of the twelve pitch classes, denoted by the numbers [0-11]. Every enharmonic spelling of the same pitch class is collapsed into the same number.

2.2.2 States. Each of the 24 musical keys, denoted by the numbers [0-23]. Similarly, all enharmonic spellings are collapsed. The numbers [0-11] denote major keys and [12-23] minor keys.

2.2.3 Initial state. We assume a uniform distribution across all possible starting keys. That is, starting in a given key is just as likely as starting in any other key.

2.2.4 Transition probability distribution. In general, this parameter determines how likely it is to change from one state to another. Here, where the states represent the musical keys, the *transition probability distribution* determines how likely it is to change from one key to another.

Table 1: Matrix of neighbouring keys. Column-wise, the keys follow the *circle-of-fifths*. Row-wise, each key is surrounded by its relative and parallel major (or minor) keys

A♯	a♯	C♯	c♯	E	e	G	g	B♭
D♯	d♯	F♯	f♯	A	a	C	c	E♭
G♯	g♯	B	b	D	d	F	f	A♭
C♯	c♯	E	e	G	g	B♭	b♭	D♭
F♯	f♯	A	a	C	c	E♭	e♭	G♭
B	b	D	d	F	f	A♭	a♭	C♭
E	e	G	g	B♭	b♭	D♭	d♭	F♭
A	a	C	c	E♭	e♭	G♭	g♭	B♭♭
D	d	F	f	A♭	a♭	C♭	c♭	E♭♭

Assuming that we know the *current* key in an excerpt of tonal music, we also know that if that the key changes, some keys (e.g., dominant or relative major/minor) are more likely to be the new key than others. Using the theoretical concept of *distance* between keys, these transitions of key can be formalized as probabilities. Thus, we define the notion of *key distance* using a table of neighbouring keys, shown in Table 1, and originally introduced by Weber [19].

Taking any one key as the *current* key and measuring the euclidean distance to the other keys in the matrix permits the grouping of the 24 keys into 9 groups. Each group is sequentially *further away* from the *current* key, and all of the keys within a group are located at the same distance from the *current* key. Table 2 shows an example of the groups obtained if the current key is C Major.[5]

Table 2: Key distance groups with respect to C Major

Group	1	2	3	4	5	6	7	8	9
	C	F	d	D	E	D♭	e♭	c♯	F♯
Keys		G	e	E♭	A♭	B	f♯	e♭	
		a	f	A	b♭				
		c	g	B♭	b				

The transition probability distributions are computed by assigning a probability to every key according to these 9 groups of key distance with respect to the *current* key. The decrease in probability of key distance between the current key group and a given subsequent key group is controlled by a *ratio* parameter. For example, a ratio of 10 means that any key in that subsequent group is 10 times less likely than any key in the current group. Figure 2 shows the distributions obtained for 3 different ratios when the current key is C Major. The distributions for other keys are obtained through transposition.[6]

2.2.5 Emission probability distributions. In general, the *emission probability distribution* parameter determines how likely it is that one state *emits* an observation symbol. Here, where the observation

[3]A chromagram is a reduction of all the harmonic content of an audio signal into 12 (or more) *bins* or classes. It is typically computed over small time windows, similar to a spectrogram.
[4]Although the example utilizes conventional music notation (including accidentals) to denote the pitch classes (e.g., E♭), in practice, the model encodes pitch classes with numbers from 0 to 11 instead.

[5]Notice that all the enharmonic keys have been collapsed into a single spelling and, when the distance between two enharmonics differs (e.g., $d(C, D\flat) \neq d(C, C\sharp)$), the smallest distance has been used.
[6]Throughout this paper, we assume that pitch classes and keys are transpositionally equivalent, although we know that research does not support that claim [13]. Further exploration is still needed to address these issues in future versions of the model.

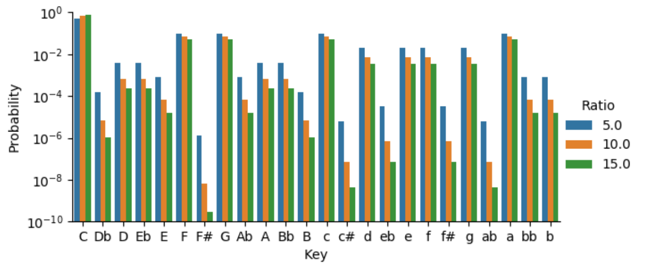

Figure 2: Probability of *transitioning* to any key if the current key is C Major

symbols represent pitch classes, the *emission probability distribution* determines how likely it is that a key *emits* a particular pitch class.

Key profiles have been used in several key-finding algorithms in the past. We consider them to be adequate for defining the *emission probability distribution* of the HMM. We collected and normalized six well-known key profiles to be used as *emission probability distributions*: Krumhansl-Kessler (KK) [8], Aarden-Essen (AE) [1], Bellman-Budge (BB) [3], Temperley (T) [16], Sapp (S) [15], and Albrecht-Shanahan (AS) [2]. Figure 3 shows the *emission* probability of each pitch class if the keys were C Major or C Minor. Similarly, the distributions of other keys are obtained through transposition.[6]

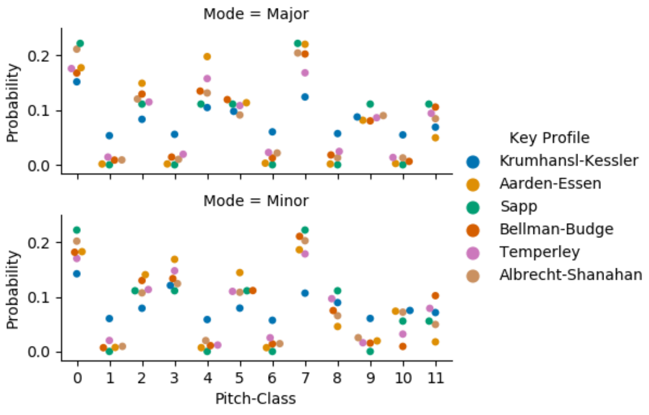

Figure 3: Probabilities of *emitting* a pitch class in C Major (top) and C Minor (bottom), obtained from six key profiles

2.3 Outputs

We consider it advantageous to divide the output of the model into two stages, as some users might be more interested in *local* keys (e.g., those studying roman numeral analysis) while others may only be interested in the *global* key (e.g., for metadata labeling).

2.3.1 Stage 1: Local key segmentation. In this stage, a key is assigned to each pitch class of the input sequence.

The process for assigning the keys depends entirely on the output of the *viterbi* algorithm [5] given the parameters of the model and the pitch-class input sequence as a series of *observations*. In general terms, the model favors a key that can explain all the pitch classes in a sequence as diatonic degrees. When the current pitch class in a pitch-class sequence can no longer be explained by the prevailing key, the model will carry out a transition to another key (typically, one that is *close* to the current key), thus segmenting the sequence into *local* keys. A simple intuition of this stage is: *for each pitch class in a sequence of pitch classes, what is the local key that most likely generated that pitch class?*

2.3.2 Stage 2: Global key. Here, the segmentation of *local* keys from the first stage is re-analyzed to find the *global* key. The process is very similar to the previous stage except that now the *local keys* (as opposed to pitch classes) become the new *observations* of the model. A simple intuition of this stage is: *given a sequence of local keys, what is the global key that best explains the sequence?*

3 EVALUATING THE MODEL

We describe the dataset utilized, discuss the *local* key-finding capabilities of the model, and evaluate its *global* key-finding performance by comparing it against other *global* key-finding algorithms in the symbolic domain.

3.1 Dataset

We used the dataset derived from Albrecht and Shanahan [2], which consists of 982 symbolic music files encoded in `**kern` format. The files have been converted to MIDI using the Humdrum Toolkit [6] and dispatched into the implementation of our HMM model.

3.2 Local keys in Prelude Op. 28 No. 20

We selected a short piece with a relatively simple structure, the Prelude Op. 28 No. 20 in C minor by Chopin (part of the Albrecht-Shanahan dataset), to illustrate the *local* key segmentation output of our algorithm. In this case, the segmentation is affected exclusively by the selection of the key profile. Figure 4 shows the segmentation of the model for this musical piece, according to which of the six key profiles was used as the emission probability distribution. The x-axis represents each individual note in the piece.

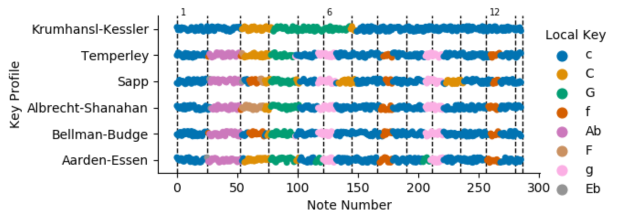

Figure 4: Output of the local key segmentation in Prelude Op.28 No.20 in C minor by Frédéric Chopin. Measures are delimited by dashed vertical lines.

Most of the key profiles output a similar *local* key segmentation, however, it is visually evident that the Krumhansl-Kessler (KK) key profile provides a segmentation with fewer (and therefore longer) local keys. An inspection of Figure 3 shows that the probability distribution in the KK key profile assigns higher values to non-diatonic pitch classes in both major and minor modes, compared to the other key profiles. Thus, relying on the KK profile has an obvious effect in the segmentation of the piece, such that the "tolerance" for non-diatonic tones is higher, and therefore they tend to be explained by the same *local* key. A contrasting example to the KK profile is the Sapp (S) key profile. An inspection of Figure 3 shows that the probability from S assigns a lower value (in fact, zero) to the non-diatonic pitch classes. In the local key segmentations, this coincides with more (and therefore shorter) local keys, which are particularly visible in Figure 4 during measures 3, 6, and 10.

Further work is still needed in order to provide a quantitative evaluation of the *local* keys provided by our model.

3.3 Global key-finding results

In the evaluation of the *global* key-finding capabilities of the model, we followed a similar methodology to Albrecht and Shanahan. We divided the dataset in two parts, 490 files reserved for training and 492 reserved for testing. Since a specific "split" of the training and testing data may provide significantly different results than another one, it is a common practice to perform cross-validation experiments with different sets of training and testing data. In our evaluation, we cross-validated our results by running our model over 20 random permutations of the dataset, splitting it in 490 files for training and 492 for testing every time. For this, we used the `ShuffleSplit` class in *scikit-learn* [11]. We report the average accuracy obtained throughout the experiment.

3.3.1 Individual key profiles. The results of the individual key profiles in the HMM can be seen in the middle group of rows in Table 3. For all of these evaluations, the model used a key distance *ratio* of 15. Therefore, the difference in the results corresponds exclusively with the key profile used as *emission probability distribution.*

Table 3: Accuracy Ratings for Key-Finding Methods in Major, Minor, and Overall pieces

Algorithm	Major	Minor	Overall
Krumhansl-Schmuckler	69.0%	83.2%	74.2%
Temperley (with KS algorithm)	96.8%	74.3%	88.6%
Bellman-Budge	94.9%	84.4%	91.1%
Aarden-Essen	90.7%	93.3%	90.4%
Sapp	92.3%	87.2%	90.4%
Albrecht-Shanahan1	92.7%	85.5%	90.0%
Albrecht-Shanahan2	89.1%	95.0%	91.3%
justkeydding (KK)	79.5%	76.3%	78.4%
justkeydding (AE)	86.1%	89.9%	87.5%
justkeydding (S)	89.2%	87.4%	88.5%
justkeydding (BB)	94.3%	81.1%	89.5%
justkeydding (T)	94.2%	83.3%	90.2%
justkeydding (AS)	93.1%	88.1%	91.3%
justkeydding (meta-classifier)	96.1%	91.5%	94.4%

3.3.2 Ensemble method. We observed that different key profiles will tend to predict or misclassify in different ways and across different pieces, motivating the use of those predictions as the input features of a meta-classifier. For this, we used an additional model, a *Logistic Regression* classifier, where the predictions from each HMM classifier are used as input features. In addition to the six HMMs derived from the different key profiles, we also used three *ratios* (5, 10, and 15) of distance between groups of keys. In total, for every training example, the meta-classifier receives 432 input features (6 key profiles * 3 *ratios* of key distance * probability assigned to 24 keys).[7]

3.3.3 Audio. In order to evaluate the robustness of the model in the audio domain compared to the symbolic domain, the previous experiment was repeated over a synthesized-audio version of the same dataset. The audio was synthesized using the default soundfont of the *MuseScore 2* music notation editor. The configurations of the model (i.e., key profiles and key distance *ratios*) were the same ones as above. The results for that experiment show a similar performance of the model, averaging 96.0% for pieces in major keys, 90.9% for pieces in minor keys, and 94.2% overall accuracy, using audio input data. We acknowledge that these results are based on the use of synthesized audio and may not reflect the performance obtained in "real" recorded audio. However, an earlier version of the HMM model was evaluated with recorded audio (from several genres) in the MIREX key estimation task of 2018.[8] This evaluated version did not include the meta-classifier and corresponds to a single HMM model with the Sapp (S) key profile. We thus suspect that our current model would out-perform our previous one.

4 CONCLUSION AND FUTURE WORK

In this paper, we presented a novel key-finding algorithm that is capable of processing data in the symbolic and audio domains. Moreover, the algorithm provides two stages of output: *local* keys and a *global* key. We consider that *local* keys could be useful in contexts where high-granularity annotations of the current key are required (e.g., roman numeral analysis), while digital music libraries could index their pieces through automatic *global* key annotations.

Although different models for processing symbolic and audio data simultaneously [4, 12, 17, 18], finding *local* keys [10], and finding *global* keys [2, 7] have been proposed over the years, we propose them within a single, comprehensive, model. We evaluate our model, showing that it maintains or exceeds the accuracy of the state-of-the-art *global* key-finding in the symbolic domain. Given these characteristics, we consider that it is the first of its kind.

An evaluation of the model in the audio domain has been presented, showing that a similar performance is maintained in a dataset of synthesized audio. Future work could investigate the use of datasets of recorded audio and the performance of our model against other *global* key-finding algorithms in the audio domain.

Similarly, further work is still needed in order to provide a quantitative evaluation of *local* keys, however, this would require expert ground truth data for *local* keys—something that is currently scarce.

ACKNOWLEDGMENTS

This research has been supported by the Social Sciences and Humanities Research Council of Canada (SSHRC) and the Fonds de recherche du Québec–Société et culture (FRQSC).

[7]The logistic regression meta-classifier was trained using the *scikit-learn* [11] library, with a parameter $C = 0.7$ and an *lbfgs* solver

[8]https://www.music-ir.org/mirex/wiki/2018:Audio_Key_Detection_Results

REFERENCES

[1] Bret J. Aarden. 2003. *Dynamic Melodic Expectancy.* Ph.D. Dissertation. The Ohio State University.

[2] Joshua Albrecht and Daniel Shanahan. 2013. The Use of Large Corpora to Train a New Type of Key-Finding Algorithm: An Improved Treatment of the Minor Mode. *Music Perception: An Interdisciplinary Journal* 31, 1 (2013), 59–67.

[3] Héctor Bellmann. 2006. About the Determination of Key of a Musical Excerpt. In *Computer Music Modeling and Retrieval*, Richard Kronland-Martinet, Thierry Voinier, and Solvi Ystad (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 76–91.

[4] Tom Collins, Sebastian Böck, Florian Krebs, and Gerhard Widmer. 2014. Bridging the Audio-Symbolic Gap: The Discovery of Repeated Note Content Directly from Polyphonic Music Audio. In *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*. Audio Engineering Society. http://www.aes.org/e-lib/browse.cfm?elib=17096

[5] David Forney. 1973. The Viterbi Algorithm. *Proc. IEEE* 61, 3 (1973), 268–278.

[6] David Huron. 2002. Music Information Processing Using the Humdrum Toolkit: Concepts, Examples, and Lessons. *Computer Music Journal* 26, 2 (2002), 11–26.

[7] Filip Korzeniowski and Gerhard Widmer. 2017. End-to-End Musical Key Estimation Using a Convolutional Neural Network. In *2017 25th European Signal Processing Conference (EUSIPCO)*. 966–970. https://doi.org/10.23919/EUSIPCO.2017.8081351

[8] Carol L. Krumhansl and Edward J. Kessler. 1982. Tracing the Dynamic Changes in Perceived Tonal Organization in a Spatial Representation of Musical Keys. *Psychological Review* 89, 4 (1982), 334–368.

[9] Matthias Mauch and Simon Dixon. 2010. Approximate Note Transcription for the Improved Identification of Difficult Chords. In *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR 2010)*.

[10] Hélène Papadopoulos and Geoffroy Peeters. 2009. Local Key Estimation Based on Harmonic and Metric Structures. In *12th International Conference on Digital Audio Effects (DAFx-09)*, Fabio Antonacci (Ed.). Politecnico de Milano, Como, Italy, 408–415. https://hal.archives-ouvertes.fr/hal-00511452

[11] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[12] Jeremy Pickens, Juan Pablo Bello, Giuliano Monti, Mark Sandler, Tim Crawford, Matthew Dovey, and Don Byrd. 2003. Polyphonic Score Retrieval Using Polyphonic Audio Queries: A Harmonic Modeling Approach. *Journal of New Music Research* 32, 2 (2003), 223–236. https://doi.org/10.1076/jnmr.32.2.223.16742

[13] Ian Quinn and Christopher Wm. White. 2017. Corpus-Derived Key Profiles Are Not Transpositionally Equivalent. *Music Perception: An Interdisciplinary Journal* 34, 5 (2017), 531–540. https://doi.org/10.1525/mp.2017.34.5.531 arXiv:https://mp.ucpress.edu/content/34/5/531.full.pdf

[14] Lawrence R. Rabiner and Biing-Hwang Juang. 1986. An Introduction to Hidden Markov Models. *IEEE ASSP Magazine* 3, 1 (1986), 4–16.

[15] Craig Stuart Sapp. 2011. *Computational Methods for the Analysis of Musical Structure.* Ph.D. Dissertation. Stanford University.

[16] David Temperley. 2002. A Bayesian Approach to Key-Finding. In *Music and Artificial Intelligence*, Christina Anagnostopoulou, Miguel Ferrand, and Alan Smaill (Eds.). Springer, Berlin, Heidelberg, 195–206.

[17] Petri Toiviainen. 2007. Visualization of Tonal Content in the Symbolic and Audio Domains. *Computing in Musicology* 15 (2007), 187–199.

[18] George Tzanetakis, Andrey Ermolinskiy, and Perry R. Cook. 2002. Pitch Histograms in Audio and Symbolic Music Information Retrieval. In *Proceedings of the 3rd International Society for Music Information Retrieval Conference.* https://doi.org/10.1076/jnmr.32.2.143.16743

[19] Gottfried Weber, James Franklin Warner, and John Bishop. 1851. *Theory of Musical Composition, Treated With a View to a Naturally Consecutive Arrangement of Topics.* R. Cocks, London.