Transactions of the International Society for
Music Information Retrieval

Anonymous, A. (2020). Unveiling the Hierarchical Structure of Music by Multi-Resolution Community Detection, *Transactions of the International Society for Music Information Retrieval*, V(N), pp. xx–xx, DOI: https://doi.org/xx.xxxx/xxxx.xx

RESEARCH ARTICLE

Unveiling the Hierarchical Structure of Music by Multi-Resolution Community Detection

Abstract

Human perception of musical structure is supposed to depend on the generation of hierarchies, which is inherently related to the actual organisation of sounds in music. Musical structures are indeed best retained by listeners when they form hierarchical patterns, with consequent implications on the appreciation of music and its performance. The automatic detection of musical structure in audio recordings is one of the most challenging problems in the field of music information retrieval, since even human experts tend to disagree on the structural decomposition of a piece of music. However, most of the current music segmentation algorithms in literature can only produce flat segmentations, meaning that they cannot segment music at different levels in order to reveal its hierarchical structure. In this paper, we propose a novel methodology for the hierarchical analysis of music structure that is based on graph theory and multi-resolution community detection. This method can perform both the tasks of boundary detection and structural grouping, without the need of particular constraints that would limit the resulting segmentation. To evaluate our approach, we designed an experiment that allowed to compare its segmentation performance with that of the current state of the art algorithms for hierarchical segmentation. Our results indicate that the proposed methodology can achieve state of the art performances on a well-known benchmark dataset, thus providing a deeper analysis of musical structure.

Keywords: Music Structure Analysis, Music Information Retrieval, Unsupervised Learning.

1. Introduction

One of the most complex aspects of music is the way a composer organises and relates his or her musical ideas within a composition. Different musical ideas are in general presented to provide contrast and surprise the listener, whereas others are repeated at different times or even varied in order to create a sense of familiarity. Not only these musical patterns are closely interrelated, but they can also be decomposed into progressively shorter ideas in light of their hierarchical organisation. This interplay between similarity, novelty and hierarchical decomposition of musical patterns is coherently organised in such a way as to convey the composer's unitary vision of the piece. This structural aspect of notated and performed music is known as musical form, which can be defined as a genre specific definition of the expectation of how a piece is composed at different time scales. Musical form is indeed one of the most evident examples of the latent hierarchical structures encoded in music. In Western music, at the most granular level it consists of individual notes and chords – the basic building blocks of a composition; by combining them sequentially and synchronously we obtain larger structural constructs such as measures, motives and phrases, which in turn contribute to the definition of sections. Examples of sections in popular music are *intro*, *chorus*, and *verse*, whereas in classical music we can find *exposition*, *development* and the *recapitulation* of a movement (Müller, 2015). These high-level structural elements determine the overall layout of a composition, and the resulting nested organisation of sounds makes it possible to visualise the hierarchical structure of a piece using tree representations of music (Lerdahl and Jackendoff, 1985).

The general goal of music structure analysis (MSA) consists in decomposing or segmenting a given music representation into patterns or temporal units that correspond to musical parts and to group these segments into musically meaningful categories (Müller, 2015). From this definition, we can identify two related subtasks that considered together make it possible to segment music effectively: the first problem is known as *boundary detection* and it aims at detecting the time instants where a transition from a music segment to the next one occurs; the second sub-task is called *structural grouping*, and it consists in labelling each segment ac-

cording to their similarity or musical function.

As detailed in (Müller, 2015), there are different criteria for segmenting and structuring music, though most of them can be categorised according to the focus they give to the following properties: *repetition*, which rephrases the segmentation problem in terms of the identification of recurring patterns; *homogeneity*, such as consistent timbre, the usage of certain harmonies or the presence of a specific instrument within each individual section; and *novelty*, where segment boundaries are expected to coincide with sudden changes in musical properties such as tempo, dynamics, or the musical keys. Not only music can be segmented according to different criteria, but also humans seem to combine different segmentation cues in an adaptive and subjective fashion when asked to recognise and derive structural information from music. As a consequence, the identification of segment boundaries is an ambiguous task even for music experts (McFee et al., 2017), and it is thus common for two listeners to disagree on the form of a piece of music. Indeed, since structural analysis records a listener's creative interpretation as much as their perception, objectivity is arguably an impossible goal for annotations (Smith et al., 2011).

The listener-dependent and context-sensitive relevance of different segmentation criteria discussed before make MSA an extremely challenging task when approached with computer-based systems. In this area (a.k.a. automatic music segmentation), the challenge is to design algorithmic procedures for automatically analysing musical form and several approaches have been developed and compared (Nieto and Bello, 2015). Because of this, in recent years it has become a focus for music information retrieval (MIR) researchers, not only due to its complexity, but also because it enables large-scale MSA and several practical applications. From a broad perspective, as outlined in (Müller et al., 2016), computational methods for structuring and decomposing digitised artefacts into semantically meaningful units are relevant not only for music content, but also for general multimedia content such as speech and video. Indeed, decomposing a complex object into smaller units is of practical importance for several content-specific applications, as it often constitutes the first step for simplifying subsequent processing and analysis tasks. The structural decomposition makes it possible to obtain compact object descriptions that can be efficiently stored, queried and transmitted, and it opens up novel ways for users to find and access music information in large, unstructured and distributed multimedia collections. Towards this direction, Kurth et al. (2005) introduced *SyncPlayer*, a framework for accessing music-related content that combines different modalities like acoustic, graphical and textual representations obtained from running music- and text-based retrieval methods on every track in a music database. Another priming example of application exploiting automatic MSA for content-based navigation is *SmartMusicKIOSK* (Goto, 2006), providing a music interface that enable users to visualise the structure of a song via a music map, which is designed to highlight the chorus and the most repetitive sections. This functionality is particularly useful for trial listening, when users need to quickly determine whether they like a specific selection of music (e.g. recommended by a system, or simply discovered), so that they will listen to the filtered selection from start to finish. In that case, which is quite common in popular music, users tend to jump to the the most salient parts of a song if an audio thumbnail or summary of that track is not available. To facilitate this process, *SmartMusicKIOSK* augments within-song browsing so that users can skip sections of a song by interactively changing the playback position while viewing the music map (Goto, 2006). This tool, originally designed for record shops, is now part of a larger system called *Songle* (Goto et al., 2011), a web service that estimates not only the music structure but also the melody line, the beat structure, and the chords of songs available on the web and visualises all of them in a synchronised way during playback. This family of tools are called active-music listening interfaces, as they allow users to enjoy music in more active ways than conventional playback. We refer to (Goto and Dannenberg, 2018) for a comprehensive overview of these interfaces.

This is also valid for music producers, when automatic procedures for MSA will be integrated into digital audio workstations to help them navigating around a particular track in an ongoing project. Besides improving the workflow of music production, MSA algorithms can also facilitate the creation of mash-ups and remixes, with the potential to be exploited by DJs during live sessions. A notable example implementing this creative use case is the *Unmixer* (Smith et al., 2019), a recently introduced web service that also utilises a source separation method to enable users to extract loops from one or more uploaded tracks, remix them, and create mash-ups of loops from different songs.

As for scientific applications, the ability to reduce the complexity of music can benefit different tasks in the field of MIR, such as beat tracking, audio thumbnailing and automatic music composition, other than providing a framework for testing theories of music perception. Indeed, if we can identify the structure of a piece of music, at least in terms of phrases and measures, most work of beat tracking is done considering that, where repetitions occur in music, the beats in the two repetitions should correspond (Dannenberg, 1983). In the context of music generation, since there is a close relation between the cognitive tasks of music analysis and composition, an MSA engine could form the basis of an algorithmic composition system, especially considering the lack of long-term structure in

music generated by automatic procedures.

Nonetheless, most of the current methods in computational MSA can only estimate large-scale structural patterns from music, typically corresponding to the sectional level of a composition. The algorithms following this approach (a.k.a. flat segmentation) can only produce a single-level segmentation of a piece of music, resulting in a sequence of non-overlapping segments that are expected to match the sections of the given track. In contrast, hierarchical segmentation techniques for MSA take into account the hierarchical organisation of music and can detect structural patterns related to musical form at different time scales. These methods produce multi-level segmentations in the form of hierarchies, where each level offers a segmentation of the given piece at a specific level of granularity. For instance, a divisive method for hierarchical MSA might identify the sectional patterns of a composition at the first level of the hierarchy; then, each section would be decomposed into smaller patterns in order to reveal sub-structures. This process is repeated recursively until the piece is completely decomposed into its most granular components (e.g. measures, notes, or beat-aggregated audio features) or a particular stopping criterion is met during segmentation. In this way, each level in the hierarchies offers a segmentation which is the result of a refinement of the segmentation performed at the previous level.

1.1 Our contribution

In this paper, we introduce MSCOM, a novel MSA algorithm that performs hierarchical segmentation of music at multiple resolution levels in order to detect structural patterns of variable different length and complexity. Our method addresses the music segmentation task as a community detection problem. First, a given music piece is partitioned into a number of consecutive audio frames, and an indirect graph reflecting both local temporal connectivity and long-term recurrence information is generated from these frames. The graph is then processed by a divisive community detection procedure based on modularity optimisation (Newman, 2004a) at multiple resolution levels (Arenas et al., 2008), which yields a structural hierarchy whereby the first level contains a single segment embracing the whole piece and the deepest level contains as many segments as the number of frames. We test and validate MSCOM on the structural annotations for large amounts of music information (SALAMI) dataset (Smith et al., 2011). By comparing our results to state of the art methodologies for hierarchical segmentation, we show that our method outperforms current approaches for hierarchical music segmentation. Furthermore, we show that MSCOM can also enable the visualisation and the analysis of musical structure at finer levels of detail, whereby tree representations of music (Lerdahl and Jackendoff, 1985) can be further enriched to reflect more structural relationships as in (Paul Lamere, 2000) and (Martin Wattenberg, 2000).

The rest of this paper is organised as follows. Section 2 introduces related work in MSA and Section 3 outlines our methodology. In Sections 4 and 5, we compare our algorithm to the current state of the art techniques in terms of segmentation accuracy and we analyse some key properties of the resulting segmentations. Finally, we summarise our findings and provide an outlook in Section 6.

2. Background

2.1 From feature extraction to music segmentation

The first step in the pipeline of an automatic procedure for audio-based MSA consists in choosing and extracting acoustic features which are related to those human observe when determining the musical form of a piece. As observed in (Bruderer et al., 2006), the instrumentation and the timbral properties of a sound source are of great importance for the human perception of musical structure, and the same can be said for the pitch content on which harmonic and melodic sequences are built upon (Paulus et al., 2010a). Therefore, two different types of audio features are often considered: *mel-frequency ceptral coefficients* (MFCC), encoding the timbral properties of the signal; and *chroma features* or *pitch class profiles*, describing the distribution of the harmonic content of the spectrum into a fixed number of bins corresponding to pitches of a musical scale. Considering that harmonic features alone have turned out to be effective mid-level representations in the context of MSA (Gómez, 2006; Bartsch and Wakefield, 2005), most of the works in the literature are based on the extraction and the subsequent analysis of chroma-based features, without taking into account the timbral properties of a recording.

However, by studying the relationship between different audio features and human annotations, Smith and Chew (2013) demonstrated that a listener's attention shifts among these features throughout a piece, and using a single audio descriptor would thus lead to undetected structural boundaries.

2.1.1 Representing musical structure

As similarity is a key element for detecting music form, the audio features of a given audio recording $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$ are then compared with each other in a pairwise manner. Comparing feature vectors $\mathbf{x}_i$, $\mathbf{x}_j$ is done via a similarity function s (e.g. based on the Euclidean or cosine distance) which enables the computation of a square and symmetric matrix defined as $S(i,j) = s(\mathbf{x}_i, \mathbf{x}_j)$. This is called *self-similarity matrix* (SSM) and was introduced in the music field by Foote (1999) to visualise the musical structure of a given track. If a musical segment of length m, starting and ending at times t_i and t_{i+m} respectively, is repeated at a later time t_j, then the sub-sequence of feature vectors

$[\mathbf{x}_i,\ldots,\mathbf{x}_{i+m}]$ should be identical to $[\mathbf{x}_j,\ldots,\mathbf{x}_{j+m}]$, with the corresponding entries in the SSM being maximised w.r.t. the co-domain of the chosen similarity function. From a visual perspective, this is represented as two stripes running across the main diagonal in the SSM.

The SSM is the starting point of most of the automatic procedures for MSA, and several works attempt at enhancing this musical structure representation in order to facilitate the subsequent extraction of structural patterns. Common enhancements of an SSM include those making it robust against musical variations such as tempo changes and transpositions (Smith and Chew, 2013; Müller and Kurth, 2006).

2.1.2 Structural analysis methods

The current methods in the MSA literature are often categorised according to the taxonomy proposed by Paulus et al. (2010a), which is based on the segmentation principles of repetition, novelty and homogeneity. The approaches belonging to the first group are based on the intuition that structural patterns usually repeat throughout a piece. As repeated segments are visualised as stripes on the diagonal and off-diagonals in an SSM, these approaches mostly rely on the extraction of such visual patterns (Müller and Kurth, 2006; Lu et al., 2004). These methods seem to be more indicated for popular music and also for the identification of the most representative segments of a recording (Goto, 2006). Novelty-based approaches instead aim at detecting the boundaries between two contrasting musical ideas. As proposed by Foote (2000), this is achieved with a novelty function computed by convoluting an SSM with a short-time checkerboard kernel. Peaks of the so-defined novelty function indicate potential structural boundaries. Finally, homogeneity-based methods assume a strong inner acoustical integrity of segments with respect to the chosen musical features (Jensen, 2006). Homogeneous passages are associated to block-like regions in an SSM, and are usually detected by utilising clustering algorithms. In case a functional role is assigned to each homogeneous group, another approach consists in defining the structural segments as the states of a hidden markov model (HMM) (Levy and Sandler, 2008; Peeters, 2003).

In addition, fusion methods take the best of both worlds by combining different segmentation principles with a single framework. For instance, Paulus and Klapuri (2009) attempt to capture homogeneity and repetition properties by a single probabilistic fitness measure, whereas Kaiser and Peeters (2013) apply a late fusion strategy on the segmentations produced by a repetition- and a novelty- based approach.

2.2 Unsupervised methods for hierarchical MSA

To the best of our knowledge, there are only two automatic methods that are capable to produce hierarchical segmentations (both of which are part of the music structure analysis framework (MSAF) (Nieto and Bello, 2015)) – the *Laplacian Structural Decomposition* (LSD) (McFee and Ellis, 2014a), for both boundary detection and structural grouping, and the *Ordinal Linear Discriminant Analysis* (OLDA) (McFee and Ellis, 2014b) that can only perform the former sub-task.

2.2.1 LSD

LSD generates hierarchies of fixed depth, where each layer i consists of $i+1$ unique segment labels. For each layer, this method first partitions the recording into a set of discontinuous clusters (segment labels), and then estimates segment boundaries according to changes in cluster membership between successive time instants. From a technical perspective, a network representing both the timbral and the harmonic similarities of a track is constructed. Then, after the computation of the Laplacian matrix from the adjacency matrix representing that graph, spectral clustering is performed on this structure. This is achieved by isolating the first k eigenvectors of the Laplacian matrix and running the *k-means* clustering algorithm on the resulting data. This allows the detection of k uniquely labelled segments in a track, so that repeating the same procedure for successive values of k, e.g. $k = 1,2,\ldots,10$, as done in (McFee and Ellis, 2014a), would produce a hierarchical segmentation of the corresponding track. LSD has two main limitations, which emerge from this incremental approach:

- although a level can have an arbitrary number of segments, the actual number of unique segment labels is always fixed, and it corresponds to the value chosen for k at that level;
- the resulting segmentation is not truly hierarchical due to the independence between levels; each one is indeed obtained from a separate clustering step, thus, a segmentation at the k-th levels is not necessarily a specialisation of the one at $k-1$.

2.2.2 OLDA

The OLDA method (McFee and Ellis, 2014b) performs agglomerative clustering of time instants into segments, resulting in a binary tree with time instants at the leaves, and the entire recording at the root. Each layer i has $i+1$ contiguous segments, and the tree is automatically pruned based on the statistics of segment lengths and the overall track duration. This results in a hierarchy of variable depth, typically between 15 and 30 levels, where each level can be seen as splitting one segment from the previous level into two. Because OLDA only estimates segment boundaries, it has to be coupled with a structural grouping algorithm in order to label the detected segments. According to (Nieto and Bello, 2015; McFee et al., 2017), the 2D-Fourier magnitude coefficients method (Nieto and Bello, 2014) is the preferred choice, since it yields state of the art results in automatic prediction of flat segment labels.

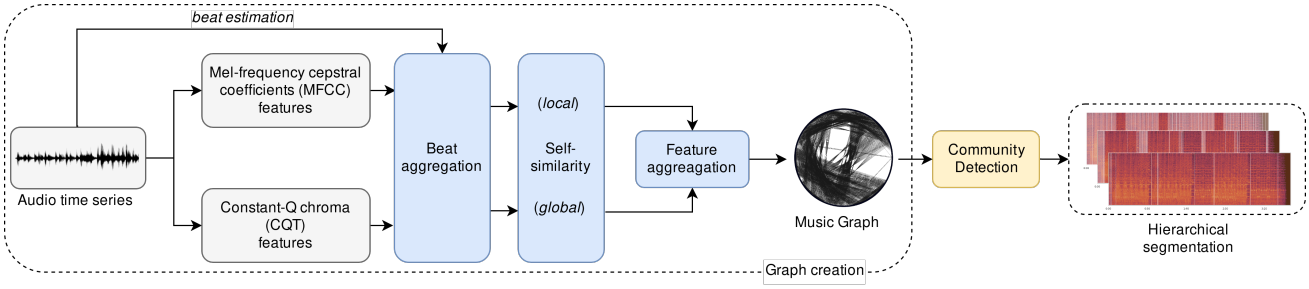

Figure 1: Schematic overview of MSCOM with all the main steps of its workflow.

3. The MSCOM algorithm

Our approach for hierarchical segmentation of music inherits some technical aspects from the LSD method outlined in the previous section, and it relies on the intuition that human-composed music has the kind of modular structure that can be uncovered by using multi-resolution community detection algorithms. By exploiting the method of McFee and Ellis (2014a) for the creation of a graph encoding both the temporal and the similarity dependencies among sample frames of a musical piece, we apply a hierarchical clustering procedure in order to detect structural communities at different resolution levels from the so-obtained graph.

We name our algorithm musical structure communities (MSCOM). A schematic illustration of MSCOM is shown in Figure 1 and a detailed technical description is provided in the following subsections.

In contrast to supervised methods based on convolutional neural networks (Grill and Schlüter, 2015), which can estimate fixed-depth segmentations based on the annotation levels in the SALAMI dataset, our algorithm is completely unsupervised. According to the taxonomies proposed by Peeters (2003) and Paulus et al. (2010b), MSCOM can be categorised as a homogeneity- or state-based procedure, identifying structural patterns based on the integrity of their acoustic features. Conversely to HMM-based methods such as (Paulus, 2010), MSCOM does not assign functional labels (e.g. verse, chorus) to the estimated segments. Our algorithm is not the first approach employing community detection techniques in MIR. More precisely, Serrà et al. (2012) investigated the use of community detection algorithms to improve the performance and the interpretability of cover identification methods. Gulati et al. (2016) applied community detection on structural segmentations of Iranian music to characterise the discovered patterns into rāga, composition-specific and gamaka motifs.

3.1 Music graph construction

The definition of the music graph that we use as part of our methodology is based on the work by McFee and Ellis (2014a). Basically, we create two distinct graphs capturing different perceptual aspects of a musical piece, and a combination of them is considered in order to obtain a single graph to process. First, two contrasting audio descriptors are extracted from a raw audio file – *harmonic features* (chroma features), for detecting long-range repeating forms, and *timbral features* (mel-frequency cepstral coefficients), for detecting local consistency. In order to reduce data dimensionality and remove transient noise, we beat-synchronise both the features by averaging all the vectors belonging to the same estimated beat. In this way, we obtain two time series feature matrices $p = \{p_1, p_2, \ldots, p_N\} \in \mathbb{R}^{d_p \times N}$ and $M = \{m_1, m_2, \ldots, m_N\} \in \mathbb{R}^{d_m \times N}$ denoting respectively the beat-synchronised chromagram and the sequence of mel-frequency cepstral coefficients (MFCC) with corresponding dimensions d_p and d_m and length equal to N – the number of estimated beats.

From the beat-synchronised feature matrices, we compute their self-similarity matrices S^P and S^M, both of dimension $N \times N$, with a similarity function (e.g. a gaussian kernel) over feature vectors, s.t. $0 \leq S^F_{ij} \leq 1$ denotes a non-negative affinity between feature vectors f_i and f_j. Since similarity is maximal between identical frames, we zero out the main diagonal of both matrices. As outlined in Section 2, MFCC and chroma-based features are both related to human perception of musical structure. Combining them into a single representation would thus provide a rich and informative descriptor that allow to detect structural patterns from different musical properties of a track.

The obtained matrices allow to define the recurrence graph R and the proximity graph Δ as follows:

$$R_{ij}(P) = \begin{cases} S^P_{ij} & p_i, p_j \text{ are mutual k-nearest neighbours} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$\Delta_{ij}(M) = \begin{cases} S^M_{ij} & |i-j| = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $k > 0$ parameterise the degree of connectivity. With this approach, beat-synchronised feature vectors are considered as nodes in the graphs, and the strength of each connection (according to the constraints imposed above) depends on the similarity between the associated nodes. As can be seen in Figure 2, the recurrence graph captures harmonic and melodic repetitions in a given track, whereas the prox-

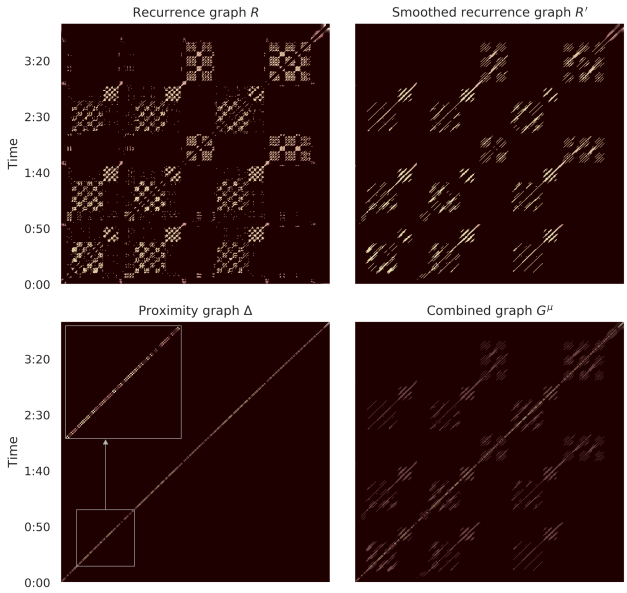

Figure 2: The main steps detailed in Section 3.1 for the creation of the music graph for the track "SALAMI 676". The recurrence graph R computed on the chroma features and its smoothed version R' to enhance diagonal stripes are illustrated in the top quadrants. The bottom-left plot represents the proximity graph Δ with a zoomed area highlighting its upper and lower off-diagonals that ensure the linkage of nodes corresponding to temporally consecutive feature vectors. The final graph G^{μ} obtained from the fusion method outlined in Equation 4 is illustrated in the last quadrant.

imity graph is meant to preserve the sequential nature of music by connecting consecutive nodes according to their timbral consistency. More precisely, two nodes in the proximity graph are connected if and only if they correspond to two temporally consecutive beats, and the weight of that link, if present, is equal to the similarity between their MFCC feature vectors. This formulation is in line with the roles of temporal proximity and similarity in human perception of musical structure (Temperley, 2004).

A smoothing filter is then applied to R in order to suppress uninformative links and promote local connectivity in the graph by filling those gaps which can blur potential repetitions. This is a common enhancement strategy for SSM, as it emphasises repeated structures which would then appear as strong diagonal stripes. The smoothing filter is implemented by using a *windowed majority vote* to each diagonal of R as:

$$R'_{ij} = maj\{R_{i+t,j+t} \mid t \in -w, -w+1, \ldots, w\}, \qquad (3)$$

where $w > 0$ controls the minimum length for a repetition sequence in terms of number of beats.

As a final step, a single graph representing the piece is obtained by combining R and Δ. Since we need to guarantee that the proximity connections are not excessively outnumbered by the repetition connections (it would be difficult to partition the resulting graph into contiguous segments otherwise), the combination is parameterised by a factor $\mu \in [0,1]$ as follows:

$$G^{\mu}_{ij} = \mu R'_{ij} + (1-\mu)\Delta_{ij} \qquad (4)$$

Since McFee and Ellis (2014a) aim for a process that on average moves either in sequence or across repetitions with equal probability, μ is set to a value that lets the combination assign equal weight to the local and repetition edges. To summarise, the so obtained graph G^{μ} is considered as an adjacency matrix, where G^{μ}_{ij} denotes the similarity of the beat-synchronised audio features corresponding to nodes i and j. This graph is the starting point of our musical structure analysis procedure outlined next.

3.2 Community detection of structural patterns

We exploit the graph representation of a musical piece to detect structural patterns at different resolution levels. Each structural pattern can be considered as a segment collecting nodes with similar features, where the propensity of a node to be part of a certain group depends on the resolution level chosen at a certain stage of the analysis. In the domain of network analysis, groups are better known as communities. In the context of this work, each community corresponds to a structural pattern, collecting nodes with homogeneous musical properties at a certain resolution level.

From Figure 2, and most evidently from the recurrence graph in the top-right plot, we can identify different square-shaped structures varying in size. From a visual analysis, each square can be associated to a community and we can easily notice their nested nature, – communities being parts of larger communities, which is related to the hierarchical structure of music form (Section 1). Identifying these communities at different levels enables us to detect musical structures at different resolutions or scales. On a coarse level, SALAMI 676 seems to exhibit a sectional structure corresponding to the ABABCABC form. Ideally, in the higher levels of the resulting hierarchical segmentation, sections A, B and C would correspond to three different communities, which will be recursively decomposed in deeper levels to detect structural patterns of finer resolutions. Unfortunately, this implies that our method would not be able to detect strophic forms (e.g. AA, AAA).

Community detection is a challenging task in graph theory, since neither the size nor the number of communities is known in advance. The goal is to partition a graph into a number of communities such that the intra-community degree of similarity is higher than the inter-community level of similarity. Community detection techniques have particularly gained popularity in the domain of computer vision, and more specifically for unsupervised image segmentation (Mourchid et al.,

2016; Camilus and Govindan, 2012).

3.2.1 Definition of modularity

Since the definition of community is rather abstract and a graph can be partitioned in several different ways, Newman (2004a) proposed the notion of "meaningful communities". According to his proposal, a community is meaningful only if the number of intra-community edges, along with their weights, is considerably different than those being expected in a randomly linked set of nodes with similar properties. To quantify the meaningfulness of a partition for a given graph, Newman (2004a) introduced a quality function called modularity, which takes values in the interval $[0,1]$ (with higher values indicating a higher quality partition) and is formally defined as:

$$Q(W,C) = \frac{1}{2w}\sum_i\sum_j (w_{ij} - \frac{w_i w_j}{2w})\ \delta(C_i,C_j) \qquad (5)$$

where W is the adjacency matrix of a graph, C is a given partition of the graph into communities, w_{ij} is the weight of the link connecting node i to node j, $w_i = \sum_j w_{ij}$ is the strength of node i, $2w = \sum_i w_i$ is the total strength of the network, C_i is the community where node i belongs to, and $\delta(C_i,C_j)$ is the Kronecker delta function which returns 1 if nodes i and j are members of the same community, 0 otherwise. In case of an unweighted graph, i.e. $w_{ij} = 1 \forall i,j$, we replace the strength w_i of node i with its degree d_i (the number of edges adjacent to node i). However, the calculation of modularity is not applicable when the graph is not strongly connected, although it could be done separately for each connected component. This condition further motivates the use of the proximity graph, which ensures the strongly connectedness of G^μ.

A partition C of W is optimal if it maximises $Q(W,C)$, though modularity optimisation is an NP-problem and the computational cost of any brute force exploration is prohibited for graphs larger than a few tens of nodes (Duch and Arenas, 2005). To mitigate this problem, heuristics-based approaches for modularity maximisation have been proposed in literature (Newman, 2004a; Duch and Arenas, 2005; Pujol et al., 2006; Newman, 2004b; Clauset et al., 2004). Nonetheless, modularity has a natural resolution limit (Fortunato and Barthelemy, 2007), meaning that vanilla modularity optimisation cannot reveal nested communities beyond a certain resolution level. To overcome this problem, Arenas et al. (2008) introduced a method revealing the community structure of a graph at different resolution levels by manipulating the total strength $2w$ of a given graph. From a technical perspective, increasing the total strength $2w$ without affecting the main structural properties of a network is achieved through the introduction of self-loops with weights equal to r for all nodes in the original adjacency matrix. The adjacency matrix is then written as $W_r = W + rI$, were W is the original adjacency matrix, r defines the granularity level and I is the identity matrix. Hence, r can be thought of as a *resistance parameter*, defining the tendency of nodes to form communities as it allows to leverage the relative importance of any link in the graph: when $r = 0$, we only access community structures within the resolution limit; when $r < 0$ we can reveal *super-structures* beyond the natural resolution limit, since nodes are more reluctant to form small scale communities; finally, when $r > 0$ we increase the importance of any individual link so as to reveal *sub-structures*.

According to Arenas et al. (2008), the approximate value of r s.t. all nodes belong to a single community is $r_{min} = \frac{-2w}{N}$, where N is the number of nodes. In this setting, the average strength of all nodes is zero and the weight of the expected edge between any pair i,j increases so that any non-trivial community would appear to be no different from a randomly drawn network. All nodes are thus assigned to one community. In contrast, r_{max} is the smaller positive number that satisfies $w_{ij} < \frac{(w_i+r)(w_j+r)}{2w+Nr}\ \forall i \neq j$, with each node forming a single-member community called *singleton*.

3.2.2 Ensuring hierarchical consistency

Even though this approach enables the detection of communities at different levels of granularity, optimising modularity for different values of r does not guarantee a truly hierarchical partitioning of the graph (Lancichinetti and Fortunato, 2011). This means that every partition obtained by maximising $Q(W_r,C)$ for a certain value of r is independent from any other partitions of the same graph for a different value of r. The same issue arises when using the LSD method (McFee and Ellis, 2014a), ending up with multi-level segmentations which are not strictly hierarchical (Section 2.2.1).

The approach proposed by Granell et al. (2012) implements a hierarchical procedure that extends the multi-resolution method of Arenas et al. (2008) in order to ensure a tree-like structure of the obtained hierarchies. As outlined in the pseudocode below, we used this procedure to hierarchically segment a music graph G^μ without imposing any constraint that could bias or limit the detection of communities, in terms of their number and type, as well as the depth and topology of the resulting hierarchy.

The community detection procedure starts by setting r to r_{min}, s.t. the modularity optimisation process distributes all nodes in the same community, and an iterative process is repeated by increasing r until we obtain an optimal partition where every community becomes a singleton. At every step, we iterate over all communities identified at the previous step and we run the modularity optimisation algorithm on each community separately using the current value of the resis-

Algorithm 1 Hierarchical community detection

Given the $N \times N$ adjacency matrix W of a graph

$l \leftarrow 1$
$r \leftarrow \frac{-2w}{N}$
$W \leftarrow W + rI$
$C^l \leftarrow \{C_1^l = \{n_1, n_2, n_3, \ldots, n_N\}\}$
while $|C^l| < N$ **do**
    $l \leftarrow l + 1$ ▷ current level
    $C^l \leftarrow \{\,\}$
    for C_j^l **in** C^l **if** $|C_j^l| > 0$ **do**
        $P_{C_j^l} \leftarrow \{C_{j,1}^l, S_{j,2}^l, \ldots, C_{j,m}^l\} =$ optimal partition of $W[C_j^l; :]$
        $C^l \leftarrow C^l \cup P_{C_j^l}$
    end for
    $r \leftarrow r + \Delta r$
    $W \leftarrow W + rI$
end while

tance parameter r. This approach ensures a strictly hierarchical community structure which allow us to segment music in a consistent manner. At the end of every iteration, we increase r by Δr so that we can progressively access communities of higher granularity.

The increments of r are domain specific and hyper-parameter tuning techniques are necessary. In the context of this work, we measured the degree by which communities split into sub-communities as we increase r, and we found that increments of $\Delta r = 0.1$ enable to access meaningful communities at various resolutions. Values higher than 0.1 lead to information loss due to unidentified communities between consecutive levels in the hierarchies, while smaller values lead to minor variations of communities between successive levels.

Since we are still segmenting music from graph representations, as similarly done by the LSD method, segment boundaries are estimated according to changes in cluster membership between consecutive nodes.

3.3 Two variants of MSCOM: baseline and dynamic

As outlined in the previous subsections, our methodology is organised in two steps: (i) construction of the music graph from a given track; and (ii) hierarchical multi-resolution detection of communities from the resulting graph. Whereas the second part is left unchanged, we implemented two versions of MSCOM:

- *MSCOM baseline* (MSCOM): makes use of the graph construction procedure of McFee and Ellis (2014a), as explained in Section 3.1;
- *MSCOM dynamic* (*DMSCOM*): based on a slightly different procedure where the self-similarity matrix S^P computed on the beat-synchronised chroma features undergoes a dynamic filtration step before the construction of the recurrence graph R as in Equation 1; more precisely, depending on the total strength of the network, we filter out those edges in S^P that do not meet the following condition:

$$S_{ij}^P > \lambda^{-1} \frac{\sum_i^N \sum_j^N S_{ij}^P}{N(N-1)} \tag{6}$$

where N is the number of nodes in the graph, the numerator is the capacity of the network of a certain track, the denominator is the maximum possible capacity of the network, and λ is a positive constant to control the severity of the filtration.

The fraction above can be considered as the average weight of a link in the graph, with the value of λ^{-1} defining a relative threshold for retaining structurally meaningful connections. We found that $\lambda = 4$ enables a reasonable filtration for the SALAMI dataset. An adequate strategy for tuning this hyper-parameter is to identify the value of λ maximising the clustering coefficient of the resulting filtered graphs. This simple method can also be considered for each segmentation.

4. Methodology

To evaluate and validate our methodology, we designed a comparative experiment for the task of hierarchical segmentation of music with the current state of the art methods outlined in Section 2.2 (LSD and OLDA). The experimental design consisted in applying both these methods and the two variants of MSCOM on the SALAMI dataset (Section 4.1). Then, a quantitative evaluation of the estimated structural segmentations by means of a consolidated measure in literature of hierarchical MSA (Section 4.2) is carried out for each algorithm. This enables a consistent comparison of these segmentation methods in regard to their ability to detect musical structure accounting for its hierarchical nature. In the following subsections we describe the experimental setup, including a description of the SALAMI dataset, the evaluation metrics used, and the configuration of the algorithms being compared.

4.1 Dataset

We used the SALAMI dataset (Smith et al., 2011) – one of the largest collections of structurally annotated tracks for hierarchical (and flat) MSA. In particular, SALAMI includes hierarchical annotations for 1359 music tracks of a variety of music styles (e.g., jazz, blues, classical, western pop and rock, and non-western ("world") music). The annotations were manually produced by 8 different human experts, all pursuing graduate studies in either music theory or composition, and contain three levels of segmentations per track: *lower*, *upper*, and *function*. The *lower* level typically corresponds to short phrases, the *upper* one represents larger sections, and the *function* level applies semantic labels to large sections (e.g. "verse" or "chorus"). Since the boundaries of the function level often coincide with those of the coarse level, we do not consider the function level in this experiment.

4.1.1 *Data collection and pre-processing*

Whereas the structural annotations of the tracks in the SALAMI dataset are accessible from the project website[1], the audio files of most of these tracks are not publicly available due to copyright restrictions. Indeed, the music files used by SALAMI come from 4 distinct sources: Codaich[2]; the Internet Archive's live music collection[3]; the RWC music database[4]; and the Isophonics corpus[5]. Among these collections, only the Internet Archive can be freely accessed as per instructions given by the maintainers of the SALAMI project. In addition, we used the scripts provided by Smith[6] to download and process the audio files of those tracks from Codaich and Isophonics that can be retrieved from YouTube. This process led to the compilation of 780 audio files which were compared to the reference human annotations in the dataset in order to ensure a consistent alignment between tracks and segmentations. After this analysis, we discarded all tracks with: (i) hierarchically inconsistent annotations – where the low level segmentation is less specific than the high level one; (ii) duration different from the one reported in the related human annotation. After these steps we ended up with 737 annotated musical pieces, of which 412 from Codaich, 29 from Isophonics and 296 from the Internet Archive. A list of all the SALAMI tracks considered for our experiments is made available in the project repository that can be accessed from the URL provided in the reproducibility section.

4.1.2 *Automatic hierarchy expansion*

As mentioned before, it is common to consider only the *upper* and the *lower* segmentation levels in each reference annotation when evaluating automatic methods for hierarchical MSA. Hence, each reference hierarchy is originally composed of two distinct levels, whereas the hierarchies estimated by the automatic procedures contain several levels of segmentation.

Recently, McFee and Kinnaird (2019) introduced a method for automatically enriching structural annotations by inferring and thus expanding hierarchical information latently encoded in the original segment labels. Given a flat segmentation, their method operates by simultaneously contracting similarly labelled segments – those differing only for a variation marker such as A and A' – and refining segments with identical labels. The so-obtained contraction and refinement levels are combined with the original annotation into a hierarchical annotation: the first level is a contraction of the variation markers (e.g. two distinct segments labelled as A and A' will get the same label); the second level is the original annotation; and finally the third level is a refinement of the labels by making each instance of a label unique (e.g. two distinct segments with the same label A would get two unique labels a_i and a_j depending on their occurrence).

By exposing detailed structure latent in the annotations, the expansion method was demonstrated to allow structure comparison methods to more accurately asses similarity between human annotations. Therefore, we believe that this method would also ameliorate the evaluation of automatic methods for hierarchical MSA. For this purpose, we adapted the expansion method to work directly on hierarchical annotations, so as to obtain an extended structural hierarchy where: the first level, called *coarse*, is the contraction of the *upper* segmentation level; the second and third levels are the original annotation and correspond to the *upper* and *lower* levels; the fourth level, called *refined*, is obtained by applying the refinement procedure on the *lower* segmentation level. An example of extended hierarchical annotation is illustrated in Figure 3.

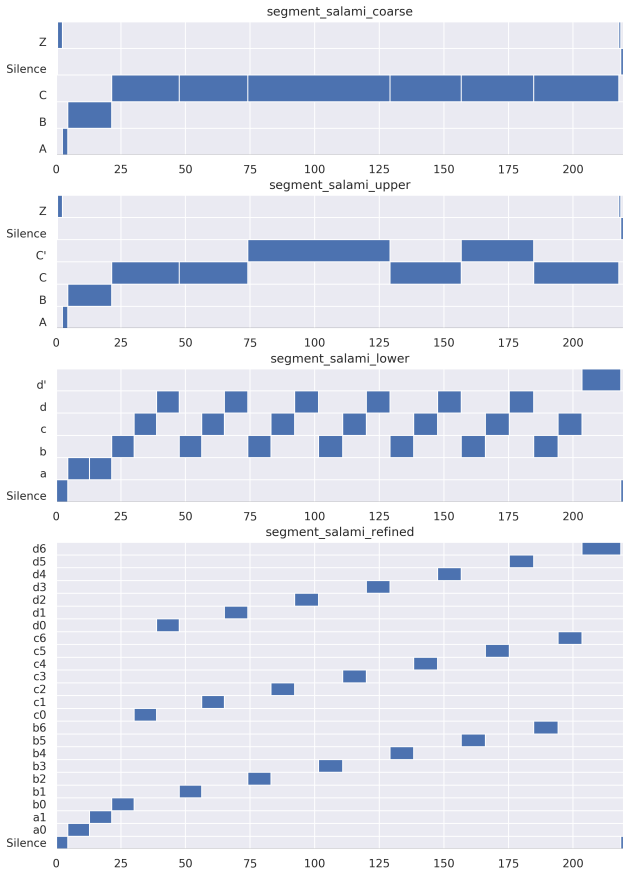

Figure 3: Hierachical expansion of the first human annotation of SALAMI 1094. The two segmentation levels denoted with the *upper* and *lower* tags define the original hierarchy, whereas the *coarse* and the *refined* levels are obtained by contracting the *upper* level and refining the *lower* level respectively.

4.2 Evaluation measures

The following subsections introduce the reader to the basic concepts of structural segmentation and provide a technical overview of the hierarchical evaluation measures used for this experiment.

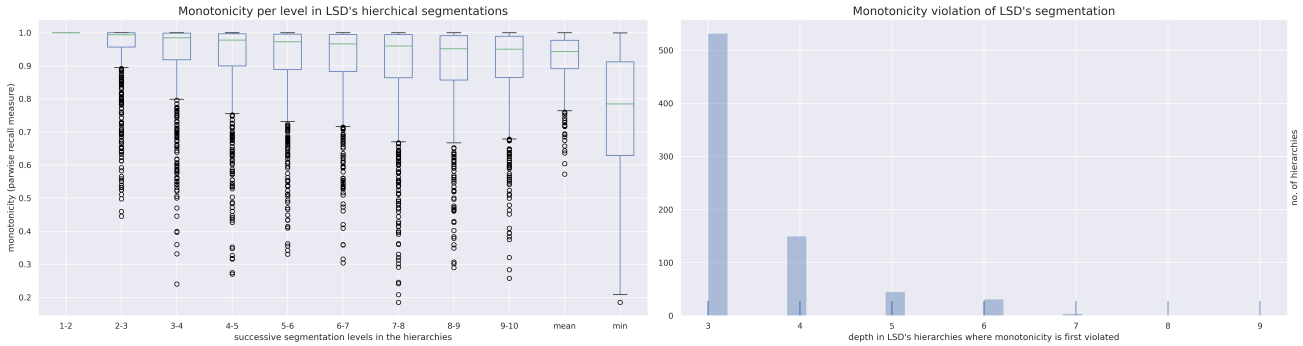

Figure 4: Analysis of monotonicity in LSD's hierarchical segmentations. Left: distribution of monotonicity for each couple of successive levels in the hierarchies estimated by LSD. Right: distribution of the level (or depth) in LSD's hierarchies at which maximum monotonicity is no longer preserved.

4.2.1 Preliminary concepts

Let $X = \{x_1, x_2, \ldots, x_T\}$ denote a set of sample frames generated from a given track at some fixed resolution, typically corresponding to 10Hz (i.e. 100ms-long frames). A flat *segmentation* S of X is defined by temporally partitioning X into a sequence of labelled time intervals, which are denoted as *segments*. Formally, a segmentation of X can be encoded as a mapping of samples $t \in [T] = \{1, 2, \ldots, T\}$ to a set of segment labels $Y = \{y_1, y_2, \ldots, y_k\}$, thus defining S as $S : [T] \to Y$. Depending on the dataset under analysis, Y may consist of functional labels, such as *intro*, *verse* and *chorus*, or section identifiers such as A and B.

A *segment boundary* is any time instant at the boundary between two segments. Let $S(i)$ identify the label of the segment containing the i-th frame in X, it usually corresponds to a change of label $S(t) \neq S(t+1)$ for $t > 1$, though boundaries between similarly labelled segments can also occur (e.g. an AA form).

As outlined before, comparing an estimated flat segmentation S^E with a reference one S^R can be done with different measures, evaluating either the agreement of segment boundaries (i.e. boundary detection) or that of segment labels (i.e. structural grouping). Nonetheless, these measures cannot be directly applied on a *hierarchical segmentation*, which is defined as a tree of flat segmentations:

$$H = (S_0, S_1, S_2, \ldots, S_m), \tag{7}$$

where each level is a refinement of the preceding layer with the ordering typically implying a coarse-to-fine structural analysis of the corresponding track. A hierarchical segmentation defined in this way is generally expected to be *monotonic*, meaning that a segmentation level S_l would only introduce new segment boundaries to those identified in S_{l-1}, while maintaining the labels of those segments that are not decomposed. Formally, we say that a hierarchy H is monotonic if for every level l, the following condition holds

$$S_l(u) = S_l(v) \implies S_{l-1}(u) = S_{l-1}(v). \tag{8}$$

This definition of monotonicity is binary, but it can be relaxed by instead measuring the proportion of time instants u and v where agreement at level l implies agreement at level $l-1$. As observed by McFee and Kinnaird (2019), this is calculated exactly by the pairwise recall measure (Levy and Sandler, 2008), when S_l is considered as the reference and S_{l-1} is the estimate segmentation. A measure close to 1 would express high monotonicity between two segmentation levels S_i, S_j (with 1 denoting full monotonicity), whereas lover values indicate violation of monotonicity.

4.2.2 The L-measure

The L-measure (McFee et al., 2017) is currently the only evaluation metric in literature that makes it possible to compare two hierarchical segmentations accounting for both the sub-tasks in MSA. To understand the intuition behind this metric, we need to introduce the concept of *meet* between sample frames. Formally, given a hierarchical segmentation H and sample frames u, v the *meet* of u and v under H is defined as:

$$M(u, v \mid H) = \max\ k \ \text{ s.t. } \ S_k(u) = S_k(v), \tag{9}$$

that corresponds to the deepest level in the hierarchy where the frames u and v receive the same label. As outlined by McFee et al. (2017), the *meet* induces a partial ordering over pairs of time instants: large values of $M(u, v \mid H)$ indicate a high degree of similarity and small values indicate low similarity.

To compare two hierarchical segmentations H^R and H^E, the L-measure is based on examining triplets of distinct time instants t, u, v in terms of the pairwise meets $M(t, u \mid H^R)$ and $M(t, v \mid H^R)$. In particular, the *reference comparison set* for a H is formulated as:

$$A(H) = \{(t, u, v) \mid M(t, u \mid H) > M(t, v \mid H)\}, \tag{10}$$

Therefore, $A(H)$ contains all those triplets of sample frames where (t, u) agree at a deeper level than the pair (t, v). Level-independent precision and recall scores can thus be obtained by comparing the size of the intersection to the reference comparison set:

$$\text{L-Precision}(H^R, H^E) = \frac{|A(H^R) \cap A(H^E)|}{|A(H^E)|} \tag{11}$$

$$\text{L-Recall}(H^R, H^E) = \frac{|A(H^R) \cap A(H^E)|}{|A(H^R)|} \tag{12}$$

These scores capture the rank-ordering of pairwise similarity between time instants. The final L-measure is computed as the harmonic mean of L-Precision and L-Recall. Rather than asking if an annotation describes two instants (u, v) as the same or different, these scores check whether (t, u) as more or less similar to each other than the pair (t, v), and whether that ordering is respected in both annotations (McFee et al., 2017).

4.3 Monotonicity of LSD's segmentations

As mentioned in Section 2.2, the hierarchies produced by LSD are not guaranteed to be monotonic, since each segmentation level S_l in a hierarchy H is the result of a separate clustering step where l distinct musical parts are identified. Therefore, it is possible that S_l does not preserve the segment boundaries detected in S_{l-1}. Nonetheless, McFee and Kinnaird (2019) pointed out that the definition of the L-measure is most intuitive when the underlying annotations are monotonic. Monotonicity though is not a strict requirement for the computation of this metric, as the L-measure depends on the maximum level of agreement between a pair of time instants. However, if a hierarchy is not monotonic, then the maximum level of agreement d between two time instants might not be consistent across the hierarchy if monotonicity is not preserved among each consecutive couple of preceding segmentation levels $(S_0, S_1), (S_1, S_2), \ldots, (S_{d-1}, S_d)$. As a consequence, the L-measure computed on non-monotonic hierarchies may lead to counterintuitive results.

To ensure a thorough comparison between the automatic segmentation methods considered in our experiments, we analysed the monotonicity of LSD's hierarchical segmentations on the SALAMI dataset. For this purpose, we followed the methodology introduced in Section 4.2.1, that is based on the level-by-level computation of the pairwise recall measure. Since the hierarchies produced by LSD consists of 10 segmentation levels, we obtain 9 measures of monotonicity for each hierarchy (Figure 4). From this analysis we found that the first two levels are always fully monotonic, whereas the average monotonicity tends to decrease as we move down through the hierarchies. As reported in Figure 4, most of the hierarchical segmentations violates full monotonicity already at the 3rd level, and none of them is fully monotonic until the 10th level.

For the sake of the evaluation, considering the previous remarks on the consistency of the L-measure, we cut LSD's hierarchies until their last fully monotonic level and we included them in our comparison together with the original (non monotonic) hierarchies. Each truncated hierarchy $H^t = (S_0, S_1, \ldots, S_l) \subseteq H = (S_0, S_1, \ldots, S_l, \ldots, S_m)$ satisfies the condition:

$$mono(S_i, S_{i+1}) = 1 \ \forall i < l \ \wedge \ mono(S_l, S_{l+1}) < 1$$

where $mono$ is the monotonicity function corresponding to the pairwise recall measure. We denote the resulting truncated hierarchies as LSDM, and we can consider this process as a refinement of the LSD method in order to ensure the monotonicity of the estimated hierarchical segmentations.

4.4 Experiments

Since only 476 (out of the 737) tracks in the SALAMI dataset are double-annotated, we limit our evaluation to the first annotation available for each track in order to make use of the whole database. To improve the ability of the L-measure to assess similarity between structural annotations, we also evaluate the segmentations yielded by each algorithm on the expanded reference hierarchies (as detailed in Section 4.1.2).

Our experiments are organised as follows:

1. Segmentation of the 737 SALAMI tracks with the algorithms under analysis – LSD, OLDA, MSCOM, DMSCOM – in order to obtain a structural hierarchy for each track and by each method.
2. Truncation of LSD's hierarchical segmentations in order to obtain fully-monotonic hierarchies that we denote as LSDM (Section 4.3).
3. Computation of the L-measures between each structural hierarchy estimated by an automatic procedure and the first reference hierarchy of the segmented track. This is done for both the original and the expanded reference hierarchies.

For the experiments conducted with the LSD and OLDA algorithms, we used the implementations that are freely available in MSAF (Nieto and Bello, 2015). As mentioned in Section 2.2, since OLDA can only perform hierarchical boundary detection, it is necessary to use an algorithm for structural grouping in order to perform a hierarchical segmentation. Similarly to (McFee et al., 2017), we used the 2D-Fourier magnitude coefficients method (Nieto and Bello, 2014) to estimate segment labels since it yields state of the art results in terms of automatic (flat) segment label prediction. For the sake of reproducibility of our experiments, we used the default hyper-parameters suggested in MSAF (Nieto and Bello, 2015), which were optimised by the authors on a collection of structurally annotated datasets (which include SALAMI).

Although the outcome of the algorithm depends on the quality of the affinity matrix described in Section

3.1, an advantage of MSCOM is that it does not require any extra hyper-parameters to be set for the further detection of community structures. In fact, the hierarchical multi-resolution community detection procedure does not depend on any particular specification that would considerably alter the result of the segmentation. Nonetheless, the user is free to define the step size for the increase of the resistance parameter r, which is currently set to 0.1 by default. Our method is implemented in Python 3.6[7] and uses librosa 0.6.2 (McFee et al., 2015) for audio feature extraction.

5. Results and analysis

5.1 Performance comparisons between all algorithms

The segmentation performance (L-measure, L-precision and L-recall) for each algorithm and experiment is shown in Table 1. From the results of the Kruskal-Wallis H-tests, we found that the distributions of the evaluations associated to the five automatic procedures under analysis differ significantly for all measures, for both the original (L-measure: $\chi^2(2) = 718.37$; L-precision: $\chi^2(2) = 264.07$; L-recall: $\chi^2(2) = 1087.59$) and the extended hierarchies (L-measure: $\chi^2(2) = 770.17$; L-precision: $\chi^2(2) = 281.13$; L-recall: $\chi^2(2) = 1144.48$). The p-values associated to these tests are all less than 0.0001. Post-hoc multiple comparisons (Kolmogorov-Smirnow tests) were then performed to detect significant differences between the performances of the various algorithms on each evaluation measure (Bonferroni corrections were applied to control for family-wise error rate of multiple comparisons). In particular, we statistically compared the performances of both our methods with those of the LSD and OLDA algorithms (individually for each performance measure and the original/extended reference hierarchies).

Results show (see Table 2) that DMSCOM performed statistically better than LSD, LSDM and OLDA in all metrics and hierarchies types (original or extended). MSCOM also outperformed OLDA and LSDM ($p < 0.001$ for all comparisons) in all metrics and hierarchies types (original or extended), but it only outperformed LSD in the L-precision measure for the extended hierarchies experiment. Finally, we also compared the performances of MSCOM with DMSCOM. As reported in the last row of Table 2, statistical analysis indicates that DSCOM outperforms MSCOM for both the L-measure and the L-precision, but not for L-recall.

In sum, our results show that DMSCOM has achieved the best results in all evaluation measures and hierarchy types considered in our experiments, and statistically outperformed all other algorithms with the exception of MSCOM on L-recall. MSCOM outperformed both OLDA and LSDM in all performance metrics and for both hierarchy types, but not LSD (with the exception of L-precision on the extended hierarchies experiments, where it performed statistically better).

It should also be noted that (see Table 1) the proposed experimental setup suggests that comparing estimated segmentations against the extended reference hierarchies (c.f. Section 4.1.2) leads to an increase of L-precision and a slight reduction of L-recall. Considering that the former increase is considerable, this translates in a greater L-measure, which is a more accurate estimate of the segmentation accuracy for hierarchical MSA, as advocated in (McFee and Kinnaird, 2019).

5.2 Hierarchical depth and performance degradation

As can be seen in Table 1, DMSCOM achieves larger average L-recall than L-precision. This result confirms prior observations of McFee et al. (2017) in regard to the tendency of automated methods to identify more detailed structures than were encoded by the human annotators. In fact, the segmentation algorithms under analysis produce much deeper hierarchies than the reference annotations. Considering that the reference annotations in the SALAMI dataset have fixed depth (see Section 4.1), this behaviour may be responsible for lower precision. However, the results obtained from this comparative experiment do not seem to indicate a trade-off between precision and recall as a function of hierarchy depth, as pointed out in (McFee et al., 2017). Indeed, the structural hierarchies produced by (D)MSCOM are much deeper than those estimated by OLDA, but still can achieve considerably higher L-precision and L-recall.

McFee et al. (2017) also noticed that another factor impacting on the low segmentation performance of the state of the art methods is their weakness in segmenting longer tracks. On such tracks, the existent algorithms are said to over-emphasise short discontinuities and otherwise label the remainder of the track as belonging primarily to one segment. Indeed, the hierarchical segmentation estimated by OLDA on SALAMI-1112 (one of the longest tracks in the dataset, at 713 seconds) achieves L- measure, precision and recall of 0.308967, 0.250026 and 0.404268 respectively, which are below the average performance of the segmenter for the corresponding metrics (Table 1). Instead, DMSCOM does not seem to have the same degradation pattern, and the segmentation of the same piece receives L- measure, precision and recall of 0.534016, 0.464445 and 0.628102. Notably, these results are more in line with the average performance of the algorithm.

To better understand this behaviour, we examined the trend of the evaluation results for both OLDA and DMSCOM as a function of track duration. This analysis is limited to these algorithms as they both estimate monotonic segmentations. As shown in Figure 5, the degradation trend of the segmentation performance is more evident (notice the line slope) for OLDA in both L- precision and recall. On the other hand, the same pattern is less evident for DMSCOM. We conjecture that the slightly decreasing trend in the evalua-

	Original reference hierarchies			Extended reference hierarchies		
	L-measure	L-precision	L-recall	L-measure	L-precision	L-recall
LSD	0.462 ± 0.128	0.394 ± 0.120	0.584 ± 0.150	0.480 ± 0.123	0.420 ± 0.120	0.577 ± 0.143
LSDM	0.301 ± 0.179	0.377 ± 0.158	0.289 ± 0.205	0.309 ± 0.179	0.402 ± 0.158	0.282 ± 0.194
OLDA	0.398 ± 0.101	0.325 ± 0.098	0.536 ± 0.111	0.415 ± 0.097	0.348 ± 0.098	0.531 ± 0.104
MSCOM	0.460 ± 0.112	0.382 ± 0.102	0.600 ± 0.135	0.478 ± 0.105	0.408 ± 0.098	0.593 ± 0.129
DMSCOM	**0.48** ± 0.111	**0.403** ± 0.103	**0.611** ± 0.133	**0.500** ± 0.104	**0.430** ± 0.100	**0.607** ± 0.127

Table 1: Overview of the segmentation performance – mean and standard deviation of the L-measures – of each algorithm under analysis with respect to the first reference annotation, provided for each track in the SALAMI dataset. The evaluation is performed for both the original (left) and the extended (right) reference hierarchies.

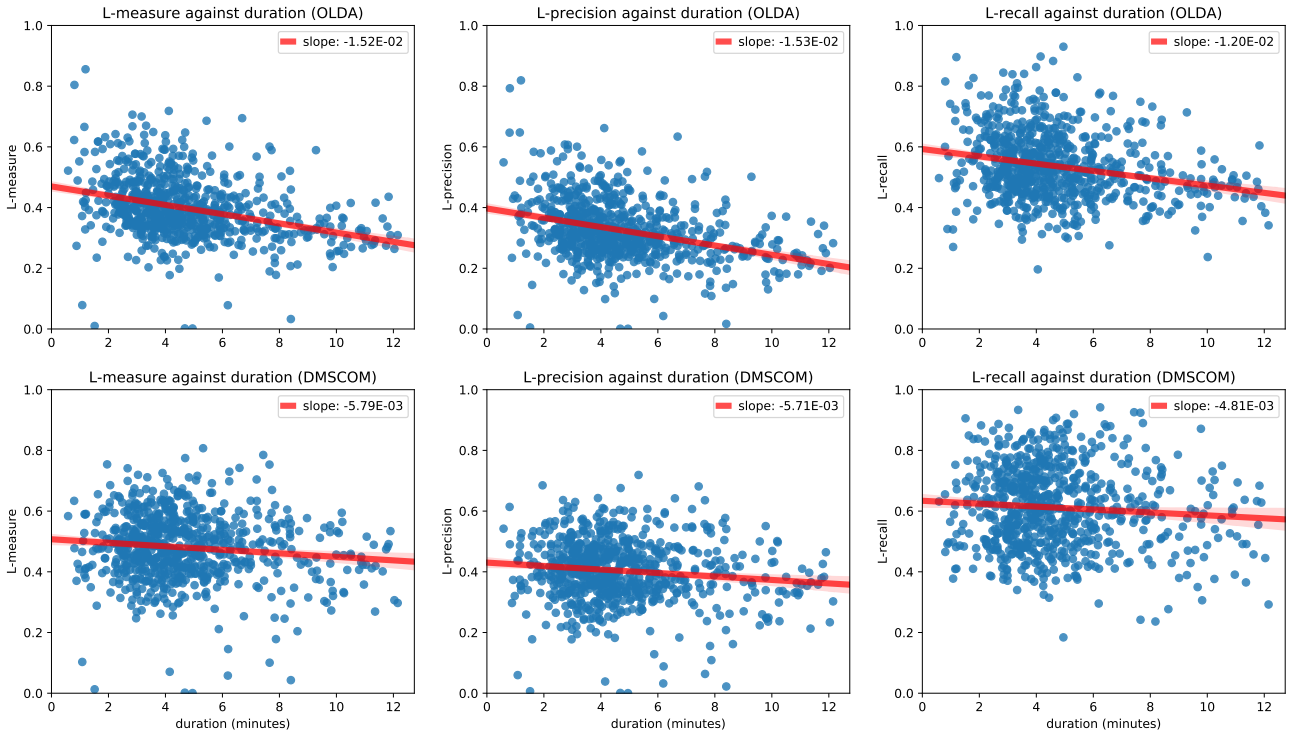

Figure 5: Segmentation performance degradation, in terms of the L measures outlined in Section 4.2, as function of track duration. The first row reports the trend for the evaluation of OLDA, whereas the second one is related to DMSCOM. A regression line is plotted along with the data to facilitate the comparison of the graphs.

tion metrics for DMSCOM might be due to the sparsity of long tracks in the SALAMI dataset partition used in our experiments. We observe that the performance of our method is less sensitive to the track length. Nonetheless, a closer analysis of this behaviour with more longer tracks available is needed to assess whether (D)MSCOM is insensitive to the duration of tracks.

5.3 Inferring segmentation accuracy

As suggested in (McFee et al., 2017), the segmentation accuracy of the automatic procedures can be estimated by assuming that the L-measures between different human annotations on the same tracks – a.k.a. *inter-annotator agreement* – define the upper limit for hierarchical MSA performance. For this purpose, we considered all those double-annotated tracks in SALAMI (476 out of 737), and for each of them we computed the L-measures between the two available human annotations. Results are shown in Table 3.

Comparing the L-measure of DMSCOM (0.48/0.5) with that corresponfing to the upper limits determined from human annotations (0.64/0.678), we can notice that there is still a considerable gap between human and automatic segmentation performance. Nevertheless, the L-recall achieved by DMSCOM (0.611/0.607) is not particularly far from the one corresponding to inter-annotator agreement (0.662/0.694). This result suggests that our method can detect most of the musical structures present in a piece of music, which is an important aspect in regard to the potential practical applications of a segmentation algorithm.

	L-measure		L-precision		L-recall	
	O	E	O	E	O	E
	MSCOM					
LSD	ns	ns	ns	*	ns	ns
LSDM	***	***	***	***	***	***
OLDA	***	***	***	***	***	***
	DMSCOM					
LSD	***	***	***	***	*	**
LSDM	***	***	***	***	***	***
OLDA	***	***	***	***	***	***
MSCOM	**	**	***	***	ns	ns

Table 2: Summary of the Kolmogorov-Smirnov statistical tests used to detect statistically significant differences between the algorithms performance on each evaluation metric. For each measure, 'O' denotes the evaluation performed on the original reference hierarchies, whereas 'E' refers to the extended counterpart. Note. *ns*: not significant, $p > 0.05$; * $p \leq 0.05$; ** $p \leq 0.01$; *** $p \leq 0.001$.

	Original hierarchies	Extended hierarchies
L-measure	0.64 ± 0.198	0.678 ± 0.168
L-precision	0.641 ± 0.197	0.683 ± 0.175
L-recall	0.662 ± 0.2	0.694 ± 0.174

Table 3: L-measures for the quantification of inter-annotator agreement: an upper limit for the segmentation performance of the automatic methods.

6. Discussion and conclusions

In the field of music information retrieval, the automatic detection of structural patterns in music is one of the most challenging task considering the current limitations of existent procedures. The problem is further compounded when the analysis of structure has to take into account the natural organisation of musical patterns as nested components in hierarchies: from sections to phrases, motives and so forth. To the best of our knowledge, there are only two algorithms in the literature which can segment music hierarchically, i.e. the ordinal linear discriminant analysis (OLDA) (McFee and Ellis, 2014b) and the Laplacian structural decomposition (LSD) method (McFee and Ellis, 2014a), with McFee et al. (2017) providing an evaluation of these procedures.

In this paper, we introduced MSCOM and DMSCOM, two novel automatic procedures that can segment music hierarchically and perform both boundary detection and structural grouping as a single task. Our approach for both the algorithms is organised in two steps: a music graph encoding the timbral and harmonic properties of a given track is constructed; and an algorithm for hierarchical multi-resolution community detection is applied on the resulting graph. This enables the detection of structural patterns in music at different resolution levels, which can even be more general (macro-structures) or more specific (micro-structures) than the corresponding human annotations.

We conducted a systematic and reproducible evaluation of (D)MSCOM and compared their performance to OLDA and LSD on a publicly available SALAMI dataset. Our results clearly demonstrate that DMSCOM outperforms other state of the art algorithms for MSA and achieves significantly better hierarchical segmentation performances.

We also demonstrated that our method also possesses key advantages over existent ones. First, the hierarchical segmentations yielded by (D)MSCOM are more robust to performance degradation associated with long tracks, a key limitation of existent approaches. Second, unlike existent state of the art algorithms, the increased depth of the hierarchies produced by (D)MSCOM does not compromise the precision of the segmenter. Third, (D)MSCOM is almost parameter-free, and this makes it potentially applicable to different styles and genres of music without the need to "tailor" its configuration for an optimal segmentation. Fourth, the hierarchies constructed by (D)MSCOM are deeper than the ones estimated by the other algorithms in the literature, since we do not restrict or limit the size and type of segments to detect nor the topology of the hierarchies.

In addition to the (D)MSCOM algorithms, we also proposed a revised methodology for the evaluation of hierarchical MSA algorithms, which leverages the recent methods for hierarchy expansion to enrich the reference annotations. Since this new method exploits the latent hierarchical information encoded in the annotations (and provides a more extensive comparison between estimated and human hierarchies), we envisage that this approach can become a standard for the evaluation of MSA algorithms.

Hierarchical multi-resolution procedures for the detection of structural communities in music tends to create deep hierarchies. Because of this, we are currently planning to devise more powerful methods that can analyse the generated hierarchies at all levels in order to make them more compact. This post-segmentation processing would make it easier and more intuitive for a human to inspect and interpret the resulting hierarchies. Another development that we are currently pursuing is the development of new quantitative measures of structural complexity in music from the estimated hierarchical segmentations, a technical challenge that would be of great interest for automatic music recommendation systems, automated music composition algorithms and to the field of Musicology (and particularly music analysis).

Reproducibility

The code of the segmentation algorithm will be soon uploaded on GitHub at `https://github.com/jonnybluesman/mscom.git`. The repository also includes all the instructions needed to obtain the dataset considered for our experiments, as well as the code to run our segmentation algorithms on arbitrary audio files. We also include a jupyter notebook with an example of structural segmentation produced by DMSCOM, which is further analysed in order to provide more insights on the segmentation process of our methods.

Competing interests

The authors have no competing interests to declare.

Notes

[1] Official repository of the SALAMI dataset: `https://github.com/DDMAL/SALAMI`

[2] `http://jmir.sourceforge.net/index_Codaich.html`

[3] `https://archive.org/`

[4] `https://staff.aist.go.jp/m.goto/RWC-MDB/`

[5] `http://isophonics.net/datasets`

[6] Repository of the SALAMI from Youtube project: `http://jblsmith.github.io/Getting-SALAMI-from-YouTube/`

[7] Python: an open source programming language. Python Software Foundation (www.python.org)

References

Arenas, A., Fernandez, A., and Gomez, S. (2008). Analysis of the structure of complex networks at different resolution levels. *New Journal of Physics*, 10(5):053039.

Bartsch, M. A. and Wakefield, G. H. (2005). Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on multimedia*, 7(1):96–104.

Bruderer, M. J., McKinney, M. F., and Kohlrausch, A. (2006). Structural boundary perception in popular music. In *ISMIR*, pages 198–201.

Camilus, K. S. and Govindan, V. (2012). A review on graph based segmentation. *International Journal of Image, Graphics & Signal Processing*, 4(5).

Clauset, A., Newman, M. E., and Moore, C. (2004). Finding community structure in very large networks. *Physical review E*, 70(6):066111.

Dannenberg, R. B. (1983). Toward automated holistic beat tracking, music analysis, and understanding.

Duch, J. and Arenas, A. (2005). Community detection in complex networks using extremal optimization. *Physical review E*, 72(2):027104.

Foote, J. (1999). Visualizing music and audio using self-similarity. In *Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, pages 77–80.

Foote, J. (2000). Automatic audio segmentation using a measure of audio novelty. In *2000 IEEE International Conference on Multimedia and Expo. ICME2000. Proceedings. Latest Advances in the Fast Changing World of Multimedia (Cat. No. 00TH8532)*, volume 1, pages 452–455. IEEE.

Fortunato, S. and Barthelemy, M. (2007). Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41.

Gómez, E. (2006). *Tonal description of music audio signals*. PhD thesis, Universitat Pompeu Fabra.

Goto, M. (2006). A chorus section detection method for musical audio signals and its application to a music listening station. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1783–1794.

Goto, M. and Dannenberg, R. B. (2018). Music interfaces based on automatic music signal analysis: new ways to create and listen to music. *IEEE SIgnal ProcESSIng MagazInE*, 36(1):74–81.

Goto, M., Yoshii, K., Fujihara, H., Mauch, M., and Nakano, T. (2011). Songle: a web service for active music listening improved by user contributions. In *ISMIR*, pages 311–316.

Granell, C., Gomez, S., and Arenas, A. (2012). Hierarchical multiresolution method to overcome the resolution limit in complex networks. *International Journal of Bifurcation and Chaos*, 22(07):1250171.

Grill, T. and Schlüter, J. (2015). Music boundary detection using neural networks on combined features and two-level annotations. In *ISMIR*, pages 531–537.

Gulati, S., Serra, J., Ishwar, V., and Serra, X. (2016). Discovering rāga motifs by characterizing communities in networks of melodic patterns. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 286–290. IEEE.

Jensen, K. (2006). Multiple scale music segmentation using rhythm, timbre, and harmony. *EURASIP Journal on Advances in Signal Processing*, 2007:1–11.

Kaiser, F. and Peeters, G. (2013). A simple fusion method of state and sequence segmentation for music structure discovery. In *ISMIR*, pages 257–262.

Kurth, F., Müller, M., Damm, D., Fremerey, C., Ribbrock, A., and Clausen, M. (2005). Syncplayer: an advanced system for multimodal music access. In *ISMIR*, volume 5, pages 381–388.

Lancichinetti, A. and Fortunato, S. (2011). Limits of modularity maximization in community detection. *Physical review E*, 84(6):066122.

Lerdahl, F. and Jackendoff, R. S. (1985). *A generative theory of tonal music*. MIT press.

Levy, M. and Sandler, M. (2008). Structural segmentation of musical audio by constrained clustering. *IEEE transactions on audio, speech, and language processing*, 16(2):318–326.

Lu, L., Wang, M., and Zhang, H.-J. (2004). Repeating pattern discovery and structure analysis from acoustic music data. In *Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval*, pages 275–282.

Martin Wattenberg (2000). The shape of song. `http://turbulence.org/Works/song/method/method.html`. Accessed: 2019-02-11.

McFee, B. and Ellis, D. (2014a). Analyzing song structure with spectral clustering. In *International Society for Music Information Retrieval (ISMIR)*, pages 405–410.

McFee, B. and Ellis, D. P. (2014b). Learning to segment songs with ordinal linear discriminant analysis. *Self*, 275:330.

McFee, B. and Kinnaird, K. M. (2019). Improving structure evaluation through automatic hierarchy expansion. In *International Society for Music Information Retrieval (ISMIR)*. *(forthcoming)*.

McFee, B., Nieto, O., Farbood, M. M., and Bello, J. P. (2017). Evaluating hierarchical structure in music annotations. *Frontiers in psychology*, 8:1337.

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., and Nieto, O. (2015). librosa: Audio and music signal analysis in python. In *Proceedings of the 14th Python in science conference*, pages 18–25.

Mourchid, Y., El Hassouni, M., and Cherifi, H. (2016). Image segmentation based on community detection approach. *The International Journal of Computer Information Systems and Industrial Management Applications. ISSN*, pages 2150–7988.

Müller, M. (2015). *Fundamentals of music processing: audio, analysis, algorithms, applications*. Springer.

Müller, M., Chew, E., and Bello, J. P. (2016). Computational music structure analysis (dagstuhl seminar 16092). In *Dagstuhl Reports*, volume 6. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Müller, M. and Kurth, F. (2006). Towards structural analysis of audio recordings in the presence of musical variations. *EURASIP Journal on Advances in Signal Processing*, 2007(1):089686.

Newman, M. E. (2004a). Analysis of weighted networks. *Physical review E*, 70(5):056131.

Newman, M. E. (2004b). Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133.

Nieto, O. and Bello, J. P. (2014). Music segment similarity using 2d-fourier magnitude coefficients. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 664–668. IEEE.

Nieto, O. and Bello, J. P. (2015). Msaf: Music structure analysis framework. In *International Society for Music Information Retrieval (ISMIR)*.

Paul Lamere (2000). The infinite jukebox. `http://infinitejukebox.playlistmachinery.com/`. Accessed: 2019-02-11.

Paulus, J. (2010). Improving markov model based music piece structure labelling with acoustic information. In *ISMIR*, pages 303–308.

Paulus, J. and Klapuri, A. (2009). Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(6):1159–1170.

Paulus, J., Müller, M., and Klapuri, A. (2010a). State of the art report: Audio-based music structure analysis. In *International Society for Music Information Retrieval (ISMIR)*, pages 625–636. Utrecht.

Paulus, J., Müller, M., and Klapuri, A. (2010b). State of the art report: audio-based music structure analysis. In *11th International Society for Music Information Retrieval Conference*, pages 625–636, Utrecht, Netherlands. ISMIR.

Peeters, G. (2003). Deriving musical structures from signal analysis for music audio summary generation:"sequence" and "state" approach. In *International Symposium on Computer Music Modeling and Retrieval*, pages 143–166. Springer.

Pujol, J. M., Béjar, J., and Delgado, J. (2006). Clustering algorithm for determining community structure in large networks. *Physical Review E*, 74(1):016107.

Serrà, J., Zanin, M., Herrera, P., and Serra, X. (2012). Characterization and exploitation of community structure in cover song networks. *Pattern Recognition Letters*, 33(9):1032–1041.

Smith, J., Kawasaki, Y., and Goto, M. (2019). Unmixer: an interface for extracting and remixing loops. In *20th International Society for Music Information Retrieval Conference*, pages 824–831, Delft, The Netherlands. ISMIR.

Smith, J. B. and Chew, E. (2013). Using quadratic programming to estimate feature relevance in structural analyses of music. In *Proceedings of the 21st ACM international conference on Multimedia*, pages 113–122.

Smith, J. B. L., Burgoyne, J. A., Fujinaga, I., De Roure, D., and Downie, J. S. (2011). Design and creation of a large-scale database of structural annotations. In *International Society for Music Information Retrieval (ISMIR)*, volume 11, pages 555–560. Miami, FL.

Temperley, D. (2004). *The cognition of basic musical structures*. MIT press.