TOPOLOGICAL FINGERPRINTS FOR AUDIO IDENTIFICATION [*]

WOJCIECH REISE [†], MARIA DOMINGUEZ [‡], XIMENA FERNÁNDEZ [§], HEATHER HARRINGTON [¶], AND MARIANO BEGUERISSE-DÍAZ [‖]

Abstract. [**MBD**: Re-visit in the end] We explore the application of topological data analysis for audio fingerprinting using persistent homology on spectral features of audio tracks. A fingerprint of an audio signal is a descriptive summary that encodes information to uniquely identify it. Audio fingerprinting enables commercial applications such as audio identification and cover song detection that underpins copyright protection in streaming platforms and other services. Using cubical complexes we extract topological features from audio files that are local, and can be used to quantify auditory similarity between different tracks. Using topological descriptors, we develop an audio-identification algorithm for pairwise comparisons of audio signals. We evaluate our method on a duplicate audio detection task, and find that topological audio ID achieves comparable performance to the leading method. Under certain obfuscations, our method achieves superior performance, which shows that methods based on topology can increase the robustness and reliability of audio identification systems.

Key words. Topological Data Analysis, Persistent Homology, Audio Identification, Signal Processing

AMS subject classifications. 55U99, 68U10, 68U99, 94A(??)

need to mention other TDA and music: arXiv:2204.09744

1. Introduction.

[**MBD**: It's important that we replace all instances of "songs" for "track" or "audio signal" because the method is general for signals with spectral representation, not just music]

[**MBD**: We need to have an introductory paragraph about audio ID, and one about TDA and PH, providing key references. Then we can go onto the rest of the intro] [**WR**:] In this work we introduce the use of topological data analysis (TDA) for audio identification (audio ID). Given a query audio track (e.g., a fragment of a song), the main task in audio ID is to identify matching tracks from a database. There are many variations of audio ID, most notably duplicate detection [1, 19, 32, 33], and cover identification [25, 28, 29]. [**MBD**: This needs a bit of an expansion]

Many tasks in audio analysis rely on spectral representations (i.e., time-frequency) of the audio such as spectrograms[**MBD**: Ref here]. A spectrogram S, (defined in section 2) is a matrix whose columns are 'local' Fourier decompositions of the audio signal, where an entry represents the intensity with which a specific frequency is present in a portion of the signal [12]. Spectrograms are often represented as heatmaps where the frequencies that appear in the audio at different times are visible. The patterns in the spectrogram correspond to auditory features that audio ID systems leverage to identify an audio query [1, 12, 32, 33]. Specifically, audio ID systems aim to extract 'fingerprints' from audio tracks, that can then be used for search and retrieval. Audio fingerprints are low-dimensional features, robust to obfuscations, and computed from spectrograms using image analysis techniques. For example, in Ref. [32], the

[*] Submitted to the editors DATE.

Funding: This work was conducted in the scope of a master dissertation, as a guest at University of Oxford, and an internship at Spotifys' London office.

[†] Spotify, EPFL (reisewojciech@gmail.com).

[‡] Spotify (mdominguez@spotify.com).

[§] Department of Mathematical Sciences, Durham University (ximena.l.fernandez@durham.ac.uk).

[¶] Mathematical Institute, University of Oxford (hharrington@maths.ox.ac.uk).

[‖] Spotify (marianob@spotify.com).

fingerprint of a track is a set of time-frequency triplets that correspond to salient points in the spectrogram. In Refs. [1, 33], time-windows (subsets of columns) from the spectrogram are decomposed using wavelets. The fingerprint of a window encodes the most-significant wavelet types from the decomposition.

Fingerprinting a set of tracks generates a database of keys, which can then be queried. Industrial audio ID methods usually consist of two steps: 1–vs–N and 1–vs–1[1]. During the first step, a set of candidate, similar audio tracks is retrieved from the database. The second, sometimes with additional comparisons, consists of deciding whether each candidate matches the query sample. This is a binary classification task, where a point is a pair of tracks, with the ground truth denoting whether they are duplicates. As an example, in Shazam [32], the 1–vs–N step returns the candidate which has the most 'aligned' fingerprints with the query snippet. The second step consists of estimating the significance of that matching. In Ref. [33], [**MBD**: Is this also Shazam or something else? maybe we can say a bit more about this implementation] [**WR**: This is something else: Shazam is wang_industrial-strength_2003] the first step has two parts: estimating a pool of candidates based on the number of matching fingerprints, and, producing an alignment score. The second step is a test of significance on the candidate with the best alignment score.

Topological data analysis is a collection of data analysis techniques, inspired by topological descriptors, like homotopy or homology groups [**MBD**: ref here]. The latter are invariant to many transformations, in particular to homeomorphisms like rotations, stretching or scaling of the underlying topological space [20]. Persistent homology (PH)— an extension of homology [7] has had successful applications [11, 16, 22], notably in time-series classification [17] or parameter inference [10]. An application of TDA to music analysis in Ref. [24] uses Takens' embeddings of waveforms of single musical notes and persistent rank functions to discriminate between musical instruments. Bendich et al represent an audio track (e.g., a song) as a point cloud, and then use principal component analysis and TDA to create a graph representation of the track which enables analysing its structure [2]. In these cases, a common assumption is that underlying the time-series of the waveform, there is a dynamical system, and its periodic nature is characterized by the persistent homology of its sliding window embedding. In the case of music, the waveform is a superposition of many changing signals (for example instruments).

In this work, we focus on the task of 1–vs–1audio identification using topological tools. Our main contribution is to demonstrate how homology can capture features of tracks relevant for audio identification. We extract audio fingerprints using persistent homology features of spectral representations of the audio, and use them to develop an identification algorithm. We compare the performance of our method under several obfuscation scenarios, and show that it performs comparably with established methods, and in certain scenarios it can attain better performance.

HERE

1.1. Outline.

After background on spectral representation in section 2, we propose the fingerprinting method section 3: a brief recall on persistent homology sections 3.1 and 3.2 is followed by how we derive robust characteristics from spectral features section 3.3. The track-level algorithm is introduced in section 3.4. The experimental results and the discussion follow in section 4 and section 5 respectively.

2. Spectral representations.

An audio signal $s \in \mathcal{C}([0,T];\mathbb{R})$ is a continuous function. It is often represented in the time-frequency domain, using the short-time

Fourier transform [12], defined in (2.1)

$$\mathcal{S}(t,f) = \int_{\mathbb{R}} s(\tau)\omega(t-\tau)\exp(-ift)d\tau, \tag{2.1}$$

where $\omega(t)$ is a bell-shaped window function, centred at zero with finite support.

In digital audio processing, the signal is a collection of samples $s = (s_i)_{i=1}^{N_s}$, where $N_s = Tf_s$ and f_s is the sampling rate. A spectral representation is obtained with the discrete short-time Fourier transform [12]

$$\hat{\mathcal{S}}(m,n) = \sum_{k=-\infty}^{\infty} s_k \omega_{k-hn} \exp\left(-ik\frac{mf_s}{N_\omega}\right), \tag{2.2}$$

where h is the hop size and $(\omega_k)_{k=0}^{N_\omega-1}$ is a discrete version of $\omega(t)$. We choose the popular Hann window [18], defined in (2.3).

$$w_k = \begin{cases} \frac{1}{2}\left(1-\cos\left(\frac{2\pi k}{N_\omega-1}\right)\right) & \text{if } 0 \leq k \leq N_\omega, \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

Finally, we define $S \in \mathbb{R}^{N\times M}$ to be the magnitude of the spectrogram $\hat{\mathcal{S}}$,

$$S_{m,n} = \left|\hat{\mathcal{S}}(m,n)\right|. \tag{2.4}$$

The entry $S_{i,j}$ is the intensity of frequency f_i in the spectral decomposition of the signal convolved with w centred at t_i. We will think of it as the loudness of a pitch of frequency f_i around the time t_i in the audio signal. We change the frequency scale of the spectrogram defined in (2.4) to the mel-scale, introduced in [27]. In this spectral representation, called the mel-spectrogram and shown in Figure 1, the frequency bins match the logarithmic frequency resolution of the human ear better than the equal-length, evenly-spaced bins of the short-time Fourier transform [27].

Because of the correspondence of visual patterns to auditory signals, we can cast the audio identification problem in terms of image comparison. Since the auditory features used for audio identification are usually local both in time and frequency, they correspond to small regions in the image. Since the image representation is not invariant to operations like tempo or pitch shifting, it is common to use fingerprints which are instead [1, 33]. We propose a topological fingerprinting technique.

In this work, we use audio files sampled at $44.1kHz$. Spectral features are obtained with an implementation [18] of the mel-spectrogram, with 128 frequency bins, a window length $N_w = 1024$ and a hop size $h = 256$. For audio signals of length $T = 30s$, the spectrogram has $Tf_s/h = 5168$ columns, and, given the fixed number of rows, the resulting mel-spectrogram is a matrix of size 128×5168, shown at the bottom of Figure 1.

3. Topological fingerprints from spectrograms. We propose a method to fingerprint audio signals. After computing the spectrogram, we segment it in time. Each such segment induces a cubical complex and a filter function. We compute the persistent homology groups of dimensions zero and one on all of the segments. We represent the resulting topological features as Betti curves and we call this collection of vectors the fingerprint of that track.

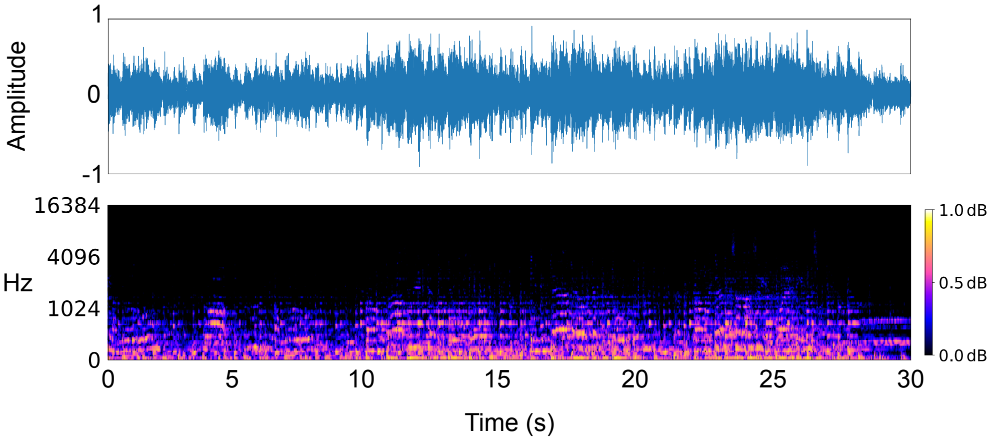
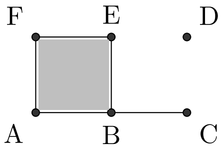


Fig. 1: Two representations of a 30-second fragment of the track The Morning, by Le Loup [15]. The top figure shows the graph of the digital audio signal in time, called the waveform. The bottom figure is the mel-spectrogram of that track - an image, where each column is a spectral decomposition of a short fragment of the track, convolved with a windowing function (2.3). The bottom rows of the spectrogram correspond to lower-frequency sounds and depict the rhythm.

3.1. Homology and persistent homology. Introduction to TDA is to be found in [21, 6], cubical complexes are studied in detail in [13], and for the algorithmic details, see [31, 6].

Homology groups provide some description of a topological space X. For example, the zero-, and one-dimensional homology groups, denoted $H_0(X)$, $H_1(X)$, correspond to connected components and holes, respectively. Formally, homology relies on the concept of chain complexes $(C_k(X))_{k\in\mathbb{N}}$, linked by homomorphisms $\partial_k : C_k(X) \to C_{k+1}(X)$ called boundary operators. We call the k-dimensional cycles elements of the kernel of ∂_k, denoted by $\ker(\partial_k)$ and boundaries the elements of $C_k(X)$ which are in the image of ∂_{k+1}, denoted by $\mathrm{im}(\partial_{k+1})$. The k-dimensional homology group is then the quotient of k-cycles by k-boundaries $H_k(X) = \ker(\partial_k)/\mathrm{im}(\partial_{k+1})$. Hence, in that interpretation, a non-trivial class in $H_k(X)$ is a k-cycle, which is not the boundary of a $k+1$-dimensional structure. An example of a cubical complex X is



(3.1)

Here X is a collection of vertices, segments and 2-cubes. If we denote by AB the edge between A and B, the formal sum $c = AB + BE + EF + FA$ is a cycle and also the boundary of the 2-cube $ABEF$. Hence, even though c is a 1-cycle, it is homologically trivial since it is also the boundary of a cube of a higher dimension. In $H_0(X)$, there are two distinct equivalent classes: one comprised of the vertex D and one of all the others.

Persistent homology [35] is an extension of homology, which applies to collections of spaces, instead of a single space. Consider an ordered set $(S, \leq)$ and a family of

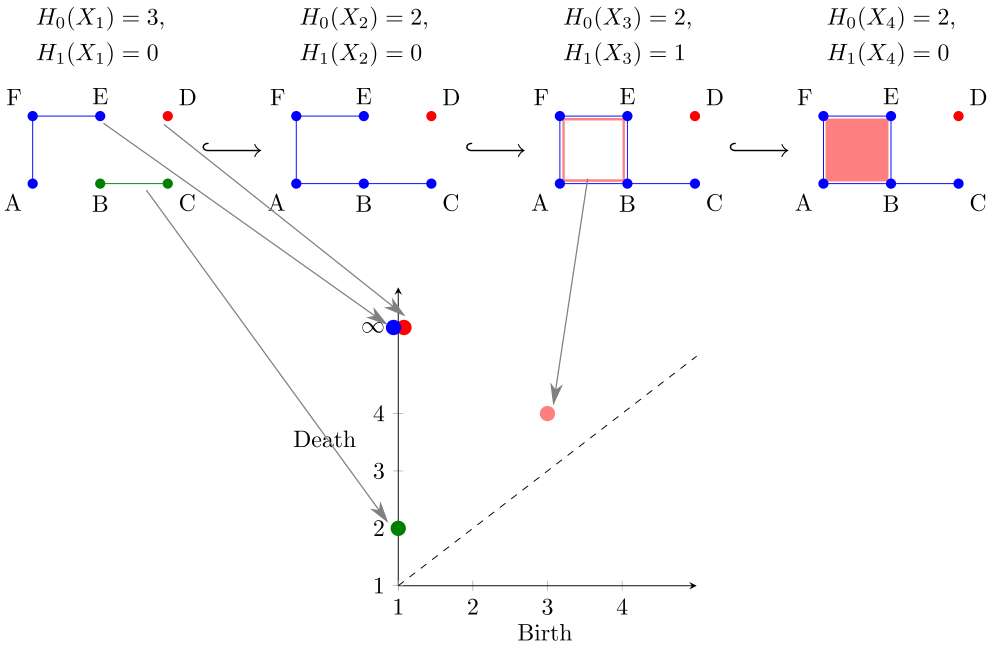


Fig. 2: **Top:** A filtration $(X_s)_{s=1}^4$ of the cubical complex X and the Betti numbers - the dimensions of the homology groups of each of the spaces. X_1 has no cycles and 3 connected components. By adding the 1-cube AB, the two corresponding homology classes become homologous. The first homology group $H_1(X_3)$ becomes non-trivial after the addition of BE, which creates the non-trivial cycle $c = AB+BE+EF+FA$. It is in turn contractible in X_4, when $ABEF$ appears. **Bottom:** The persistence diagrams for the filtration in Figure 2. The persistence diagram of dimension 0 is comprised of 2 points: $(0, \infty)$ with multiplicity 2 and $(1, 2)$ with multiplicity 1. The diagram of dimension 1 has a unique point $(3, 4)$. The dashed diagonal corresponds to points for which the birth is equal to the death: it is formally added to persistence diagrams, with infinite multiplicity.

spaces $(X_s)_{s\in\mathcal{S}}$. We call it a filtration, if it preserves the order induced by inclusions $s \leq s' \implies X_s \subseteq X_{s'}$ [35]. The inclusion morphisms $X_s \hookrightarrow X_{s'}$ induce morphisms between homology groups $\iota_{s,t} : H_k(X_s) \to H_k(X_t)$, that we can calculate for each space. We are particularly interested in the cases when $\iota_{s,t}$ is not an isomorphism, as this implies either the creation or annihilation of a non-trivial homology class.

In applications, since S is finite, we often suppose $S \subset \mathbb{N}$. Then, for a not-surjective (or non-injective) map ι_s^{s+1}, we say that a homology class is born (or dies) at X_{s+1}. We call $s + 1$ a birth (or death) value.

The birth and death values are paired with persistent homology, and summarize the information it contains [5]. One representation is the persistent diagram — a multi-set of birth-death pairs (b, d), with the diagonal (s, s), $s \in S$ with infinite multiplicity. An example of persistence diagrams for persistent homology groups of dimensions 0 and 1 is shown in Figure 2.

3.2. Persistent homology of cubical complexes. We have introduced homology, in the general setting of topological spaces. In applications, we often work

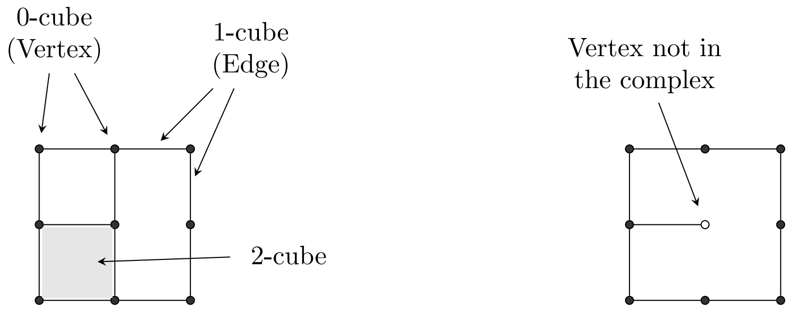


Fig. 3: On the left, an example of a collection of cubes of different dimensions, where two 0-cubes, two 1-cubes and one 2-cube are annotated. While the cubes are of different dimensions (0, 1 or 2), they share the embedding dimension 2. This collection is a cubical complex. On the right, a collection of cubes that is not a cubical complex. The annotated vertex is not in the collection, but the 1-cube incident to it is, violating the first property from Definition 3.1.

with discrete structures, what makes the homology computations tractable. The method we propose uses cubical complexes [13] - collections of cubes of different dimensions, depicted in Figure 3. A cube $Q = I_1 \times \ldots \times I_d$ is a product of elementary intervals $I_1, \ldots, I_d$ of the form $[a, a]$, $[a, a+1]$, $a \in \mathbb{Z}$. We say that

- d is the embedding number of Q,
- $\dim(Q) = |\{l \mid I_l \text{ is not degenerate}\}|$ is the dimension of Q,
- Q is called a vertex if $\dim(Q) = 0$.

Cubes have geometric faces. A cube Q_2 is said to be a face of a cube Q if $Q_2 \subset Q$. Moreover, if $\dim(Q_2) = \dim(Q) - 1$, Q_2 is a proper face of Q.

Definition 3.1. *Let K be a collection of cubes of the same embedding dimension. Then, K is a cubical complex if*

- *for any cube $Q \in K$, its faces are also in K,*
- *for all cubes $Q_1, Q_2 \in K$, the intersection $Q_1 \cap Q_2 \in K$ is either empty or a face of Q_1 and Q_2,*

Examples of a cubical cubical complex and a collection of cubes that is not a cubical complex are presented in Figure 3.

We work with cubical complexes, because they mimic the structure of spectrograms or matrices. There are two concurring ways of representing an image as a cubical complex [9]: T-construction [34] and the V-constuction [23]. In the former, each pixel corresponds to a two-cell in the complex, while, in the latter, each pixel is a vertex, and a higher dimensional structure is built on it. The difference between the two constructions is illustrated in Figure 4. We choose the vertex-construction, which reflects the proximity in the spectral domain more than the top-cell construction: only neighbouring pixels from the same row (or column) can be directly connected, while there is no edge (one-cube) between the diagonal elements.

From the overview of persistent homology from 3.1, we need to specify three elements to be able to compute persistent homology from images: the group of chains and the boundary operator for a cubical complex, as well as the filtration.

The groups of chains are the algebraic counterparts of the geometric cubes, so that, a 1-cube has its associated 1-chain. We ‘add’ two cubes by taking their union,

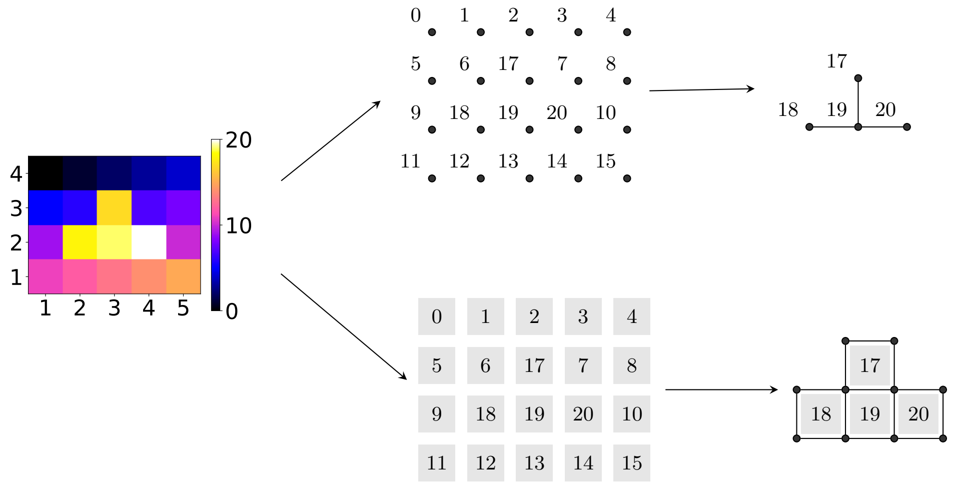


Fig. 4: On the left, a grayscale image. In the vertex construction, at the top, each pixel from the image is a 0-cube, while in the top-cell construction, at the bottom, it is a 2-cube. The top-right figure illustrates the super-level set K^{17} - the 1-and 2-cubes were added to the 0-cubes on the figure in the centre and assigned filtration values according to the upper-star filtration construction. An analogous process is shown on the bottom figure, where 0- and 1-cubes were added.

and the group of chains is spanned by the linear combinations of cubes. The boundary operator acts on cubes and returns the geometric boundary, decomposed on the set of cubes of lower dimension. For example, the boundary of a 1-cube is a collection of two 0-cubes - its endpoints. The detailed definitions are available in introductory material [31, 13].

Regarding the filtration, we first recall that the values of pixels in the image give us an $\mathbb{R}$-valued function on the vertices, which we will call a filter function. There are two ways natural ways to extend it to the complex and which lead to filtrations.

Definition 3.2. *Let $f : V(K) \to \mathbb{R}$ be a function defined on the vertices $V(K)$ of a cubical complex K. The lower-star filtration associated to f is $K_f = (K_s)_{s\in\mathbb{R}}$, where*

$$K_s = \{Q \subset K \mid f(v) \leq s, \forall v \in V(Q)\}. \tag{3.2}$$

We work with the upper-star filtration $(K^s)_{s\in\mathbb{R}}$, which is defined analogously to Definition 3.2, but reversing the order in (3.2),

$$K^s = \{Q \subset K \mid f(v) \geq s, \forall v \in V(Q)\}. \tag{3.3}$$

Strictly speaking, $(K^s)_{s\in\mathbb{R}}$ is not a filtration. The order of inclusions is now reversed $K_s \subset K_{s'}$, for all $s \geq s'$, but thanks to the monotonicity of the sequence, we can still compute the persistent homology, with the morphisms $i_s^{s'}$ in the other direction. Therefore, we will still call it a filtration.

In practice, computing persistent homology on an upper-star filtration from a function f is done by computing that of the lower-star filtration on $\tilde{f} : s \mapsto -f(s)$,

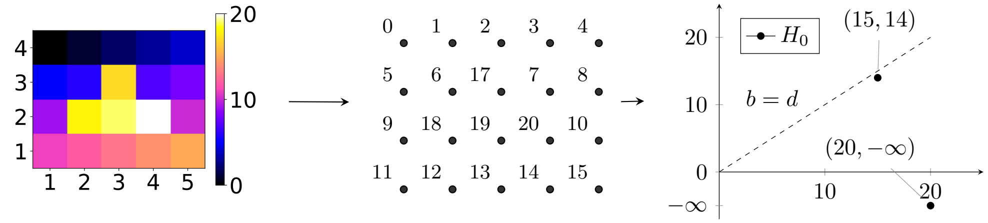


Fig. 5: In the middle, the vertex construction from the image on the left. The super-level set filtration of the underlying cubical complex yields non-trivial persistent homology classes only in dimension zero they are shown on the persistence diagram on the right.

which induces the same ordering of vertices as the upper-star filtration would have. To retrieve the original scale and ordering, we map the points from the persistence diagram $(b, d) \mapsto (-b, -d)$. This leads to the unusual situation, where $-b \geq -d$ and points appear below the diagonal in the birth-death plane.

3.3. Fingerprinting of spectral features. In section 3, we summarized the fingerprinting process for an image. We can now provide more details regarding the underlying complexes, built on windows from the spectrogram, the filter functions and the Betti curve representations. We divide the spectrogram into one-second, overlapping windows, of size $N_f \times N_T = 128 \times 170$. With the overlap between successive windows set to 0.4, a 30-second snippet results in 51 windows that start at 0., 0.6, 1.2, ... For each window W, we define a cubical complex K, with vertices in a $N_f \times N_T$ lattice. We convert each window to the decibel scale,

$$W_{i,j} \mapsto \frac{\log_{10}(W_{i,j}) - \log_{10}(\min(W))}{\log_{10}(\max(W)) - \log_{10}(\min(W))}, \tag{3.4}$$

and normalize (3.4) it, before interpreting it as a function $f_W : V(K) \to \mathbb{R}$, which we extend to a filtration of the full-complex K via an upper-star filtration. We compute the persistent homology groups on this filtered complex. Finally, we represent the persistent diagrams as Betti curves [26]. For the k-th diagram D_k, the k-th Betti curve (3.5) represents the evolution of Betti numbers throughout the filtration, where the Betti number associated to a homology group is its rank.

$$\begin{array}{rccc} \beta_k : & \mathbb{R} & \to & \mathbb{N} \\ & x & \mapsto & \sum\limits_{(d,b)\in D_k} 1_{]d,b]}(x), \end{array} \tag{3.5}$$

where $1_{]d,b]}(x) = 1$ if and only if $x \in]d, b]$ and 0 otherwise.

3.4. Comparing tracks. We compare tracks by comparing their fingerprints and state whether the two tracks match or not. Similarly to [32], we use the fingerprints of s, s', and their time-stamps, trying to find an alignment of the former in the latter. However, given that fingerprints section 3.3 are not hash values, we do not look for exact matches, but ‘similar enough” fingerprints. We introduce the notation for comparing sets of fingerprints, and present the algorithm. Consider two tracks s and s' and divide their spectrograms into collections of overlapping windows

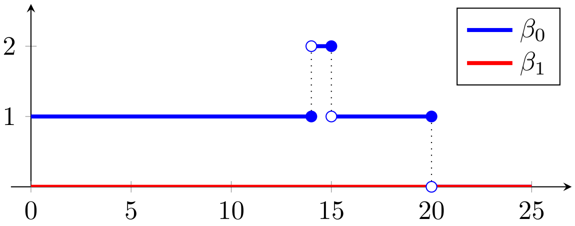


Fig. 6: The Betti curves β_0 and β_1 for the persistent diagrams of dimension zero and one from Figure 5.

$\{W_i\}_{i=1}^{N_s}$, $\{W'_j\}_{j=1}^{N'_s}$ as described above. For each window, W_i, we compute its Betti curves $\beta_{i,0}$, $\beta_{i,1}$ for homology dimensions zero and one and we proceed similarly for W'. Given two windows W_i and W_j, we compare them via the distance between their Betti curves. We compare every window W_i to all windows $\{W'_j\}_j$. Repeating so for all windows $\{W_i\}_i$, we obtain distance matrices M_0, $M_1 \in \mathbb{R}^{N_s \times N_{s'}}$ defined in (3.6)

$$(M_k)_{i,j} = \|\beta_{i,k} - \beta'_{j,k}\|_{L^1}. \tag{3.6}$$

Now, we have synthesized all the information from two tracks in matrices M_0, M_1, and the midpoint locations of windows t_k, t'_k. We define C as a linear combination of matrices M_0, M_1,

$$C = \lambda M_0 + (1-\lambda)M_1 \quad \in \ \mathbb{R}^{N_s \times N_{s'}}, \tag{3.7}$$

with $\lambda \in [0, 1]$. Then, $C_{i,j}$ represents how distant the window W_i centred at t_i in s, is from the t'_j-centred window W'_j in s'. We find a minimal-cost matching in C using an implementation of the Hungarian algorithm [30, 14]. The solution is a binary matrix X of size $N_s \times N_{s'}$, where $X_{i,j} = 1$ if the window centred at t_i is paired with the window centred at t'_j. This gives us a set of points $P = \{(t_i, t'_j - t_i) \mid X_{i,j} = 1\}$.We do a linear regression on P, obtaining a function $L(t)$ and assess the quality of the fit with the median of the error (3.8),

$$\Delta_{L,P} = \text{median}_{(t_i,t'_j)\in P} \left| L(t_i) - (t'_j - t_i) \right|. \tag{3.8}$$

We use a logistic regression model on $\Delta_{L,P}$ to determine the whether the tracks s, s' match.

The robustness of the method stems from two factors. First, if s was in fact s' modified with an obfuscation, we expect to find that the matching windows are aligned in time, what corresponds to points from P forming a line. With (3.8), the error is zero, if at least half of the pairs match well. Second, when we change the assignment of two windows ($X_{i_1,j_1} = 1$, $X_{i_2,j_2} = 1$ becomes $X_{i_1,j_2} = 1$, $X_{i_2,j_1} = 1$, for some row indices i_1, i_2 and column indices j_1, j_2), the linear model is altered, but the median error should remain small.

4. Experiments and results.

The fingerprinting and comparison methods described in sections 3.3 and 3.4 are tested on the duplicates problem: for a pair of tracks s, s', it consists of deciding whether one has been modified, through obfuscations, to yield the other. For example, if s' is s, to which a low-pass filter has been applied, they constitute a positive pair and we say that s is the parent of s'.

Type	Degree
Low-pass filter	1000, 1500, 2000, 3000,
High-pass filter	1500, 2000, 2500, 3000,
White-noise	0.05, 0.1, 0.2, 0.4,
Pink-noise	0.05, 0.1, 0.2, 0.4,
Reverb	40, 70, 100,
Pitch shift	0.8, 0.85, 0.9, 1.05, 1.15, 1.2,
Tempo shift	0.8, 0.85, 0.9, 1.05, 1.15, 1.2.

Table 1: Obfuscation types and degrees that were used to generate the set of obfuscated tracks. In low- and high-pass filters, the degree is the threshold frequency, so the higher the threshold, the smaller (greater) the obfuscation respectively. For white-noise, pink-noise and reverb, the smaller the degree, the closer the obfuscated track is to the original. Finally, for tempo and pitch shifts, a degree of 1 is the identity, while a displacement in either of the two directions increases the obfuscation.

We generate a dataset of 3466 positive and negative pairs of tracks. First, we sample $35,393$ tracks from the Million Song Dataset [3] and we download the corresponding 30-second preview snippets using the Spotify Web-API. To generate a positive (matching) pair, we choose a track, an obfuscation type and an obfuscation degree. We apply the obfuscation to that track, thus creating a new audio signal, and pair it with the original track. A negative pair consists of two different, possibly obfuscated, tracks.

We consider seven different types of obfuscations, each with three to six degrees of intensity. The degree determines to what extent the track is distorted- for example, a tempo shift with a factor of 1.05 indicates that the track has been sped up by a factor of 0.05, without changing its pitch. Obfuscations, summarized in Table 1, are generated using a Python wrapper of SOX [4]. We show an example of a raw fragment and its obfuscated version in Figure 7. While the addition of noise has no visible effect on the whole spectrogram, we can see its influence when examining the small windows. The corresponding Betti curves and the distances between them are shown below. The distances between the pairs of windows from the two tracks lead to the cost matrix.

We conducted the experiment on a subset of pairs. We have sampled 3466 (out of the 62000) positive and negative pairs at uniformly at random, what resulted in 50.2% positive pairs. We use the fingerprinting method from section 3.3 and the comparison algorithm proposed in section 3.4. The cross-validation results shown in Figure 8 led us to setting $\lambda = 0.33$, with which we obtain 84.7% recall and 99.0% precision.

The ROC is shown in Figure 9 and the AUC is 0.9374. On the same set, using the reference method and the counts [32] as a signal, we obtain get an AUC of 0.9214. However, in that case, the thresholds are integers, what may bias the AUC.

Our method is challenged by the high-pass obfuscations, which would indicate that most of the information is encoded in the lower regions of the spectrogram. We believe that despite the attenuation of the low frequency spectrum in the mel-spectrogram compared to the spectrogram, the maximum of the filter function is still attained in that region. Hence, as a high-pass filter attenuates this maximum, it also impacts the distribution of values of the filter function, what is reflected by big differences in the Betti curves. The benchmark method shows poor performance on

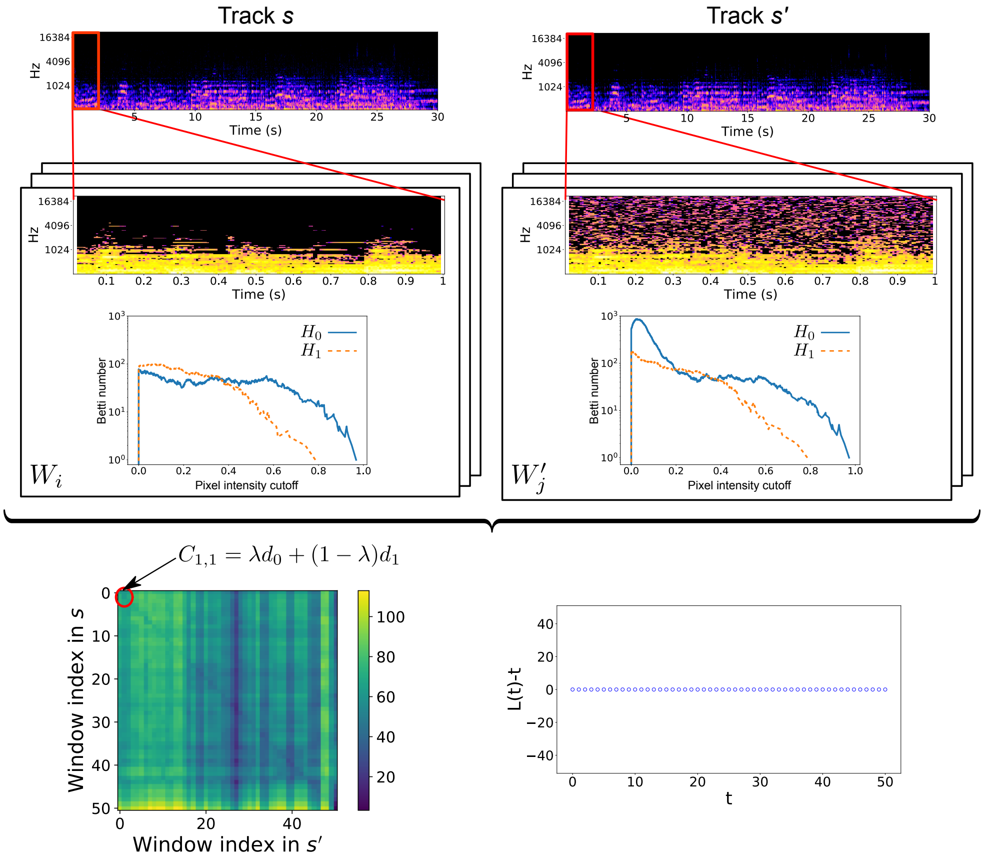


Fig. 7: The fingerprinting and comparison algorithm applied to a track [15] and its obfuscated version. From the two normalized mel-spectrograms, we create the cost matrix based on topological fingerprints. In particular, for each pair of windows, we compare the Betti curves and record the norm of their difference. From that distance matrix, we compute a minimal matching and build the collection of points shown at the bottom.

time-stretched tracks, due to the alignment algorithm. It is different from our as it takes only the mode of the offsets of matching hashes, rather than calculating a possible stretch factor as we do (3.8).

5. Discussion. We proposed a new audio fingerprinting method based on the persistent homology of spectral representations of tracks. We test the fingerprints, and a tailored identification algorithm, on a set of obfuscated pairs of tracks. Our results indicate that, compared to a standard 1–vs–N method, our fingerprints present more invariance to time stretching, but are more affected by filters which attenuate the low-frequency region. While the fingerprints are more expensive to compute and compare, making immediate generalizations to a 1–vs–N method impractical, the results show that the invariance captured by persistent homology is relevant for audio identification.

Further research should address two issues: poor performance on high-pass filters

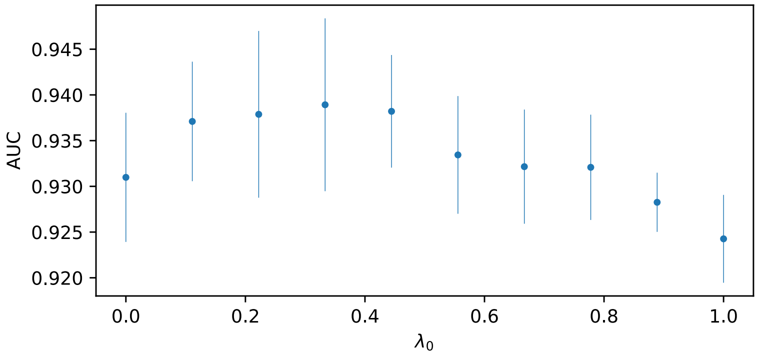


Fig. 8: Cross validation with 4 folds. Highest AUC values are reached for $0.1 \geq \lambda \leq 0.45$. While the maximum is attained at $\lambda = 0.33$, any value in that range could be acceptable due to the large the standard deviation of the metric between folds. Nevertheless, we notice that the information carried by β_1 proves relevant.

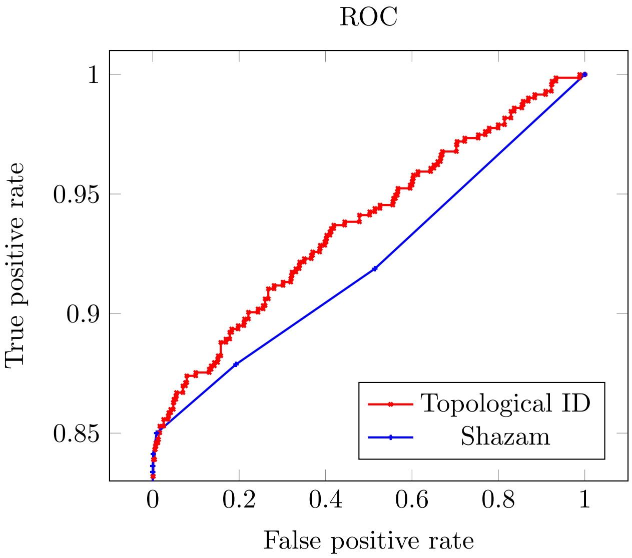


Fig. 9: The receiver operating characteristic in red, for binary classification, using the error defined in (3.8). We can see that the false positive rate is below 1%, even for recall around 87%. In blue, the analogue curve for the Shazam method. The area under the ROC is 0.9391 and 0.9214 for our and the Shazam method respectively.

and potential generalization to a 1–vs–N method. In an attempt to address the first, we have tried to section the spectral windows not only in time, but also in frequency (comparing windows from within the same frequency only). While this fine-grained comparison allows to 'filter-out' whole spectral regions which may have been altered, the results on the whole dataset did not improve what indicates that another matching

Obfuscation type	Degree	Recall	Recall Shazam
highpass	1500.00	0.040	**1.000**
	2000.00	0.000	**1.000**
	2500.00	0.045	**1.000**
	3000.00	0.000	**1.000**
lowpass	1000.00	0.952	**1.000**
	1500.00	1.000	1.000
	2000.00	1.000	1.000
	3000.00	1.000	1.000
pinknoise	0.05	1.000	1.000
	0.10	1.000	1.000
	0.20	1.000	1.000
	0.40	0.950	**1.000**
pitch	0.80	1.000	1.000
	0.85	1.000	1.000
	0.90	1.000	1.000
	1.05	1.000	1.000
	1.15	1.000	1.000
	1.20	1.000	1.000
reverb	40.00	1.000	1.000
	70.00	1.000	1.000
	100.00	0.909	**1.000**
tempo	0.80	**1.000**	0.000
	0.85	**1.000**	0.000
	0.90	**1.000**	0.000
	1.05	**1.000**	0.276
	1.15	**1.000**	0.067
	1.20	**1.000**	0.000
whitenoise	0.05	1.000	1.000
	0.10	1.000	1.000
	0.20	1.000	1.000
	0.40	0.727	**1.000**

Table 2: Comparison of recall for the proposed method against the benchmark [32]. The proposed method shows perfect robustness to the applied tempo obfuscations. On the other hand, it is affected the most by highpass filters. For higher degrees of obfuscation, there a few non-identified matches for white- and pink-noise, as well as reverb.

algorithm might be necessary. A 1–vs–N system can be proposed provided that we have a method to search for nearest neighbors in the space of features derived from persistence diagrams. A possible solution consists of replacing Betti curves with hash functions for persistence diagrams. While a family of hash functions has been proposed for persistence diagrams [8], its suitability for this particular application needs to be assessed.

Acknowledgments. We would like to thank Prof Kathryn Hess for facilitating the student internship with HAH and thank the Mathematical Institute at Oxford for hosting WR. We thank researchers from Spotify Dr. Martin Gould, Dr. Johnny

Hunter, Laurence Pascall, Dr. Brian Brost, Shahar Elisha for their help and continuous support. HAH acknowledges funding from a Royal Society University Research Fellowship and EPSRC EP/R018472/1 supporting the Centre for Topological Data Analysis.

REFERENCES

[1] S. Baluja and M. Covell, *Audio Fingerprinting: Combining Computer Vision & Data Stream Processing*, in 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07, Honolulu, HI, USA, 2007, IEEE, pp. II–213–II–216, https://doi.org/10.1109/ICASSP.2007.366210, http://ieeexplore.ieee.org/document/4217383/ (accessed 2019-05-14).

[2] P. Bendich, E. Gasparovic, J. Harer, and C. J. Tralie, *Scaffoldings and Spines: Organizing High-Dimensional Data Using Cover Trees, Local Principal Component Analysis, and Persistent Homology*, Springer International Publishing, Cham, 2018, pp. 93–114, https://doi.org/10.1007/978-3-319-89593-2_6, https://doi.org/10.1007/978-3-319-89593-2_6.

[3] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, *The million song dataset*, in Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011), 2011.

[4] R. M. Bittner, E. Humphrey, and J. P. Bello, *PySOX: Leveraging the audio signal processing power of SOX in Python*, New York City, USA, Aug. 2016, p. 3.

[5] F. Chazal, V. de Silva, M. Glisse, and S. Oudot, *The structure and stability of persistence modules*, arXiv:1207.3674, (2012), http://arxiv.org/abs/1207.3674 (accessed 2019-05-18). arXiv: 1207.3674.

[6] H. Edelsbrunner and J. Harer, *Persistent homology—a survey*, in Contemporary Mathematics, J. E. Goodman, J. Pach, and R. Pollack, eds., vol. 453, American Mathematical Society, Providence, Rhode Island, 2008, pp. 257–282, https://doi.org/10.1090/conm/453/08802, http://www.ams.org/conm/453/ (accessed 2019-11-07).

[7] H. Edelsbrunner, D. Letscher, and A. Zomorodian, *Topological Persistence and Simplification*, (2002), p. 13.

[8] B. T. Fasy, X. He, Z. Liu, S. Micka, D. L. Millman, and B. Zhu, *Approximate Nearest Neighbors in the Space of Persistence Diagrams*, arXiv:1812.11257 [cs], (2018), http://arxiv.org/abs/1812.11257 (accessed 2019-11-12). arXiv: 1812.11257.

[9] A. Garin, T. Heiss, K. Maggs, B. Bleile, and V. Robins, *Duality in Persistent Homology of Images*, arXiv:2005.04597 [cs, math], (2020), http://arxiv.org/abs/2005.04597 (accessed 2020-07-27). arXiv: 2005.04597.

[10] S. Gholizadeh and W. Zadrozny, *A Short Survey of Topological Data Analysis in Time Series and Systems Analysis*, arXiv:1809.10745 [cs], (2018), http://arxiv.org/abs/1809.10745 (accessed 2019-03-03). arXiv: 1809.10745.

[11] Y. Hiraoka, T. Nakamura, A. Hirata, E. G. Escolar, K. Matsue, and Y. Nishiura, *Hierarchical structures of amorphous solids characterized by persistent homology*, Proceedings of the National Academy of Sciences, 113 (2016), pp. 7035–7040, https://doi.org/10.1073/pnas.1520877113, http://www.pnas.org/lookup/doi/10.1073/pnas.1520877113 (accessed 2018-09-16).

[12] Julius O. Smith, *Spectral Audio Signal Processing*, http://ccrma.stanford.edu/jos/sasp/, 2011. online book, 2011 edition.

[13] T. Kaczynski, K. M. Mischaikow, and M. Mrozek, *Computational homology*, Springer, New York; London, 2011. OCLC: 1063425526.

[14] H. W. Kuhn, *The Hungarian method for the assignment problem*, Naval Research Logistics Quarterly, 2 (1955), pp. 83–97, https://doi.org/10.1002/nav.3800020109, http://doi.wiley.com/10.1002/nav.3800020109 (accessed 2019-11-10).

[15] Le Loup, *Morning Song*, Sept. 2019, https://p.scdn.co/mp3-preview/49edd394d29827343c7bbdda08304745d0f6b6f1?cid=774b29d4f13844c495f206cafdad9c86https://p.scdn.co/mp3-preview/49edd394d29827343c7bbdda08304745d0f6b6f1?cid=774b29d4f13844c495f206cafdad9c86.

[16] C. Li, M. Ovsjanikov, and F. Chazal, *Persistence-Based Structural Recognition*, in 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, June 2014, IEEE, pp. 2003–2010, https://doi.org/10.1109/CVPR.2014.257, http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6909654 (accessed 2019-03-08).

[17] J.-Y. Liu, S.-K. Jeng, and Y.-H. Yang, *Applying Topological Persistence in Convolutional Neural Network for Music Audio Signals*, arXiv:1608.07373 [cs], (2016), http://arxiv.org/

abs/1608.07373 (accessed 2019-03-03). arXiv: 1608.07373.

[18] B. McFee, C. Raffel, D. Liang, D. Ellis, M. McVicar, E. Battenberg, and O. Nieto, *librosa: Audio and Music Signal Analysis in Python*, Austin, Texas, 2015, pp. 18–24, https://doi.org/10.25080/Majora-7b98e3ed-003, https://conference.scipy.org/proceedings/scipy2015/brian_mcfee.html (accessed 2019-03-03).

[19] M. Mohri, P. Moreno, and E. Weinstein, *Robust Music Identification, Detection, and Analysis*, in Proceedings of the International Conference on Music Information Retrieval, Vienna, Austria, Sept. 2007, pp. 135–138.

[20] J. R. Munkres, *Elements of Algebraic Topology*, Addison-Wesley Publishing Company, Cambridge, Mass, 1984, http://people.dm.unipi.it/benedett/MUNKRES-ETA.pdf (accessed 2020-04-10).

[21] N. Otter, M. A. Porter, U. Tillmann, P. Grindrod, and H. A. Harrington, *A roadmap for the computation of persistent homology*, EPJ Data Science, 6 (2017), https://doi.org/10.1140/epjds/s13688-017-0109-5, http://epjdatascience.springeropen.com/articles/10.1140/epjds/s13688-017-0109-5 (accessed 2019-02-14).

[22] J. Reininghaus, S. Huber, U. Bauer, and R. Kwitt, *A Stable Multi-Scale Kernel for Topological Machine Learning*, arXiv:1412.6821 [cs, math, stat], (2014), http://arxiv.org/abs/1412.6821 (accessed 2019-02-20). arXiv: 1412.6821.

[23] V. Robins, M. Saadatfar, O. Delgado-Friedrichs, and A. P. Sheppard, *Percolating length scales from topological persistence analysis of micro-CT images of porous materials: PERCOLATION FROM PERSISTENCE*, Water Resources Research, 52 (2016), pp. 315–329, https://doi.org/10.1002/2015WR017937, http://doi.wiley.com/10.1002/2015WR017937 (accessed 2019-07-13).

[24] N. Sanderson, E. Shugerman, S. Molnar, J. D. Meiss, and E. Bradley, *Computational Topology Techniques for Characterizing Time-Series Data*, arXiv:1708.09359 [cs], (2017), http://arxiv.org/abs/1708.09359 (accessed 2019-04-09). arXiv: 1708.09359.

[25] M. Sarfati, A. Hu, and J. Donier, *Ensemble-based cover song detection*, arXiv:1905.11700, (2019), http://arxiv.org/abs/1905.11700 (accessed 2019-12-08). arXiv: 1905.11700.

[26] A. Sizemore, C. Giusti, and D. Bassett, *Classification of weighted networks through mesoscale homological features*, arXiv:1512.06457, (2015), http://arxiv.org/abs/1512.06457 (accessed 2019-06-16). arXiv: 1512.06457.

[27] S. S. Stevens, J. Volkmann, and E. B. Newman, *A Scale for the Measurement of the Psychological Magnitude Pitch*, The Journal of the Acoustical Society of America, (1937), p. 7, https://doi.org/10.1121/1.1915893.

[28] C. J. Tralie, *Early MFCC And HPCP Fusion for Robust Cover Song Identification*, CoRR, abs/1707.04680 (2017), p. 11, http://arxiv.org/abs/1707.04680.

[29] C. J. Tralie and P. Bendich, *Cover Song Identification with Timbral Shape Sequences*, (2015), p. 12.

[30] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors, *SciPy 1.0–Fundamental Algorithms for Scientific Computing in Python*, arXiv e-prints, (2019), arXiv:1907.10121, p. arXiv:1907.10121, https://arxiv.org/abs/1907.10121.

[31] H. Wagner, C. Chen, and E. Vuçini, *Efficient Computation of Persistent Homology for Cubical Data*, in Topological Methods in Data Analysis and Visualization II, R. Peikert, H. Hauser, H. Carr, and R. Fuchs, eds., Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 91–106, https://doi.org/10.1007/978-3-642-23175-9_7, http://link.springer.com/10.1007/978-3-642-23175-9_7 (accessed 2019-01-28).

[32] A. L.-C. Wang, *An Industrial-Strength Audio Search Algorithm*, in Proceedings of the 4 th International Conference on Music Information Retrieval, 2003, pp. 713–718.

[33] Yan Ke, D. Hoiem, and R. Sukthankar, *Computer Vision for Music Identification*, in 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, San Diego, CA, USA, 2005, IEEE, pp. 597–604, https://doi.org/10.1109/CVPR.2005.105, http://ieeexplore.ieee.org/document/1467322/ (accessed 2019-05-15).

[34] M. Zeppelzauer, B. Zieliński, M. Juda, and M. Seidl, *Topological descriptors for 3d surface analysis*, arXiv:1601.06057, (2016), http://arxiv.org/abs/1601.06057 (accessed 2019-12-14). arXiv: 1601.06057.

[35] A. Zomorodian, *Computing Persistent Homology*, Discrete & Computational Geometry, 33

(2005), p. 15, https://doi.org/https://doi.org/10.1007/s00454-004-1146-y.