TOPOLOGY-PRESERVING DEFORMATIONS OF DIGITAL AUDIO

Georg Essl *

Department of Mathematical Sciences
University of Wisconsin - Milwaukee
Milwaukee, U.S.A.
essl@uwm.edu

ABSTRACT

Topology provides global invariants for data as well as spaces of deformation. In this paper we discuss the deformations of audio signals which preserve topological information specified by sublevel set persistent homology. It is well known that the topological information only changes at extrema. We introduce *box snakes* as a data structure that captures permissible editing and deformation of signals and preserves the extremal properties of the signal while allowing for monotone deformations between them. The resulting algorithm works on any ordered discrete data hence can be applied to time and frequency domain finite length audio signals.

1. INTRODUCTION

Sound synthesis and manipulation methods operate on a space of variability of sound under certain chosen properties. Recently, topological methods have emerged as one way to structure sound synthesis methods, either by giving more flexibility to existing synthesis methods [1] or by providing direct deformation strategies of oscillators [2]. Topology describes properties that are global or flexible in nature. Data and signals can often be deformed under certain rules that do not alter the underlying topology of the space. Hence topological properties serve as a kind of invariant of the signal under deformations. Previous methods suggest a geometric setting that is higher dimensional, such as oscillators embedded in a plane, or winding paths on the surface of a torus in three dimensions.

This paper explores the analysis and manipulation performed directly on finite length digital audio data, both in the time and frequency domain. Hence we no longer need to leave the sampled 1-dimensional setting that is typical of audio data. This setting is both convenient and familiar to practitioners in digital signal processing.

Specifically, we develop the notion of *box snakes* as a structure that describes the space of deformability of audio signals that does not alter the underlying topology as characterized by sublevel set persistent homology [3]. A characterization coined *topological signature* has been developed in a related but somewhat different spirit and context [4]. We also extend existing algorithms to work with all possible sample sequences without requiring exceptions or other technical assumptions on the data. The algorithm will allow for monotonic deformation of audio samples between local extrema. Given that the algorithm works for any finite ordered discrete series data, we demonstrate its use both on time domain as well as on frequency domain data, hence also give topology-preserving deformations of spectra.

2. RELATED WORK

In the last two decades fields such as Computational Topology [5, 6], Applied Topology [7], Topological Signal Processing [8] have emerged as vibrant fields of inquiry across many disciplines. A range of topological techniques have already been proposed for audio and more generally time series data [9]. Sliding window persistent homology utilizes the embedding of time series in higher dimensions and then computes point cloud persistent homology on the data [10]. This approach has already demonstrated to be useful in musical instrument detection on the example of the differentiation of a clarinet and a viola [11]. Audio data has also been modeled using sheaf-theoretic approaches [1]. Finally, synthesis of audio data can also be achieved by generalizing circular oscillators to more general path-connected spaces [2].

In this paper we will work with sublevel set persistence [3, 12, 13], which has been used to tackle numerous problems in time-series signals such as zero-crossing detection [14], damping parameter estimation [15], and additive noise analysis [16]. The sublevel set persistence algorithm used in these cases [16] is available in Python as part of the Teaspoon topological data and signal analysis library [17]. Sublevel set persistence is also the basis of more sophisticated graph theoretic characterization of functions [18, 12, 19, 20, 21].

This work introduces a data structure called box snakes to specify a space of deformation for resynthesis and manipulation and an algorithm to compute it. The idea is that levelset persistence provides an invariant due to the pattern of extrema which form an alternating snake-like pattern (snakes was studied as mathematical objects by Arnold [22]), and any data that yields the same topological characterization is topologically equivalent. This yields a resynthesis technique in the time domain. Some of these ideas resemble Extended Waveform Segment Synthesis as developed in a thesis by Valsamakis [23, p. 120-121]. This work suggested sound segmentation along numerous identifiable local markers such as amplitude levels, zero crossings, as well as minima, and maxima. It, however, does not require any invariant properties, such as the preservation of topological information) from the selection of local markers. The thesis also contains a review of an array of proposed interpolation techniques, a monotone version of which applies to the approach proposed here. The technique presented in our work can be viewed as a topological version of an analysis-synthesis technique, where the analysis is topological and the sound synthesis is the space of allowable deformations. It is also relatable to

* This work was made possible in part by a fellowship of the John Simon Guggenheim Foundation

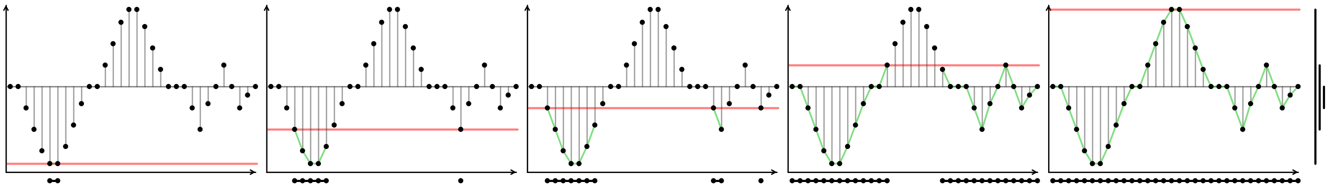


Figure 1: Sublevel Set Persistent Homology of a finite Audio signal. As we raise the level line, more of the signal is included. The connected component at each level (red) is depicted below the time series graph. At extrema, connected components are created or merged.

graphical synthesis methods that draw waveforms, and specifically interpolation synthesis techniques [24, for a review].

The algorithm technically operates on any sequence of discrete data points, hence it can be naturally applied to discrete finite-length data in the frequency domain, such as for example the Amplitude or real spectrum of a discrete Fourier Transform (DFT). Previous work includes the study of weighted Fourier series [25] and Walsh-Fourier Transforms [26] in the context of sublevel set persistence. The former is based on the assumption of continuous Morse functions and the latter uses a transform not particularly well-suited for Audio analysis. In our work we do not assume continuous functions, merely the discrete time series data. We compute discrete Fourier transforms and compute the level set persistence on the real spectrum.

In order to compute the sublevel set persistence on finite length discrete data without any restriction on the data, we modify an existing algorithm by Baryshnikov [12]. The original algorithm has some technical restrictions. It requires that all extrema be generic and isolated. The genericity (also called uniqueness) criteria means that no two extrema can be on the same value. This restriction is unnatural for audio data. Basic signals such as silence (all zeroes) or any undamped sinusoid have extrema of the same value as periods repeat [27, Appendix A]. The second restriction of isolated extrema is that the algorithm does not allow for *flat* extrema, meaning extrema which have the same value for consecutive samples. This too is not unusual in especially artificial audio samples, in fact any constant function in this context can be viewed as a minimum (and maximum) of all equal values, hence again does not fit the restriction. Rectangular waves, even if antialiased, can have flat portions, and certainly arbitrary audio can have passages that contain repeating values at extrema on occasion.

The modified algorithm gives equivalent topological information to Baryshnnikov's in all comparable cases, hence removes the restrictions in an appropriately principled way. This algorithm computes similar data as one given by Myers *et al* [16] but is different in detail. the Python library `scipy.signal` provides the `find_peaks` function which will return flat extrema, and this function is used to implementation of the algorithm described in [16]. Other algorithms exist [28] though often with similar restrictions as Baryshnikov's remain [29][1].

[1]The given restrictions are primarily motivated by classical Morse theory of differentiable functions, as well as being able to study binary merge trees rather than more general ones, both notions we do not have space to explain here. The interested reader can find more detail in [12] and references therein.

3. TOPOLOGY OF DISCRETE SEQUENCE OF SAMPLES

A sequence of audio samples is just an array of numbers. On its face a discrete set of the sequence has a discrete topology. However, the sequence of numbers also has a sequential order or successor relationship. We can identify which samples are neighbors of others. There are various ways to represent this additional structure. For example, one could connect neighboring samples by "graph lines" that are meant to capture this relationship. Another approach is to simply recognize that samples are connected by successor/predecessor relationships. Algorithmically, this is captured by increment of array indices and the recognition that the order of array entries matters. In this paper we will use all these concepts to illustrate how topological notions operate on discrete samples in our given context. We call a sample a *boundary* if it has only one neighbor and we call a sequence of samples *periodic* if the successor of the last sample is the first sample in the sequence (and the predecessor of the first sample is the last one). Equivalently this means that indices computed modulo the length of the sequence preserved this successor/predecessor relationship. This means that sequences with boundaries in our case are not connected between endpoints whereas a periodic sequence connects the endpoints of the array. If samples in a subsequence of the original sequence are neighbors of each other we say that they are *connected*. A sequence of all consecutively connected neighbors is called a *connected component*. An isolated point is its own connected component. Throughout this paper we will assume that all our sample data is bounded, which is a natural assumption for audio data.

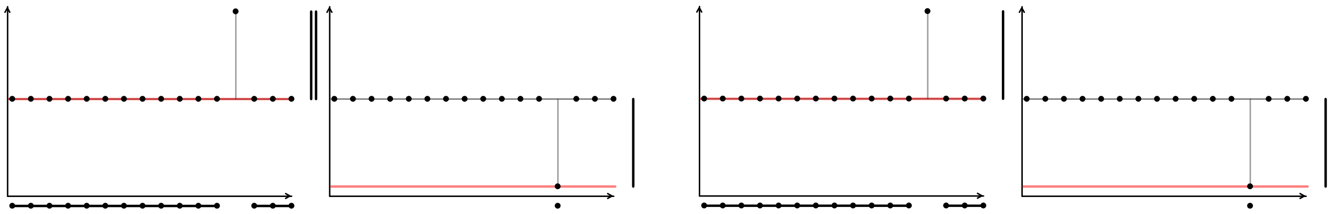
4. SUBLEVEL SET PERSISTENCE HOMOLOGY

Sublevel set persistent homology computes the changes in connectivity as a level set moves across a function[2]. The idea is depicted in Figure 1. The figure shows a red horizontal line. This line marks a level. A sublevel set is a subset of the original sequences of samples such that a sample is included if and only if the its value is at or below the level set.

We visualize a current sublevel set by a graph-like horizontal structure below the sample plot as copies of the included sample without amplitude, but with their neighborhood relationships depicted by connecting lines.

Think of moving the level that selects the sublevel set from below the global minimum to above the global maximum. Steps of this process are illustrated in Figure 1. Each subfigure illustrates what happens when a local minimum or maximum is reached. Notice that if the level is set below the global minimum, nothing is

[2]Due to the limited space, the exposition here is necessarily somewhat compressed. For a more leisurely introduction of topological notions for a DAFx readership the author has prepared expository material that is more expansive on many of the notions used here [30, 31].



(a) Interval with Boundaries

(b) Circular Domain

Figure 2: Barcodes of positive and negative impulses on (a) an interval with boundaries and on (b) a circular domain. Sublevel line (red) and connected component (below graph) are shown at the minima. The symmetry for the circular domain is broken for the bounded interval.

included. But as we hit local minima, samples become included in the sublevel set. If the minimum is flat, all points at that level are included (leftmost subfigure). If the minimum is a local point then just one sample is included (second subfigure). Graphically we depict that by showing a connecting line segment, but the reader should be cautious not to think of that as a geometric line, but rather like a line in an abstract graph representing connectivity.

As we move upward with the red level line, notice that connected components grow with the monotonically increasing samples until a local (potentially flat) maximum is reached (the rightmost two subfigures). At the local maxima connected components from the left and right of the maximum fuse and form one bigger connected component. The key observation is that connected components are created at minima, and that connected components are joined at maxima. A connected component is a topological entity that is counted in the 0th Homology. The number counting them is called 0th-Betti number[3] or Betti-0 for short. For our purposes it is sufficient to understand that we are counting connected components and their change (creation, joining). The stretches of raising the level that does not cause any change number of connected components give rise to the word *persistence* of a connected component and motivates the name *persistent homology*. Equivalently these changes also reflect monotone sequences between extrema. A sequence is *monotone* if all consecutive samples are partially ordered, that is $x[n] \leq x[n+1]$ for all samples in a monotonically increasing subsequence and $x[n] \geq x[x+1]$ a monotonically decreasing subsequence. Note that the condition of monotonicity implies that there are no extrema in the subsequence.

One can keep track when a connected component is created at a minimum and when it is joined with another is via a *barcode* depiction (see the vertical bars in the far right of Figure 1)[4]. The bottom is the starting level of the connected component existing and the top is the moment when the connected component is joined with another. As one moves the level set higher and higher more and more of the signal is included, local minima can spawn new connected components while local maxima will join some, hence reducing the number that exist. But ultimately once one passes the global maximum of the audio signal, the whole signal will be included hence forming one single connected component. Philosophically that connected component lives on even as we move into a theoretical infinity. Hence there is always one barcode left that never terminates. This is the topology of the audio signal as a whole. In our depictions of barcodes we will not honor this theoretical notion and show the end barcodes at the global maximum. Barcodes are, however, just one way to capture the changes in topology at minima and maxima. Another popular way to organize this data are *merge trees*, which simply record creation at minima as leafs and joins of connected components as interior joints of a tree structure. It is known that our setting leads to multiple branches associated with an internal node [18], but literature treating this case explicitly is scant. We are not considering merge trees in this paper.

Finally, recognizing that minima and maxima necessarily need to alternate on a sequential discrete data, one can also depict the information as a *snake* [22, 18, 12]. We will use barcodes to depict a given levelset persistence, and we will use an extended notion of snakes as the primary data structure to allow for topology-preserving deformations of our sequential discrete data.

[3] See [30] for more detailed exposition of Homology.

[4] All our barcodes are of the 0th Homology hence represent connected components.

5. PERIODIC DOMAIN VERSUS INTERVAL WITH BOUNDARIES

Especially with respect to periodic signals and their discrete characterization via the Discrete Fourier transform, it is known that the array is periodic, meaning that the successor of the last entry in the array of audio samples is the first entry. An alternative, perhaps naively more natural assumption is that the array is just a sequence of data with a starting and an end point, assuming no periodicity. Topologically this makes the start and end points into *boundary points* which are different from other points in the sequence in that they have only one neighbor under a successor/predecessor relationship. The existence of boundaries distinguishes these two cases. In the following we explore some properties as a consequence of each choice with respect to sublevel set persistent homology, to support the notion that periodic assumption give more natural results for the purpose of topological characterization of digital audio.

First we consider a positive and a negative impulse somewhere in the sequence away from the array boundary as shown in Figure 2 (a). The left side of the figure shows the sublevel set persistent homology of the interval with boundaries. Notice that at zero level, two connected components are created, one for each side of the impulse. These then merge at the peak of the impulse. Hence the presence of boundaries means that we have two minima and one maximum. Hence we have two barcodes, one for each minimum. The right side of the figure shows the sublevel set persistent homology of a negative impulse on the interval with boundaries. Notice that now there is only one minimum hence only one connected component is created and at zero both sides join with it to create one connected component joining with two maxima. Hence we have only one barcode. Hence we find that the inverse of a signal does not necessarily have the same topological characterization as the original signal! This is awkward for audio where axial symmetry is a central property. Furthermore notice that the number of maxima and minima do not match, and the way they mismatch changes under axial symmetry.

Now we consider the same signal over a periodic domain as shown in Figure 2 (b). This means that the left of the sequence is considered to be connected with the right (and vice versa) though that is not explicitly depicted in the figure. For the positive impulse the zero level now forms one connected component (due to this connection!) and hence we get one minima at zero and one maxima at the peak. The same effect but with inverted roles of minima and maxima plays out for the negative impulse. We get one minimum from the negative peak, but now one connected maximum at zero. Hence now we get a single barcode representing both the positive and negative impulse. Hence the presence and handling of extrema at the boundary introduces an asymmetry that disappear in the periodic case.

The case of the boundary is further complicated if the impulse occurs at the boundary samples, because in that case the bar code changes again, and in fact only one is created for both the positive and the negative impulses. Hence the interval with boundaries is not consistently encoding barcodes for impulses over the whole length of the interval but gives different results whether the impulse is at the boundary or in the interior. The original algorithm proposed by Baryshnikov [12] fixes this by introducing an artificial global minimum next to one boundary and an artificial global maximum at the other. This has the effect of making all points in the original sequence interior points, hence removing the boundary effects, and additionally guarantees that the number of minima and maxima over the extended interval is equal. However, this fix has no inherent justification from the signal.

The periodic domain does not suffer this effect, nor does it require mitigating assumptions. The barcode of the impulse is invariant under any periodic shift (with the period being the length of the sequence) of the impulse on the domain. And furthermore notice that the number of minima and maxima of a non-constant sequence can be proven to always match. Assume that there is a discrepancy in the number. This implies that the signal would have to grow in one direction indefinitely, but the number of samples to represent the signal is finite, arriving at a contradiction.

From a topological perspective these effects make sense. Boundaries are central aspects of homological information. The presence and absence of boundaries changes the homology [30]. Hence for the purpose of signal characterization it is important to be clear what one assumes about the signal to get a well-fitted topological characterization. Periodicity assumption on sequences appear formally in techniques such as finite length discrete Fourier transforms, hence this assumption matches well the underlying structure of standard spectral analysis techniques in digital audio. At the same time the prevailing paradigm of understanding perception of audio is via assumptions of periodicity detection [32]. Hence, in this paper we use periodic domains for all examples outside this section.

Circular domains when both the time and frequency domain are sampled (and due to duality made circular in the other domain), as shown by Steiglitz [33] (see Figure 3). Hence our homological arguments for circular domains of sublevel set persistence computations match the known circular properties of the DFT of finite length. This is equivalent to treating the the indexing of the samples under modulo arithmetic [30].

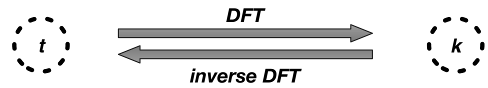


Figure 3: Duality of discrete finite length time and frequency data under the Discrete Fourier Transform [33].

6. COMPUTING SUBLEVEL SET PERSISTENCE

A sublevel set persistence algorithm takes as input a finite sequence of discrete data and returns a set of barcodes. Our starting point is an algorithm proposed by Baryshnikov [12]. Rather than provide a separate algorithm we will include our barcode construction within the box snake algorithm 3.

We assume that the discrete data set is periodic, thus if we have a discrete finite set of samples of length p and our index starts at 0 then we have that the successor of $x[p]$ is $x[0]$. This means that the our indexing of the sample arrays will happen in integer arithmetic $\texttt{mod}\ p$. The consequence for modifying the algorithm is largely straightforward. It means that one has to pick a point on the domain and check if one completed a cycle of length p. Various specific strategies for this adoption have appeared in the literature, such as separating the cycle and reconnecting it after computing the separated case [28]. In this paper we always compute in mod p directly. Our sublevel set persistence computation differs from other examples in the literature in that we make no assumptions of genericity or isolate extrema of our discrete data. This means that multiple barcodes can start at the same level and multiple barcodes (not necessarily the same set) can end at the same level as well. This means that there is no unique hierarchy or order of emergence of bar codes, and we will ignore the usual ordering by the elder rule. Our bar codes will guarantee to cover creation of a connected component at a local minimum, and the joining to at least one neighboring connected component at the nearby local maximum. This simplifies the handling of bar codes as we no longer try to reconstruct the elder prioritization. Furthermore we need to handle the potential of flat segments in the data. Flat segments are neighboring samples that are at the same level ($x[n] = x[n+1] = \cdots = x[n+l-1]$ for a flat of length l. The handling of flats is done in a straightforward way by using a function for advancing samples that advances over flat segments as shown in algorithm 2. The actual construction of the bar code happens when a local extremum is detected. Pairs of extrema need to be organized into a barcode, which is straightforward but reproduced for completeness in the function presented in algorithm 1.

7. BOX SNAKE OF DISCRETE SAMPLE SEQUENCES

Arnold [22] coined the term *snake* for a discrete pattern of numbers that alternate in total order: $x[0] < x[1] > x[2] < \cdots > x[n-1] < x[n]$. This structure reflects the fact that extrema of one dimensional sequences of numbers necessarily need to alternate between minima and maxima. Hence snakes are contained as subsets of any finite audio time series. Note that we take $x[0]$ here as an arbitrary reference point, and the sequence could continue with negative index or be labeled with an alternative indexing.

For our purposes we want to create a structure that captures the snake behavior as well as carries through audio samples between extrema. To this end we create a box snake structure. A

Algorithm 1 Construct Bar from Extremum

```
barcode ← empty barcode array
bars ← 0
function CONSTRUCTBAR(data,dir)
    ▷ data is data at extremum, and dir is −1 for minimum and 1 for maximum

    if barcode[bars] not created then
        barcode[bars] = new array[2]
    end if                              ▷ Convert dir from −1,1 to 0,1
    barcode[bars][(dir+1)/2]=data

    if barcode[bars] is filled then
        bars ← bars +1
    end if
end function
```

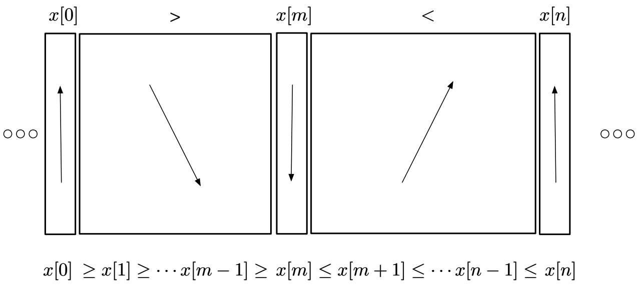


Figure 4: The box snake structure of alternating extrema with monotone sequences between them.

box snake is a snake with allows for additional samples between extrema that are required to be monotone. $x[0] < x[1] \leq \ldots \leq x[n-1] < x[n] > x[n+1] \geq \ldots \geq x[m-1] > x[m]$. While the snake structure is necessary (unless there function is constant), the monotone stretches are optional and can be empty. This is the case when a local extrema is followed immediately by another (minimum following a maximum or vice versa). The structure of a box snake is depicted in Figure 4.

The key connection between sublevel set persistence and box snakes is that only extrema alter homological information (the 0th Betti number counting the number of connected components). Hence, the monotonic sequences between extrema do not impact this information. Any arbitrary monotonic sequence is permissible and will not change the sublevel set persistent homology. Hence any deformation of these sequences that preserves monotonicity is allowable. This insight forms the key ingredient in the (re)synthesis method of this paper.

Figure 5 shows only the monotone boxes computed by the box snake algorithm 3. The vertical boundaries of the boxes are given by the levels of adjacent extrema. The samples included are all the samples not part of extrema, which can be potentially flat. This latter choice is somewhat arbitrary. Flat sequences of samples are not essential to maintaining a barcode. Strictly only one sample of the flat needs to stay at its level to maintain a barcode without change. Hence one can in principle move samples from flat extrema into monotone adjacent boxes. The inverse also holds as we will see in Figure 6.

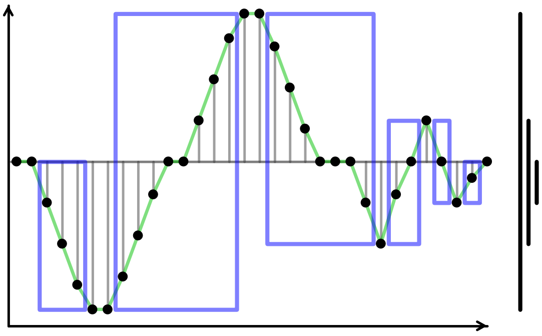


Figure 5: Monotone boxes (blue) computed from the box snake algorithm. These are the rectangular areas where monotone deformations of the samples will not alter the bar codes on the right. The green connecting lines are for visualization purpose only and are not meant to imply linear functional values between samples.

8. ALGORITHM TO COMPUTE BOX SNAKES

Algorithm 3 gives the Snake Boxes associated with a finite data sequence (usable for both time and frequency domain). It assumes that the array is periodic, that is, $succesor x[p-1] = x[0]$. Hence our index is computed modulo p whenever array entries are indexed. However, it can be convenient to have sequences successively increase, hence the algorithm also uses indices that are lifted into periodic repetitions of p (Topologists think of this the lift into the cover). Our algorithm has the following properties: (1) Handle local flat extrema as belonging to the extremum. (2) Allow for flats within monotone sequences.

Flat sequences are relatively straightforward to handle though there is scant discussion in current literature about how to do so. The function that handles this described in algorithm 2. The function takes a current index as input as well as the modulus (which is equivalently the length of the sequence) and returns the position to skip to in order to skip over a flat region. Using this function as sequence incrementer will handle flats and also allow us to keep track of them by storing information about pre-function indices and length skipped.

Algorithm 2 Skip number of successive flats from a start position

```
function SKIPFLATS(s,p)
        ▷ s is the start index, p is the total number of samples
    for i ← s, s + p do
        if data[i mod p] ≠ data[(i + 1) mod p] then
            return i + 1
        end if
    end for
    return s + p
end function
```

The full algorithm uses this function whenever it needs to advance the index through the sequence. The function is also useful to turn existing algorithms that compute sublevel set persistent homology on sequence data that do not support flats into ones that do. We have adopted Barychnikov's algorithms for sublevel set persistent homology to allow flats and used it to compute the bar codes in illustrations throughout the paper. The algorithm was further adopted to allow extrema at the same level (remove genericity

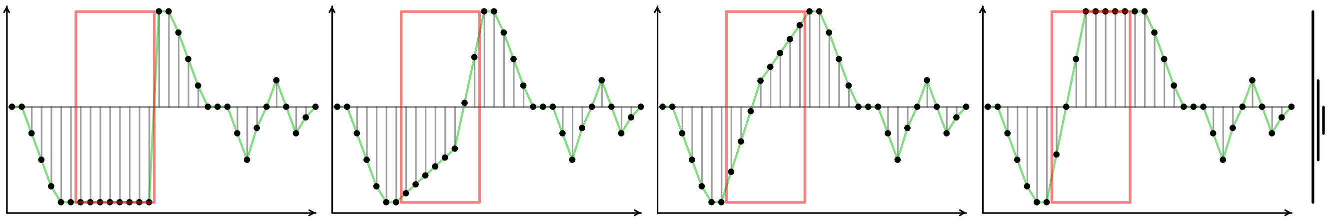


Figure 6: Deformations within one Snake Rectangle (left) at the level of the minimum (middle two) intermediate piece-wise linear interpolations (top) partially at the level of the maximum. Notice that none of these deformations change the bar codes on the right. The green connecting lines are for visualization purpose only and are not meant to imply linear functional values between samples.

requirements) and allow for operation on periodic sequences.

Clearly algorithm 3 has computational complexity of $O(p)$ where p is the number of samples. In the worst case $2 * p - 1$ samples are inspected, when a $p - 1$ length monotone leads into an extremum pair.

9. MONOTONE DEFORMATIONS WITHIN A SNAKE BOX

In principle, any monotone deformation that stays within the bounds of a snake box is permissible. To illustrate concrete deformation we implemented straight-forward piecewise linear interpolation with one intermediate point (that can either interactively or programmatically controlled). Piece-wise linear signal synthesis has been proposed by Bernstein and Cooper [34] and hence this can be thought of as a topologically preserving version of that technique in this case. Four example deformations using this approach can be seen in Figure 6. The leftmost subfigure shows a linear interpolation that at all sample points aligns with the local minimum bounding the snake box. This deformation is permissible because it does not alter the bar code. If we reran the snake box algorithm it would however now identify these points to a flat minimum and not a monotone segment. This is the ambiguity of flats belonging to extrema or monotones previously mentioned. The middle two subfigures show piecewise linear interpolations strictly away from the adjacent extrema. The final example shows a subset of samples at the level of the adjacent maximum while the remaining samples linearly and monotonically descent to the minimum. This again is permissible.

10. RESIZING SNAKE BOXES

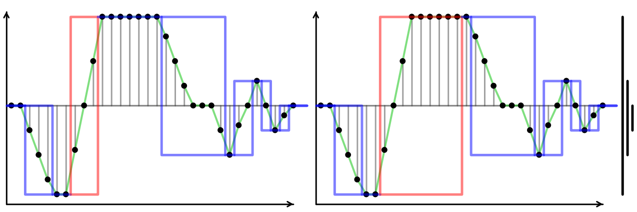
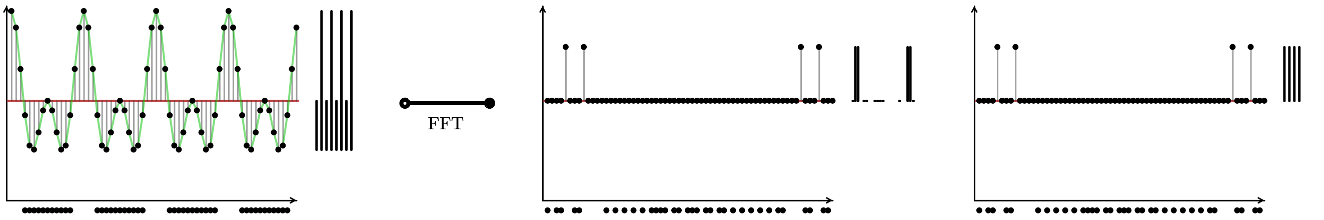


Figure 7: Neighboring extrema and monotone boxes can be resize. (Left) All samples at the flat level of extremum are part of the extremum. (Right) All but one sample at the extremum are moved into the neighboring monotone box and could be deformed there.

Algorithm 3 Construct Box Snake for a Data Set

```
Require: data, i                                    ▷ i is an arbitrary starting index
Require: class Box {sx,sy,ex,ey,dir,type}
Boxes ← empty stacks of type Box                    ▷ Stores box snake
p ← data.length                   ▷ p is the modulus of the periodic series
f← i                                           ▷ Start of a potential flat skip
i← skipFlats(i,p)                              ▷ Advance index across flats
if i = f+p then                                          ▷ Everything is flat?
    return "Constant function. Betti-0 is 1, Betti-1 is 1."
end if
dir ← sign(data[i]−data[i − 1])                  ▷ Get inclination direction
                                                      ▷ Skip initial monotone
while (data[i]−data[i − 1])∗dir ≥ 0 do
    f ← i
    i ← skipFlats(i,p)
end while                          ▷ Loop ends if direction changes at extremum
periodstart ← i                                             ▷ Start position
                                                ▷ Process one period from here
while i <periodstart+p do
                                                        ▷ Record an Extremum.
    boxes.push(new Box(
            f,data[f mod p],i − 1,data[i − 1 mod p],
            dir,"extremum"))
    constructBar(data[i − 1 mod p,dir)                     ▷ Construct Bar
    dir ← −dir                                       ▷ New monotone direction
                                            ▷ Find potential Monotone Sequence.
    sm ← i                                         ▷ Start of monotone sequence
    smdata ← data[(i − 1) mod p]                     ▷ Data before sequence
                                                           ▷ Follow monotone
    while (data[(i) mod p]−data[(i − 1) mod p])∗dir ≥ 0 do
        f ← i
        i ← skipFlats(i,p)

        if sm < f then                               ▷ Is Monotone non-empty?
                                                  ▷ Record a non-empty Monotone
            boxes.push(new Box(
                sm,data[(sm−1) mod p],f−1,data[f mod p],
                dir,"monotone"))
        end if
    end while
end while
```



(a) Time Domain

(b) Frequency Domain without Flat Noise Tolerance

(c) Frequency Domain with Flat Noise Tolerance

Figure 8: Time domain signal (a) of two additively mixed sinusoids at a 1:2 ratio and their real symmetric Fast Fourier Transformed (FFT) Frequency Response with (b) strict flatness and with (c) flat noise level tolerance. Notice that the flat noise reduction removes small barcodes due to numerical inaccuracies of the FFT.

Snake boxes are not necessarily uniquely defined. Specifically, flat extrema and neighboring monotone boxes are ambiguously defined. Any connected part of the flat can be interpreted as belonging to the extremum or the monotone section as long as at least one sample remains with the extremum to preserve the bar code. This leads to the ability to resize boxes along extrema flats as illustrated in Figure 7. The resizing of flats implies that, within any current flat, boxes can be resized to move extrema.

11. GLOBAL EDITING AND DEFORMATION

In principle snake boxes can be edited independently. However, it can be convenient to manipulate all boxes in a snake together. Snake boxes can, however, have very different sizes. Various globalized editing schemes are thinkable, such as editing all snake boxes of same width, height, or treating editing points within them relative to box dimensions.

We implemented a percentage scheme that places editing points relative to the box dimensions in order to globalize editing points. Furthermore, we explored an option to mirror the horizontal editing dimension to match the slope, hence edits will create identical but mirrored slopes. This mimics the the typical behavior of pure sinusoidal functions that have a mirror symmetry around extrema.

Global resizing of snake boxes is implemented as grow-or-shrink-over-flats motions that are propagated through all boxes, and performed if permissible locally.

12. NUMERICAL CONSIDERATIONS

A strict implementation of algorithm 3 treats any variation of the signal at any level. Hence a very noisy signal may contain great numbers of short bars. Given that we consider both time and frequency domain signals, numerical inaccuracy of the discrete Fourier transform used will also cause noise-like variations. A comprehensive solution of denoising via sublevel set persistent homology operates on removing consideration of very short bar codes [16] which in our context translates into also reducing the number of box snakes. In practice we found that variations introduced by numerical inaccuracies tend to primarily create spurious bar codes when they perturb flat levels. Variations on monotone inclines from numerical inaccuracies tend to be so small as to not form local extrema. Hence the practical and computationally cheap solution is to modify the function 2 from checking and skipping exact flatness, to allow for very small variations. Hence if we replace the non-flat condition in algorithm 2 with the following $|\text{data}[i \bmod p] - \text{data}[(i+1) \bmod p]| > \text{noiselevel}$ with a small but non-zero *noiselevel*, flat skipping will become sufficiently insensitive to variations.

To illustrate this process, consider the time-domain mixture of two sine functions of Figure 8, its symmetric real spectrum via a FFT, all using 64 data points, with and without flats tolerant of a *noiselevel* of 10^{-12}. The red levelset line is placed at zero and we can see from the sub levelset that not all points at zero are included that are expected. However, with our noise tolerance in flat detection, the unwanted barcodes disappear leaving only the desired ones.

13. CONCLUSIONS

In this paper we developed a method to resynthesis and manipulate finite-length digital audio data that preserve the topology of the signal via sublevel set persistent homology. We introduced the notion of a box snake to capture various aspects of deformation permissible within a given topological constraint. We demonstrated that treating the finite length audio data as periodic in the domain both fits traditional time-frequency analysis of discrete signals as well as avoids undesirable properties in the topological characterization. Ultimately this leads to a scheme of manipulating audio data up to any monotone deformation on monotone segments between potentially flatly extended extrema, as well as the ability to resize and change flat extrema sizes. Our approach removes technical requirements usually associated with sublevel set persistence of functions, given that we deal only with finite length discrete data. Local and global editing schemes are illustrated on examples. This paper suggests deformation directly on audio signals as compared to previous work that deformed oscillators or used sheaf construction.

However, in many ways the current paper is just a first step in understanding the relationship of topological properties of time and frequency domains. There is hope that the relationship of properties between these two domains can be systematically studied as the discrete Fourier transform is a maximum rank linear map over the complex numbers, and topological characterizations of barcodes is increasingly understood in terms of modules. This characterization is future work.

14. ACKNOWLEDGMENTS

I gratefully acknowledge insightful discussions with Robin Benton and Nicole Sanderson on the topic of discrete sublevel set persistence, and helpful comments from four anonymous reviewers.

15. REFERENCES

[1] Georg Essl, "Topologizing Sound Synthesis via Sheaves," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, 2021, pp. 260–267.

[2] Georg Essl, "Deforming the Oscillator: Iterative phases over parametrizable closed paths," in *Proceedings of the International Conference on Digital Audio Effects (DAFX)*, 2022.

[3] Paul Bendich, Herbert Edelsbrunner, Dmitriy Morozov, and Amit Patel, "Homology and Robustness of Level and Interlevel Sets," *Homology, Homotopy and Applications*, vol. 15, no. 1, pp. 51–72, 2013.

[4] Wojciech Reise, Bertrand Michel, and Frédéric Chazal, "Topological signatures of periodic-like signals," *arXiv preprint arXiv:2306.13453*, 2023.

[5] Herbert Edelsbrunner and John L. Harer, *Computational topology: An introduction*, American Mathematical Society, 2010.

[6] Tamal Krishna Dey and Yusu Wang, *Computational Topology for Data Analysis*, Cambridge University Press, 2022.

[7] Robert W. Ghrist, *Elementary Applied Topology*, Createspace, 2014.

[8] Michael Robinson, *Topological Signal Processing*, Springer, 2014.

[9] Nalini Ravishanker and Renjie Chen, "Topological Data Analysis (TDA) for Time Series," *arXiv preprint arXiv:1909.10604*, 2019.

[10] Jose A. Perea, "Topological Time Series Analysis," *Notices of the American Mathematical Society*, vol. 66, no. 5, pp. 686–694, 2019.

[11] Nicole Sanderson, Elliott Shugerman, Samantha Molnar, James D. Meiss, and Elizabeth Bradley, "Computational topology techniques for characterizing time-series data," in *Advances in Intelligent Data Analysis 16*, 2017, pp. 284–296.

[12] Yuliy Baryshnikov, "Time Series, Persistent Homology and Chirality," *arXiv preprint arXiv:1909.09846*, 2019/2022.

[13] Yu-Min Chung, William Cruse, and Austin Lawson, "A persistent homology approach to time series classification," *arXiv preprint arXiv:2003.06462*, 2020.

[14] Sunia Tanweer, Firas A Khasawneh, and Elizabeth Munch, "Robust Zero-crossings Detection in Noisy Signals using Topological Signal Processing," *arXiv preprint arXiv:2301.07703*, 2023.

[15] Audun D. Myers and Firas A. Khasawneh, "Damping parameter estimation using topological signal processing," *Mechanical Systems and Signal Processing*, vol. 174, pp. 109042, 2022.

[16] Audun D. Myers, Firas A. Khasawneh, and Brittany T. Fasy, "ANAPT: Additive Noise Analysis for Persistence Thresholding," *Foundations of Data Science*, vol. 4, no. 2, pp. 243–269, 2022.

[17] Audun D. Myers, Melih Yesilli, Sarah Tymochko, Firas Khasawneh, and Elizabeth Munch, "Teaspoon: A comprehensive python package for topological signal processing," in *TDA & Beyond*, 2020.

[18] Justin Curry, "The Fiber of the Persistence Map for Functions on the Interval," *Journal of Applied and Computational Topology*, vol. 2, pp. 301–321, 2018.

[19] Lida Kanari, Adélie Garin, and Kathryn Hess, "From Trees to Barcodes and Back Again: Theoretical and Statistical Perspectives," *Algorithms*, vol. 13, no. 12, pp. 335, 2020.

[20] Justin Curry, Jordan DeSha, Adélie Garin, Kathryn Hess, Lida Kanari, and Brendan Mallery, "From Trees to Barcodes and Back Again II: Combinatorial and Probabilistic Aspects of a Topological Inverse Problem," *Computational Geometry*, vol. 116, pp. 102031, 2024.

[21] Robin Belton, Bree Cummins, Brittany T. Fasy, and Tomáš Gedeon, "Extremal Event Graphs: A (Stable) Tool for Analyzing Noisy Time Series Data," *arXiv preprint arXiv:2203.09552*, 2022.

[22] Vladimir I. Arnol'd, "The calculus of snakes and the combinatorics of Bernoulli, Euler and Springer numbers of Coxeter groups," *Russian Mathematical Surveys*, vol. 47, no. 1, pp. 1, 1992.

[23] Nikolas Valsamakis, *Non-Standard Sound Synthesis with Dynamic Models*, Ph.D. thesis, University of Plymouth, 2013.

[24] Curtis Roads, *The Computer Music Tutorial*, MIT press, 1996.

[25] Renjie Chen, Jingyue Zhang, Nalini Ravishanker, and Karthik Konduri, "Clustering Activity-Travel Behavior Time Series using Topological Data Analysis," *Journal of Big Data Analytics in Transportation*, vol. 1, pp. 109–121, 2019.

[26] Yuan Wang, Hernando Ombao, and Moo K. Chung, "Topological Data Analysis of Single-Trial Electroencephalographic Signals," *The Annals of Applied Statistics*, vol. 12, no. 3, pp. 1506, 2018.

[27] Nicolas Boutry, Laurent Najman, and Thierry Géraud, "Some Equivalence Relation between Persistent Homology and Morphological Dynamics," *Journal of Mathematical Imaging and Vision*, vol. 64, no. 7, pp. 807–824, 2022.

[28] Marc Glisse, "Fast Persistent Homology Computation for Functions on $\mathbb{R}$," *arXiv preprint arXiv:2301.04745*, 2023.

[29] Sebastiano Cultrera di Montesano, Herbert Edelsbrunner, Monika Henzinger, and Lara Ost, "Dynamically Maintaining the Persistent Homology of Time Series," in *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2024, pp. 243–295.

[30] Georg Essl, "Topology in Sound Synthesis and Digital Signal Processing–DAFx2022 Lecture Notes," *arXiv preprint arXiv:2211.05821*, 2022.

[31] Georg Essl, "Combinatorial Hodge Theory in Simplicial Signal Processing – DAFx2023 Lecture Notes," *arXiv preprint arXiv:2311.03469*, 2023.

[32] Brian C. J. Moore, *An Introduction to the Psychology of Hearing*, Brill, 6th edition, 2012.

[33] Ken Steiglitz, *A Digital Signal Processing Primer, with Applications to Digital Audio and Computer Music*, Addison Wesley, 1997.

[34] Alan D. Bernstein and Ellis D. Cooper, "The Piecewise-Linear Technique of Electronic Music Synthesis," *Journal of the Audio Engineering Society*, vol. 24, no. 6, pp. 446–454, 1976.