

Topological Data Analysis (TDA) for Time Series

Nalini Ravishanker^{*1} and Renjie Chen^{†1}

¹Department of Statistics, University of Connecticut

Abstract

The study of topology is strictly speaking, a topic in pure mathematics. However in only a few years, Topological Data Analysis (TDA), which refers to methods of utilizing topological features in data (such as connected components, tunnels, voids, etc.) has gained considerable momentum. More recently, TDA is being used to understand time series. This article provides a review of TDA for time series, with examples using R functions. Features derived from TDA are useful in classification and clustering of time series and in detecting breaks in patterns.

Keywords Periodicity, Persistence Diagram, Persistence Landscape, Point clouds, Sublevel sets on functions, Supervised learning, Takens's embedding, Unsupervised learning.

1 Introduction

Topological Data Analysis (TDA) is now an emerging area for analyzing complex data. TDA refers to a class of methods that garner information from topological structures in data that belong to a topological space, i.e., a mathematical space that allows for continuity, connectedness, and convergence (Carlsson, 2009; Edelsbrunner and Harer, 2010). Output from TDA may then be used for effective statistical learning about the data. TDA combines algebraic topology and other tools from pure mathematics to allow a useful study of *shape* of the data. The most widely discussed topologies of data include connected components, tunnels, voids, etc., of a topological space. Computational (or algorithmic) topology, is an overlap between the mathematical underpinnings of topology with computer science, and consists of two parts, i.e., measuring the topology of a space and persistent homology (Chazal and Michel, 2017). Using computational topology, TDA aims at analyzing topological features of data and representing these features using low dimensional representations (Carlsson, 2009). In particular, the space must first be represented as simplicial complexes, the Vietoris-Rips complex and the Čech complex being the most common pathways to obtaining output to characterize the topology.

Persistent homology refers to a class of methods for measuring topological features of shapes and functions. It converts the data into simplicial complexes and describes the topological structure of a space at different spatial resolutions. Topologies that are more persistent are detected over a wide range of spatial scales and are deemed more likely to represent true features of the underlying space rather than sampling variations, noise, etc. Persistent homology therefore elicits persistence of

^{*}nalini.ravishanker@uconn.edu

[†]renjie.chen@uconn.edu

essential topologies in the data and outputs the birth and death of such topologies via a persistence diagram, which is a popular summary statistic in TDA. Data inputs for persistent homology are usually represented as point clouds or as functions, while the outputs depend on the nature of the analysis and commonly consist of either a persistence diagram, or a persistence landscape. A point cloud of data represents a sample of points from an underlying manifold and its persistent homology approximates the topological information of the manifold. If data is represented as a Morse function (i.e., a smooth function on the manifold such that all critical points are non-degenerate), the persistent homology of the function is mathematically equivalent to analyzing the topological information of the manifold. For rigorous expositions on algebraic topology and computational homology, see Munkres (1993) and Edelsbrunner and Harer (2010).

TDA has been used in cosmic web (Van de Weygaert et al., 2011), shape analysis (Carlsson et al., 2004; Chazal et al., 2009; Di Fabio and Landi, 2011, 2012; Chazal et al., 2014; Li et al., 2014; Carrière et al., 2015; Bonis et al., 2016), biological data analysis (DeWoskin et al., 2010; Nicolau et al., 2011; Heo et al., 2012; Kovacev-Nikolic et al., 2016; Bendich et al., 2016; Wang et al., 2018), sensor networks (Silva and Ghrist, 2007; De Silva and Ghrist, 2007; Adams and Carlsson, 2015), as well as other fields.

Development of TDA for time series is a relatively new and fast growing area, with many interesting applications in several different domains. Berwald et al. (2013) discussed the use of TDA in climate analysis. Khasawneh and Munch (2016) used notions of persistence of 1-th homology groups of point clouds (obtained via Takens’s embedding) within multiple windows of time series to track the stability of dynamical systems, while Seversky et al. (2016) explored stability of various single-source and multi-source signals. Perea and Harer (2015) used the notion of maximum persistence of homology groups to quantify periodicity of time series. Pereira and de Mello (2015) used features derived from persistent homology to cluster populations of *Tribolium* flour beetles. Umeda (2017) used topological features of one and two dimensional homology groups as inputs into convolutional neural networks for classification of time series in three different domains, showing that their approach outperformed the baseline algorithm in each case. One illustration consisted of motion sensor data of daily and sports activities, an area also investigated using TDA by Stolz et al. (2017). Truong (2017) as well as Gidea (2017); Gidea and Katz (2018) and Gidea et al. (2018) explored the use of TDA on financial time series. We discuss some of these applications in detail later in this paper.

It is well known that time series do not naturally have point cloud representations. Transformation from a time series to a point cloud is implemented through Takens’s embedding (Takens et al., 1981), guaranteeing the preservation of topological properties of the time series. The approach consists of transforming a time series $\{x_t, t = 1, 2, \dots, T\}$, into its phase space, i.e., a point cloud or a set of points $\mathbf{v}_i = \{x_i, x_{i+\tau}, \dots, x_{i+d\tau}\}, i = 1, 2, \dots, T - d\tau$, where τ is a delay parameter and d specifies the dimension of the point cloud. We discuss Taken’s embedding in Section 2.3 and the selection of d and τ in Section 2.3.1. TDA of time series through suitable functions is much less explored. Wang et al. (2018) proposed TDA on weighted Fourier series representations (Morse functions) of electroencephalogram (EEG) data. They used a randomness test approach to examine properties of the proposed method and show its robustness to different transformations of the data. TDA of time series through sublevel set filtration of functions is discussed in Section 3.

The format of this paper follows. Section 2 provides a review of TDA from point clouds and then describes TDA for time series via the Takens’s embedding method. Section 3 provides a review of persistent homology on functions and then describes TDA for time series analysis starting from second-order spectra or Walsh Fourier transforms. Section 4 discusses constructing TDA based features which are then used in learning about time series, with applications on classification,

clustering and detecting changes in patterns. Section 5 gives a discussion and summary.

2 Persistent Homology Based on Point Clouds

In the section, we first describe persistent homology of a manifold starting from point cloud data, followed by its construction and use in time series analysis using Takens's embedding.

2.1 Point Clouds to Persistence Diagrams - A Basic Review

Starting from a point cloud, we show the procedure to elicit topological features of data. Denote the point cloud as $\mathcal{P} = \{\mathbf{v}_i : i = 1, 2, \dots, N\}$, where $\mathbf{v}_i \in \mathcal{R}^d$. When $d = 2$, the points lie on the plane. Let $\mathbf{D}_E = \{D_E(\mathbf{v}_i, \mathbf{v}_j)\}$ be the $N \times N$ matrix of Euclidean distances, for $i, j = 1, \dots, N$. For each $\mathbf{v}_i \in \mathcal{P}$, let $\mathbf{B}_\lambda(\mathbf{v}_i) = \{\mathbf{x} : D_E(\mathbf{x}, \mathbf{v}_i) \leq \lambda/2, \mathbf{x} \in \mathcal{R}^d\}$ denote a closed ball with radius $\lambda/2$; here, $0 \leq \lambda \leq U$, where the upper-bound U is usually pre-determined as the maximum of the distances in \mathbf{D}_E . A Vietoris-Rips simplex (Edelsbrunner and Harer, 2010) corresponding to a given λ is defined as the set of points $\mathcal{P}_V(\lambda) \subset \mathcal{P}$ such that any points $\mathbf{v}_{i_1}, \mathbf{v}_{i_2}$ in $\mathcal{P}_V(\lambda)$ satisfy $D_E(\mathbf{v}_{i_1}, \mathbf{v}_{i_2}) \leq \lambda$, $1 \leq i_1, i_2 \leq N$. For a given λ value, a simplicial complex $\tilde{\kappa}(\lambda)$ denotes the set of Vietoris-Rips simplexes such that for any two Vietoris-Rips simplexes $\mathcal{P}_V^{(1)}(\lambda), \mathcal{P}_V^{(2)}(\lambda) \in \tilde{\kappa}(\lambda)$, we have (i) $\mathcal{P}_V^{(1)}(\lambda) \cap \mathcal{P}_V^{(2)}(\lambda) \in \tilde{\kappa}(\lambda)$ and (ii) if $\mathcal{P}' \subset \mathcal{P}_V^{(1)}(\lambda)$, then $\mathcal{P}' \in \tilde{\kappa}(\lambda)$.

A simplicial complex consisting of $(\tilde{p} + 1)$ points (from different Vietoris-Rips simplexes) is a \tilde{p} -dimensional simplicial complex. In algebraic topology, \tilde{p} is at most $N - 1$ when the point cloud had N points. The topology of the point cloud is studied through the topology of the simplicial complexes, denoted by $\{\tilde{\alpha}_{\tilde{p},k} : k = 1, 2, \dots, k_{\tilde{p}}\}$, and $\tilde{\alpha}_{\tilde{p},k}$ is a homology group, consisting of a set of \tilde{p} -dimensional simplicial complexes which are homomorphic. For the theory and computation of homomorphisms, refer to Munkres (1993); Carlsson (2014) and Edelsbrunner and Harer (2010). As the parameter λ gradually increases, the birth and death of homology groups $\{\tilde{\tau}_{\tilde{p},k} = (\lambda_{\tilde{p},k,1}, \lambda_{\tilde{p},k,2}) : k = 1, 2, \dots, k_{\tilde{p}}\}$ are recorded in the persistence diagram. A \tilde{p} -th Betti number of λ is the number of \tilde{p} -th homology groups at λ , denoted as $k_{\tilde{p}}^{(\lambda)}$.

Computation of the topological features are summarized in the following steps.

Step 1. Compute the Euclidean distance matrix $\mathbf{D}_E = \{D_E(\mathbf{v}_{i_1}, \mathbf{v}_{i_2})\}$ for $i_1, i_2 \in \{1, \dots, N\}$; this is the default distance for a point cloud in R-TDA.

Step 2. Construct birth and death of homology groups for increasing values of λ . For each λ , compute $\tilde{\alpha}_{\tilde{p},k}$ from $\tilde{\kappa}(\lambda)$ using closed balls $\mathbf{B}_\lambda(\mathbf{v}_i)$ of \mathbf{v}_i with radius $\lambda/2$. If an elder topology $\tilde{\alpha}_{\tilde{p},k_1}$ and a younger one $\tilde{\alpha}_{\tilde{p},k_2}$ merge into a single $\tilde{\alpha}_{\tilde{p},k}$ at some λ , $\tilde{\alpha}_{\tilde{p},k_1}$ would become $\tilde{\alpha}_{\tilde{p},k}$ and $\tilde{\alpha}_{\tilde{p},k_2}$ would die.

Step 3. The persistence diagram is an output of the set of points representing birth-death of homology groups from the point cloud and is denoted as $\tilde{\Omega} = \{\tilde{\tau}_{\tilde{p},k} = (\lambda_{\tilde{p},k,1}, \lambda_{\tilde{p},k,2}) : \tilde{p} = 0, 1, \dots; k = 1, 2, \dots, k_{\tilde{p}}\}$. We plot $\lambda_{\tilde{p},k,1}$ on the x -axis and $\lambda_{\tilde{p},k,2}$ on the y -axis (Edelsbrunner and Harer, 2010).

Example 2.1. Point Cloud to Persistence Diagram. We illustrate construction of the persistence diagram for a point cloud with $N = 60$ points sampled from the unit circle $x_1^2 + x_2^2 = 1$:

```
set.seed(1); PC <- circleUnif(n = 60, r = 1)
plot(PC, main = "(a)")
```

The point cloud is shown in Figure 1(a). We expect to see a total of $k_0 = 60$ values of $\tilde{\alpha}_{0,k}$ and $k_1 = 1$ value of $\tilde{\alpha}_{1,k}$. We use the function `ripsDiag` from the `R-TDA` package for constructing the persistence diagram (Fasy et al., 2014a). In the R code chunk shown below, `PC` denotes the input point cloud, `maxdimension` is the maximum dimension \tilde{p} of points $\tilde{\tau}_{\tilde{p},k}$ to be calculated, and `maxscale` is the maximum value that the filtration parameter λ can assume. We set `maxdimension` to be 1. The default `dist` is the Euclidean distance. The output `pers.diag.1` returns the persistence diagram, as a matrix with three columns which summarize topological features of the point cloud.

```
(pers.diag.1 <- ripsDiag(X=PC, maxdimension = 1, maxscale = max(dist(PC))) )
$diagram
      dimension      Birth      Death
[1,]          0 0.0000000 1.999999902
[2,]          0 0.0000000 0.306978455
[3,]          0 0.0000000 0.245260715
.....
[60,]         0 0.0000000 0.027923092
[61,]         1 0.3190835 1.737696840
```

The first row with $\tilde{\tau}_{0,1} = (0, 2.00)$ in the output records that there is a 0-th homology group (connected component) whose birth happens at $\lambda = 0$ and whose death happens at about $\lambda = 2.00$. The second row with $\tilde{\tau}_{0,1} = (0, 0.31)$ records that the second connected component is born at $\lambda = 0$ and is dead at $\lambda = 0.31$, etc. We see that all 0-th homology groups have birth time 0, and all $N = 60$ points start as connected components. These 60 connected components are shown in decreasing order of persistence (slower death). Row 61 with $\tilde{\tau}_{1,1} = (0.32, 1.74)$ describes the birth and death of a 1-th homology group (tunnel) at $\lambda = 0.32$ and $\lambda = 1.74$ respectively. Figure 1(b) corresponds to the filtration parameter $\lambda = 0$ and is obtained using this code:

```
plot(x=1,y=1,type="n",ylim=c(0,2),xlim=c(0,2),ylab="death",xlab="birth",main="(b)")
abline(v = 0, lty = 2)
```

The dashed vertical line indicates the birth time of connected components; the plot has no points because none of the connected components has died. Figure 1(c) corresponds to the point cloud when $\lambda = 0.1$:

```
plot(PC, pch = 16, cex = 5, col = "blue", main = "(c)")
```

The blue balls $\mathbf{B}_\lambda(\mathbf{v}_i)$ around each point enlarge and connect with others, resulting in fewer connected components. The black dots in Figure 1(d) denote the birth-death times of the merged connected components which have died before $\lambda = 0.1$:

```
death.time = sort(pers.diag.1$diagram[pers.diag.1$diagram[, 1]==0, 3])
plot(x = rep(0, sum(death.time<=0.1)), y = death.time[which(death.time<=0.1)],
      ylim = c(0, 2), xlim = c(0, 2), ylab = "death", xlab = "birth", main = "(d)")
abline(v = 0, lty = 2); abline(h = 0.1, lty = 2)
```

When $\lambda = 0.32$ in Figure 1(e), all points connect together and a tunnel emerges, which is the white area surrounded by the blue circle:

```
plot(PC, pch = 16, cex = 12, col = "blue", main = "(e)")
```

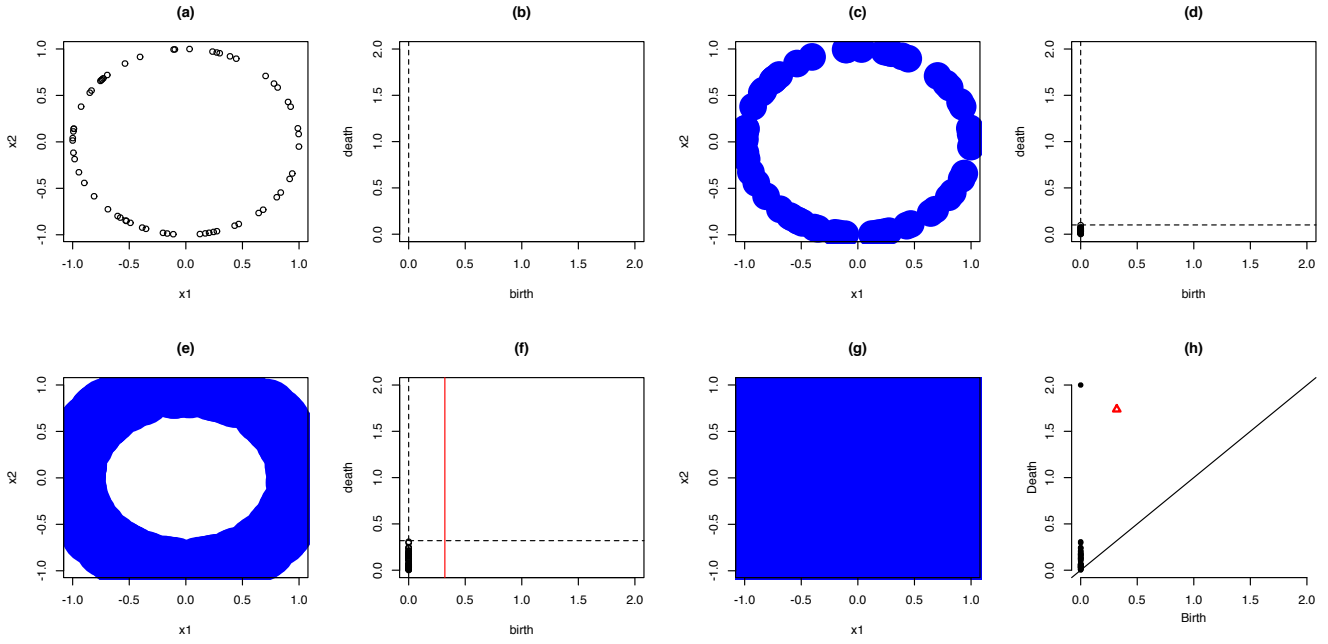


Figure 1: Persistence diagram corresponding to a point cloud. (a) shows the raw point cloud and (h) shows the persistence diagram. (c), (e) and (g) are intermediate steps for the filtration by varying λ , while (b), (d) and (f) are intermediate steps for constructing the persistence diagram.

The birth time of this tunnel is recorded as $\lambda = 0.32$, which is shown as the red dashed line in Figure 1(f). Further, there are more black dots in this figure since there are more connected components that have died before $\lambda = 0.32$:

```
plot(x = rep(0, sum(death.time<=0.32)), y = death.time[which(death.time<=0.32)],
ylim = c(0, 2), xlim = c(0, 2), ylab = "death", xlab = "birth", main = "(f)")
abline(h = pers.diag.1$diagram[pers.diag.1$diagram[, 1]==1, 2], lty = 2)
abline(v = 0, lty = 2); abline(v = 0.32, col = "red", lty = 1)
```

When λ reaches its maximum value of 2 (which is the largest value in $D_E(\mathbf{v}_{i_1}, \mathbf{v}_{i_2})$), the algorithm stops and outputs the persistence diagram (see Figure 1(h)) which finally shows the birth-death times for all connected components (the dots) and the tunnel (the red triangle):

```
plot(PC, pch = 16, cex = 40, col = "blue", main = "(g)")
plot(pers.diag.1$diagram, main = "(h)")
```

R-TDA also supports construction of a persistence diagram given an arbitrary distance matrix as input, as shown in the example below.

Example 2.2. Distance Matrix to Persistence Diagram. The input to `ripsDiag` can be a distance matrix computed from the point cloud generated in Example 2.1. Here, we use the default `dist="euclidean"`. Other options are "manhattan", "maximum", etc.

```
dist.PC <- dist(PC)
(pers.diag.2=ripsDiag(X=dist.PC,dist="arbitrary",maxdimension=1,maxscale=max(dist.PC)))
$diagram
```

	dimension	Birth	Death
[1,]	0	0.0000000	1.999999902
[2,]	0	0.0000000	0.306978455
[3,]	0	0.0000000	0.245260715
.....			
[60,]	0	0.0000000	0.027923092
[61,]	1	0.3190835	1.737696840

2.2 Distances Between Persistence Diagrams

Two distance metrics are commonly used to quantify the dissimilarity between two persistence diagrams $\tilde{\Omega}_1$ and $\tilde{\Omega}_2$, the Wasserstein distance and the bottleneck distance (Mileyko et al., 2011). We define these distances and describe their computation using the R-TDA package.

The q -Wasserstein distance between two persistence diagrams is defined by

$$W_{q,\tilde{p}}(\tilde{\Omega}_1, \tilde{\Omega}_2) = \left[\inf_{\eta: \tilde{\Omega}_1 \rightarrow \tilde{\Omega}_2} \sum_{\tilde{\tau}_{\tilde{p},k} \in \tilde{\Omega}_1} |\tilde{\tau}_{\tilde{p},k} - \eta(\tilde{\tau}_{\tilde{p},k})|_\infty^q \right]^{1/q}, q = 1, 2, \dots, \quad (1)$$

where \tilde{p} is referred to as its dimension and q is its degree. When $q = \infty$, (1) is the bottleneck distance of dimension \tilde{p} defined by

$$W_{\infty,\tilde{p}}(\tilde{\Omega}_1, \tilde{\Omega}_2) = \inf_{\eta: \tilde{\Omega}_1 \rightarrow \tilde{\Omega}_2} \sup_{\tilde{\tau}_{\tilde{p},k} \in \tilde{\Omega}_1} |\tilde{\tau}_{\tilde{p},k} - \eta(\tilde{\tau}_{\tilde{p},k})|_\infty. \quad (2)$$

The bottleneck distance is obtained by minimizing the largest distance of any two corresponding points of diagrams, over all bijections between $\tilde{\Omega}_1$ and $\tilde{\Omega}_2$ and is less sensitive to details in the diagrams.

Example 2.3. Wasserstein and Bottleneck Distances. Let $\tilde{\Omega}_1$ be `pers.diag.2`, the persistence diagram obtained in Example 2.2. Let $\tilde{\Omega}_2$ be `pers.diag.3`, the persistent diagram we construct from a different point cloud from the same unit circle. We use R-TDA to compute the Wasserstein distance with $q = 1$ (denoted by the argument `p=1` below):

```
set.seed(2); PC2 <- circleUnif(n = 60, r = 1)
pers.diag.3 <- ripsDiag(X=PC2, maxdimension = 1, maxscale = max(dist(X))$diagram
wasserstein(pers.diag.2, pers.diag.3, p=1, dimension = 0)
1.034579
```

The function `bottleneck` enables us to compute the bottleneck distance between the two persistence diagrams. The Wasserstein distance is larger than the bottleneck distance since the former measures more detailed difference between the diagrams.

```
bottleneck(pers.diag.2, pers.diag.3, dimension = 0)
0.06954618
```

It is important to construct persistence diagrams using the same distance functions (Chazal et al., 2018), as we show below. For instance, we can construct a persistence diagram `pers.diag.4` for the point cloud in Example 2.1 using the Manhattan distance ($D_M(\mathbf{v}_i, \mathbf{v}_j) = \sum_{\ell=1}^d |v_{i,\ell} - v_{j,\ell}|$, $1 \leq i, j \leq N$) instead of the Euclidean distance.

```

dist.PC.man <- dist(PC, method = "manhattan"); max.dist=max(dist.PC.man)
pers.diag.4=ripsDiag(X=dist.PC.man,dist="arbitrary",maxdimension=1,
maxscale=max.dist)$diagram
wasserstein(pers.diag.2, pers.diag.4, p=1, dimension = 0)
2.145083
bottleneck(pers.diag.2, pers.diag.4, dimension = 0)
0.8279462

```

2.3 TDA of Time Series via Point Clouds

Time series do not naturally have point cloud representations, and are transformed to point clouds using Takens's Embedding Theorem (Takens et al., 1981) before we can do TDA as discussed in Section 2.1. This approach has been used in the literature mostly for quantifying periodicity in time series (Perea and Harer, 2015), clustering time series (Seversky et al., 2016), classifying time series (Umeda, 2017), or finding early signals for critical transitions (Gidea, 2017; Gidea and Katz, 2018). Takens's embedding guarantees the preservation of topological properties of a time series but not its geometrical properties.

2.3.1 Takens's Delay Embedding for Time Series

Let $\{x_t, t = 1, 2, \dots, T\}$ denote an observed time series. We use Takens's embedding to convert the time series into a point cloud with points $\mathbf{v}_i = (x_i, x_{i+\tau}, \dots, x_{i+(d-1)\tau})'$, where d specifies the dimension of the points and τ denotes a delay parameter. For example, if $d = 2$ and $\tau = 1$, then, $\mathbf{v}_i = (x_i, x_{i+1})'$, whereas if $d = 15$ and $\tau = 2$, $\mathbf{v}_i = (x_i, x_{i+2}, \dots, x_{i+28})'$. Both d and τ are unknown and must be determined in practice.

Choice of τ . Researchers have used different approaches for choosing the delay parameter τ . It may be selected as the smallest time lag h where the sample autocorrelation function (ACF) $\hat{\rho}_h$ becomes insignificant, i.e., smaller in absolute value than the critical bound $\frac{2}{\sqrt{T}}$ (Khasawneh and Munch, 2016). Truong (2017) also used the ACF, but in a slightly different way. He chose τ as the smallest lag for which $(\hat{\rho}_\tau - \hat{\rho}_{\tau-1})/\hat{\rho}_\tau > 1/e$ and $\hat{\rho}_\tau < \frac{2}{\sqrt{T}}$. Pereira and de Mello (2015) determined τ using the first minimum of the auto mutual information (the mutual information between the signal and its time delayed version).

Choice of d . Truong (2017) and Khasawneh and Munch (2016) used the false nearest neighbor method (Kennel et al., 1992) to determine the embedding dimension d as the integer such that the nearest neighbors of each point in dimension d remain nearest neighbors in dimension $d + 1$, and the distances between them also remain about the same. Alternately, an R function `false.nearest` in the package `tseriesChaos` which implements an approach due to Hegger et al. (1999) may be used. Some authors (Pereira and de Mello, 2015; Seversky et al., 2016), simply assume d to be 2 or 3, while Perea has suggested the use of $d = 15$ on time series after a cubic spline interpolation (see Section 2.3.2).

The choice of d and τ then determine the number of points N in the point cloud. In Example 2.4, we illustrate one approach for constructing a point cloud from pure periodic signals with no noise and then obtaining a persistence diagram. In Example 2.5, we discuss another approach described in Perea et al. (2015) for noisy time series, when the focus is on finding series with the same periodicity.

Example 2.4. Pure Signals to Persistence Diagrams. We generate point clouds from three periodic cosine signals of length $T = 480$ with periods 12, 48, and 96 respectively, and then construct their persistent diagrams. We set $d = 2$, and use the ACF method discussed above to choose τ . We show R code for the time series `ts1`:

```
per1=12;ts1 = cos(1:T*2*pi/per1);d=2;
tau <- which(abs(acf(ts.ex, plot = F)$acf) < 2/sqrt(T))[1]-1
PC=t(purrr::map_dfc(1:(T-(d-1)*tau+1),~ts.ex[seq(from=.x, by=tau, length.out=d)]))
diag=ripsDiag(PC, maxdimension=1, maxscale=max(dist(PC)))
ts.plot(ts.ex);plot(PC,xlab = "x1",ylab="x2",main="PC");plot(diag$diagram)
```

In Figure 2, the top row shows the signals, the middle row shows the point clouds and the bottom row shows the persistence diagrams. The black dots represent the birth-death of 0-th homology groups and their persistence shows the dispersion of the points in the point cloud. When there are more black dots close to the diagonal, the point cloud is more dispersed. Particularly, the point cloud PC.3 from the time series with period 96 have points close to each other compared with PC.1, so that it has more black dots in the persistence diagram closer to the diagonal.

The red triangles represent the birth-death of 1-th homology groups, indicating circles in the point cloud. The red triangle from the time series with period 96 is further away from the diagonal compared to the series with period 12, and thus has longer persistence in the 1-th homology group. Seeing a circle indicates that the time series is periodic. This is in contrast to the persistence diagram for the same time series based on sublevel set filtration of a function, as discussed in Example 3.2.

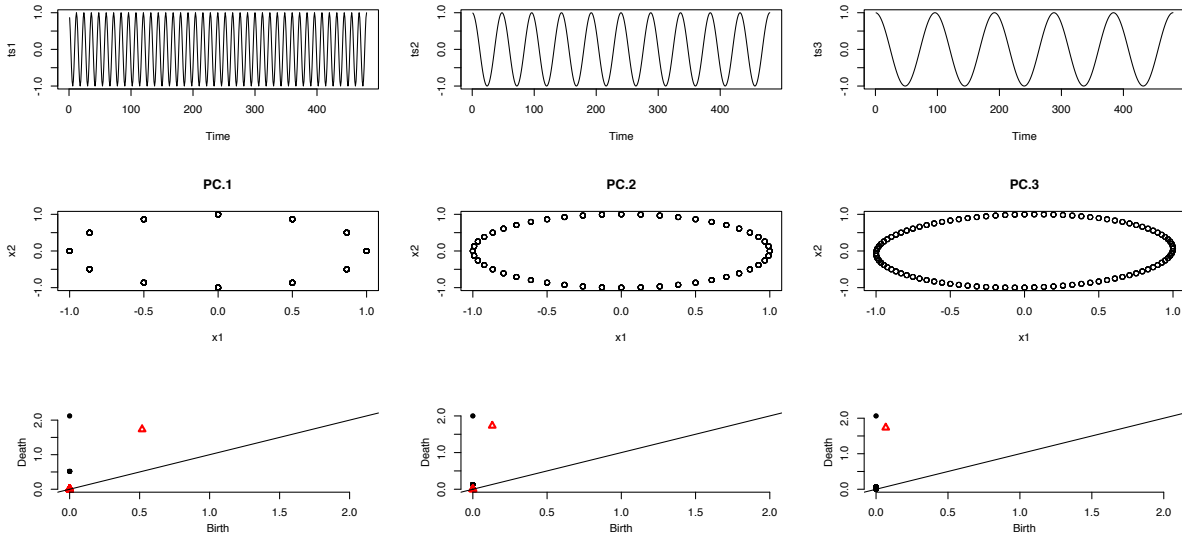


Figure 2: Pure Periodic Signals to Persistence Diagrams.

Pairwise bottleneck distances between the three persistence diagrams are shown in Table 1, computed using code as shown below:

```
round(bottleneck(diag1$diagram, diag2$diagram, dimension = 0), digits = 2)
```

To study the effect of d , we repeat the computations for $d = 3$ and $d = 15$ and also show all pairwise bottleneck distances in the table. While the values of the distances between the diagrams change as

d changes, the relative behavior is preserved, independent of d . Specifically, the bottleneck distances between $\tilde{\Omega}_1$ and $\tilde{\Omega}_2$ and between $\tilde{\Omega}_1$ and $\tilde{\Omega}_3$ are larger than the distance between $\tilde{\Omega}_2$ and $\tilde{\Omega}_3$ in the 0-th and 1-th homology groups.

Table 1: Pairwise Bottleneck Distances Between Persistence Diagrams for Different d .

	$d = 2$			$d = 3$			$d = 15$		
	(1,2)	(1,3)	(2,3)	(1,2)	(1,3)	(2,3)	(1,2)	(1,3)	(2,3)
0th	0.26	0.26	0.07	0.36	0.36	0.09	0.73	0.73	0.18
1th	0.39	0.45	0.06	0.53	0.63	0.09	1.09	1.28	0.19

2.3.2 Point Cloud Construction using SW1PerS

The SW1PerS (Sliding Windows and 1-Persistence Scoring) method is an alternate, more comprehensive approach proposed by Perea et al. (2015) to detect periodicity from noisy time courses whose underlying signals may have different shapes. The approach addresses the following items.

Denoising. The approach considers two types of denoising that are left as options to the user. The first type smooths the raw time series by a moving average in order to make it easier to detect the signal. The second type is a moving average on the point cloud. As an alternative to moving averaging, Pereira and de Mello (2015) used the Empirical Mode Decomposition (EMD) (Huang et al., 1998) on the raw time series.

Spline Interpolation. The spline interpolation allows handling unevenly spaced time series, or time series with low temporal resolution.

Point Cloud Standardization. Standardization helps with signal dampening and to make the procedure amplitude blind.

The pipeline for this approach is described in the following steps.

Step 0. Optionally (Perea and Harer, 2015), denoise the observed time series $\{x_t, t = 1, 2, \dots, T\}$ using a simple moving average whose window size is no higher than one third of the selected dimension d . They recommended an embedding dimension of $d = 15$ and $N = 201$ as the size of the point cloud; then $T_1 = N + d = 216$.

Step 1. For selected values of d and τ (see below), create a point cloud from the (possibly denoised) time series using Steps 1.1 and 1.2.

Step 1.1. Recover a continuous function $g : [0, 2\pi] \rightarrow \mathcal{R}$ by fitting a cubic spline to the denoised time series $\{x_t, t = 1, 2, \dots, T\}$.

Step 1.2. Using values $g(t_1), g(t_2), \dots, g(t_{T_1})$ from the continuous spline fit $g(\cdot)$ at evenly spaced time points $0 = t_1 \leq t_2 \leq \dots \leq t_{T_1} = (T_1 - 1)\tau = 2\pi$, construct a point cloud with $N = T_1 - d$ points $\mathbf{v}_t^{(0)} = (g(t), g(t+\tau), \dots, g(t+(d-1)\tau))' \in \mathcal{R}^d, t = 0, \tau, \dots, 2\pi - (d-1)\tau$ and so $\tau = \frac{2\pi}{N+d-1}$.

Step 2. Pointwise Point cloud standardization:

$$\mathbf{v}_t = \frac{\mathbf{v}_t^{(0)} - \bar{v}_t^{(0)} \mathbf{1}}{\|\mathbf{v}_t^{(0)} - \bar{v}_t^{(0)} \mathbf{1}\|}; \quad \bar{v}_t^{(0)} = \sum_{i=1}^d v_{t,i}^{(0)} / d, \quad \|\mathbf{v}_t^{(0)} - \bar{v}_t^{(0)} \mathbf{1}\| = \sqrt{\sum_{i=1}^d (v_{t,i}^{(0)} - \bar{v}_t^{(0)})^2}, \quad (3)$$

where $\mathbf{v}_t^{(0)} = (v_{t,1}^{(0)}, v_{t,2}^{(0)}, \dots, v_{t,d}^{(0)})'$ and $\mathbf{1}$ is the d -dimensional vector of 1's.

Step 3. Construct the persistence diagram from the point cloud as shown in Section 2.1.

This method is powerful for detecting periodicity in time series. To develop a score for quantifying the periodicity, Perea et al. (2015) first found the longest persistence of the birth-death of the 1-th homology groups $(\lambda_{1,k_M,1}, \lambda_{1,k_M,2})$, where $k_M = \arg \max_k (\lambda_{1,k,2} - \lambda_{1,k,1})$ is chosen to indicate maximum persistence, and used it to compute

$$S = 1 - \frac{\lambda_{1,k_M,2}^2 - \lambda_{1,k_M,1}^2}{3}.$$

Since $0 \leq \lambda_{1,k_M,1} \leq \lambda_{1,k_M,2} \leq \sqrt{3}$, for periodic (nonperiodic) time series, the score is close to zero (one). The R code for implementing Step 1-Step 3 for Case 1 is shown below (the code for other cases is similar):

```
x.ts = ts1; d=15; N=201; T1 = 216;
x.ts <- pracma::movavg(x.ts, 5, type = "s") #step 0
sp.ts <- stats::spline(1:T*2*pi/T, x.ts, n=T1)$y #step 1.1
PC <- plyr::ldply(map(1:N, ~sp.ts[.x:(.x+d-1)])) #step 1.2
X.PC=t(apply(PC,1,FUN=function(x){(x-mean(x))/sqrt(sum((x-mean(x))^2))})) #step2
diag <- ripsDiag(X=x.PC, maxdimension = 1, maxscale = sqrt(3)) #step 3
```

The main contributions of Perea et al. (2015) are the use of extensive simulation studies to show that topological features of time series are largely the same under various non-sinusoidal shapes as well as under differences in amplitude, phase, mean, frequency, or trend. The results may be affected by a differences in noise variances, as well as by the shapes of the noise and signal distributions.

Example 2.5. Using SW1PerS for Periodicity Quantification. We generate white noise ϵ_t with variance $\sigma_\epsilon^2 = 0.64$, and then generate periodic time series signals each of length T ($= 480$), denoted by **ts1**, **ts2**, **ts3**, **ts4**, and shown in the top row of Figure 3.

- Case 1: $x_t = \cos(2\pi t/12) + \epsilon_t$,
- Case 2: $x_t = 0.05t + 10(\cos(2\pi(t - \varphi)/48) + \epsilon_t)$,
- Case 3: $x_t = 10(\cos(2\pi t/12) + \epsilon_t) \exp(-0.01t)$,
- Case 4: $x_t = \epsilon_t$

The series **ts2** differs from **ts1** in frequency, phase, and linear trend, whereas **ts3** only differs from **ts1** in shape and amplitude; **ts4** is the white noise series. Figure 3 shows that the method in Perea et al. (2015) is insensitive to different periodicities. The top row shows the simulated time series under the four cases. The middle row presents a two dimensional view of the point clouds constructed with $d = 15$ and $\tau = 1$, via the first two principal components. The bottom row shows the persistence diagrams, along with the periodicity score S . For periodic (nonperiodic) time series, the score is close to zero (one), so that the scores for Case 1 to Case 3 are close to 0, while Case 4 has a score close to 1.

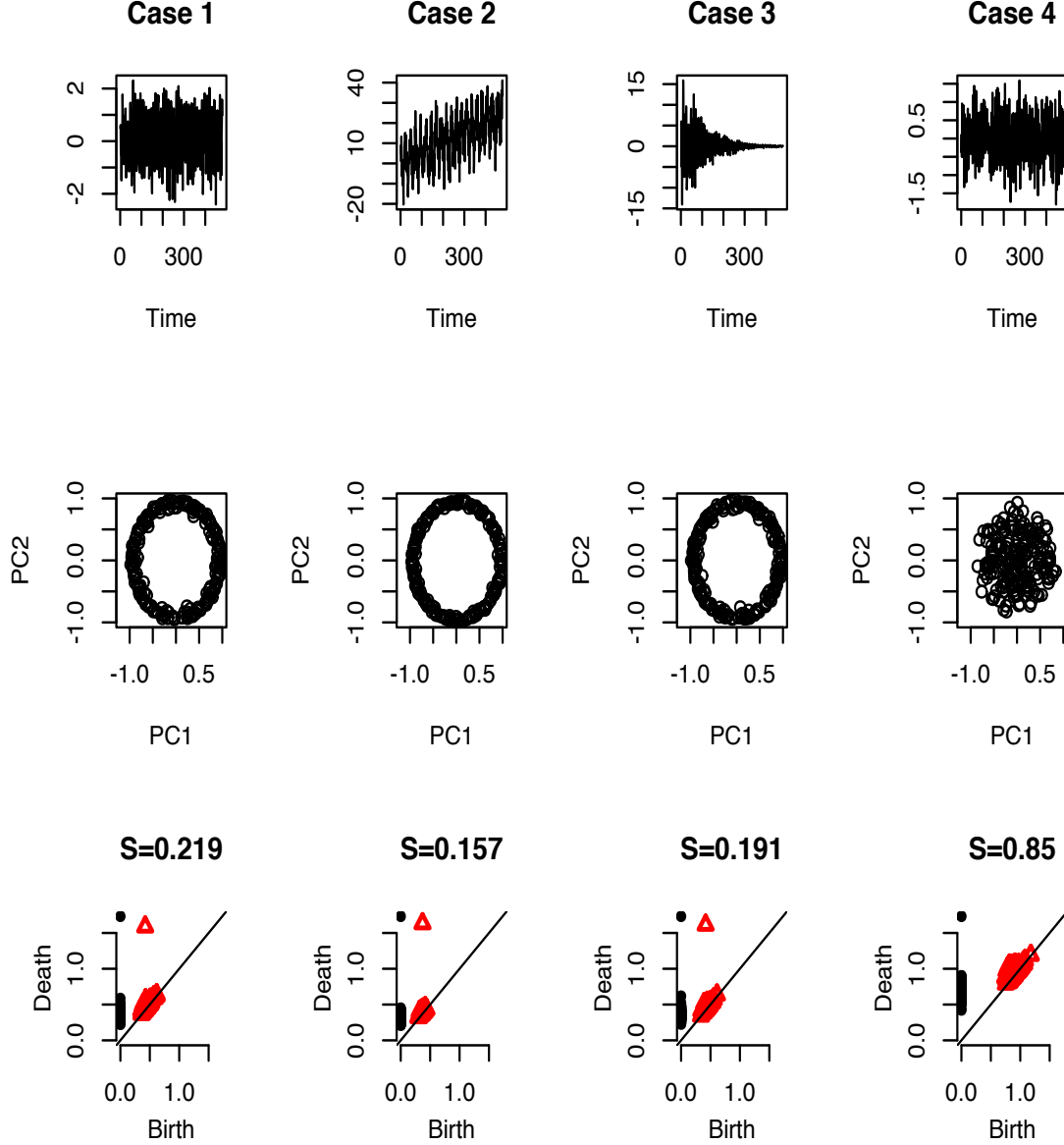


Figure 3: Persistence Diagrams of Periodic Time Series with Different Shapes

3 Persistent Homology Based on Functions

We first give a basic review of TDA based on functions, followed by the use of frequency domain representations of time series as starting points for TDA.

3.1 Function to Persistence Diagram - A Basic Review

When data is in the form of a continuous function $f : \mathcal{R}^d \rightarrow \mathcal{R}$, or can be converted to such a function, TDA using sublevel set filtration is carried out by discretizing the function into grids and then implementing computational homology on the discretized function. Suppose the components

of the function are $\mathbf{z} = (\ell_1\delta, \ell_2\delta, \dots, \ell_d\delta)$, for $\ell_1, \ell_2, \dots, \ell_d = 0, \pm 1, \pm 2, \dots$, where $d > 0$ and $\delta > 0$. The sublevel set of the function is defined as

$$L_\lambda(f) = \{\mathbf{z} : f(\mathbf{z}) \leq \lambda, \mathbf{z} \in \mathcal{R}^d\}, \quad (4)$$

where $0 \leq \lambda \leq \max_{\mathbf{z}} f(\mathbf{z})$. Define a simplex as a set of components in $L_\lambda(f)$ which are “neighbors”, i.e., $\mathbf{z}_1, \mathbf{z}_2 \in L_\lambda(f)$, and $|z_{1,j} - z_{2,j}| \leq \delta, j = 1, 2, \dots, d$.

Recall from Section 2.1 that a simplex with $(\tilde{p} + 1)$ components is called a \tilde{p} -simplex. Since only adjacent points on the grid can be neighbors, the function $f(\mathbf{z})$ can admit at most a d -simplex, so that $\tilde{p} \leq d - 1$ (Edelsbrunner and Harer, 2010). When $d = 1$, there are only connected components, whose births and deaths are given by $\tilde{\tau}_{0,k} = (\lambda_{0,k,1}, \lambda_{0,k,2}), k = 1, 2, \dots, k_0$. The computations in the steps that are summarized below are done using the R function `gridDiag` (Fasy et al., 2014a).

Step 1. Assume a filtration parameter starting at $\lambda = \min_{\mathbf{z}} f(\mathbf{z})$ and let $L_\lambda(f) = \{\mathbf{z} : f(\mathbf{z}) = \min_{\mathbf{z}} f(\mathbf{z})\}$.

Step 2. Construct topological features for increasing values of λ . For each λ , simplicial complexes can be constructed from the sublevel set $L_\lambda(f)$, and $\tilde{\alpha}_{\tilde{p},k}$ can be computed using computation homology, where $0 \leq \tilde{p} \leq d - 1$. If an elder topology $\tilde{\alpha}_{\tilde{p},k_1}$ and a younger topology $\tilde{\alpha}_{\tilde{p},k_2}$ merge into a single $\tilde{\alpha}_{\tilde{p},k}$ at some λ value, then $\tilde{\alpha}_{\tilde{p},k_1}$ becomes $\tilde{\alpha}_{\tilde{p},k}$ and $\tilde{\alpha}_{\tilde{p},k_2}$ dies at λ , using the *Elder Rule* (Edelsbrunner and Harer, 2010).

Step 3. The persistence diagram is the output of the set of points representing birth-death of homology groups $\{\tilde{\tau}_{\tilde{p},k} = (\lambda_{\tilde{p},k,1}, \lambda_{\tilde{p},k,2}) : \tilde{p} = 0, 1, \dots; k = 1, 2, \dots, k_{\tilde{p}}\}$.

Example 3.1. Discretized Function to Persistence Diagram. We present an example of using TDA on a one-dimensional discretized real function generated using the code chunk below, where `funval` contains values of the function taken over a grid:

```
funval = c(1, 0.5, 1, 1.5, 0.5, 0, 1, 1, 0.5, 1)
(pers.diag.4 <- gridDiag(FUNvalues = funval, sublevel = TRUE) )
$diagram
  dimension Birth Death
[1,]        0    0.0   1.5
[2,]        0    0.5   1.5
[3,]        0    0.5   1.0
```

The persistence diagram contains three connected components born at $\lambda = 0$ and $\lambda = 0.5$, corresponding to the function having three local minima at these two distinct λ values. In Figure 4(a), a connected component emerges when $\lambda = 0$ and is marked as a blue dot (it is the earliest/oldest connected component).

```
plot(funval, x=1:10, type = "l", yaxt='n', ylim = c(0,2), cex.axis=1.4, xlab = "z",
     ylab= "y", cex.lab= 1.3, cex=1.2, lwd=1.5, lty=1, pch=1, bty='n', main="(a)")
points(x=6, y=0, pch=16, col="blue", type = "p")
ticks<-c(0, 0.5, 1, 1.5, 2); axis(2,at=ticks,labels=ticks)
abline(a=0, b=0, lty=2, lwd=1,pch=1)
```

The vertical dashed line in Figure 4(b) corresponds to the birth time $\lambda = 0$, while the horizontal dashed line tracks the current filtration parameter λ . There is no point on the birth/death plot yet, as no connected components have died at $\lambda = 0$.

```
plot(rep(0,11),x=0:10/5,type = "l",cex.axis=1.4,xaxt='n',xlab= "birth",yaxt='n',
ylim=c(0,2),ylab="death",cex.lab=1.3,cex=1.2,lwd=1.5,lty=2,pch=1,bty='n',main="(b)")
axis(1,at=ticks,labels=ticks); axis(2,at=ticks,labels=ticks); abline(v=0, lty=2)
```

Figure 4(c) corresponds to $\lambda = 0.5$. There are two more connected components indicated by the blue dots at (2, 0.5) and (9, 0.5). The blue dot (6, 0) in the middle with a blue line connecting it to the dot (5, 0.5) indicates that the oldest connected component enlarges.

```
plot(funval, x=1:10, type = "l",yaxt='n',ylim = c(0,2),cex.axis=1.4,xlab = "z",
      ylab = "y",cex.lab= 1.3, cex=1.2,lwd=1.5, lty=1, pch=1, bty='n', main = "(c)")
axis(2,at=ticks,labels=ticks); abline(a=0.5, b=0, lty=2, lwd=1,pch=1)
points(x=6, y=0, pch=16, col="green4", type = "p",xlab = "z", ylab= "y")
points(x=c(2,5,9), y=rep(0.5,3), pch=16, col="blue", type = "p",xlab="z",ylab="y")
segments(x0=6, y0=0, x1=5, y1=0.5, lty = 1, pch=1, lwd=2.5, col = "blue")
```

Figure 4(d) has one more vertical dashed line, which gives the birth time for the other two new connected components. There is no connected component dead yet, and so no points are shown on the second plot either. The code chunks for plotting these are similar and are not shown due to space limitations. When $\lambda = 1$, in Figure 4(e), all components enlarge and one newer component is killed by the elder one because they are merged. There is a black dot at (0.5, 1) in Figure 4(f), which indicates the newer connected component that is born at $\lambda = 0.5$ and is dead at $\lambda = 1$. When $\lambda = 1.5$ reaching the maximum of the function in Figure 4(g), the last component is killed. The black dot at the location (0, 1.5) in Figure 4(h) is for the last component. The other black dots corresponding to (0.5, 1.5) and (0.5, 1) show the birth and death of other connected components.

Example 3.2. Morse Function to Persistence Diagram. An alternate technique for persistent homology which is robust to noisy point cloud data uses the R function `gridDiag` (Chazal et al., 2018). As mentioned in the introduction, a point cloud is assumed as a sample from an underlying manifold. To learn the topology of the manifold, `gridDiag` enables us to construct a Morse function such as the distance-to-measure (DTM) function from the point cloud using the sublevel set filtration. Suppose the point cloud is $\{\mathbf{x}_i \in \mathcal{R}^d, i = 1, 2, \dots, N\}$. Represent the DTM function from $\mathcal{R}^d \rightarrow \mathcal{R}$ as

$$f^{(DTM)}(\mathbf{x}) = \sqrt{\sum_{\mathbf{x}_i \in \mathcal{N}_k(\mathbf{x})} \|\mathbf{x}_i - \mathbf{x}\|^2 / k},$$

where $k = \lceil mN \rceil$ (m is the parameter `m0`) and $\mathcal{N}_k(\mathbf{x})$ is the set containing the k nearest neighbors of \mathbf{x} in the point cloud.

A higher value of $f^{(DTM)}(\mathbf{x})$ means that \mathbf{x} is further away from most of the points. DTM is also robust to outliers (Chazal et al., 2018). We illustrate using the same point cloud data from Example 2.1:

```
m0=0.05; by <- 0.065; Xlim <- range(PC[,1]); Ylim <- range(PC[,2])
(pers.diag.5 = gridDiag(X=PC, FUN=dtm, lim=cbind(Xlim, Ylim), by=by, m0=m0) )
$diagram
      dimension      Birth      Death
```

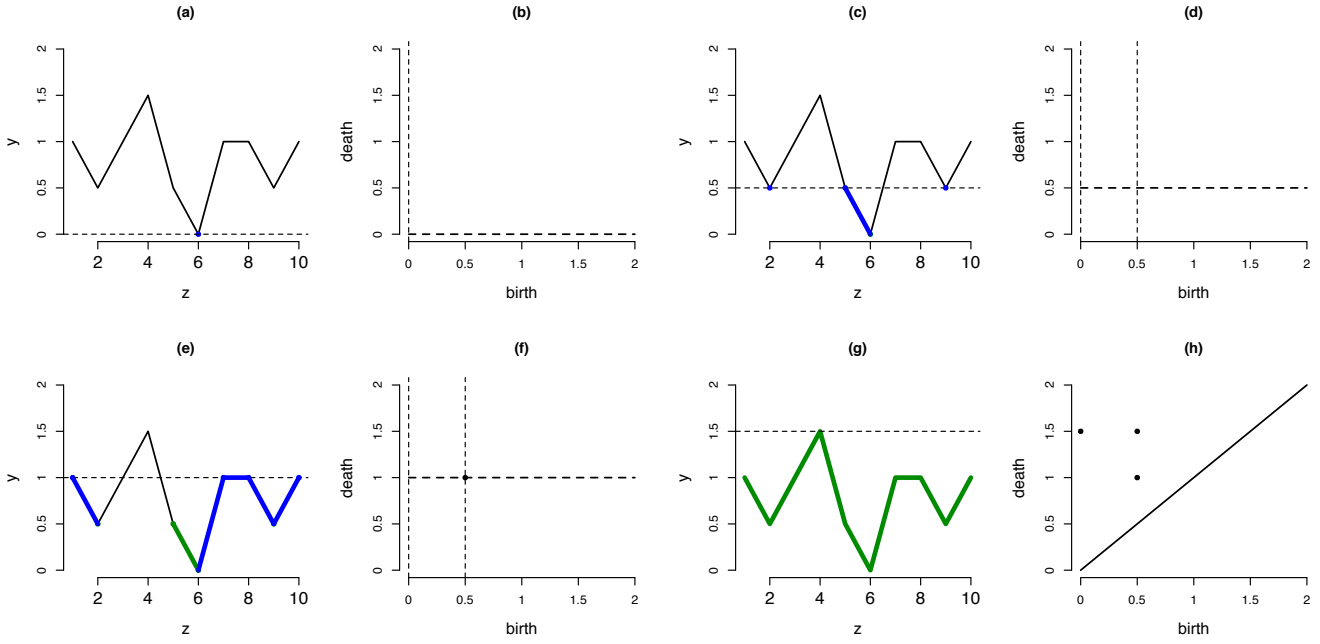


Figure 4: Construction of a persistence diagram corresponding to a one-dimensional continuous real function. (a) is the function and (h) is the persistence diagram. (c), (e) and (g) show the sublevel set filtration procedure, while (b), (d) and (f) are the intermediate steps for constructing the persistence diagram.

```
[1,]      0 0.02414385 0.96912324
[2,]      0 0.02623549 0.15885911
[3,]      0 0.03489488 0.15662338
...
[20,]     0 0.14527336 0.15092905
[21,]     1 0.20234735 0.96912324
```

Here, `lim` specifies the range of the point cloud in different dimensions, `by` is the step size for increasing λ , and `m0` (which lies in $(0, 1)$ with 0.05 as the default value) is the smoothing parameter of the DTM method. The code for constructing Figure 5 is shown below:

```
par(mfrow = c(1,2))
Xseq <- seq(from = Xlim[1], to = Xlim[2], by = by)
Yseq <- seq(from = Ylim[1], to = Ylim[2], by = by)
Grid <- expand.grid(Xseq, Yseq); DTM = dtm(X = PC, Grid = Grid, m0 = m0)
persp(x = Xseq, y = Yseq, z = matrix(DTM, nrow = length(Xseq), ncol = length(Yseq)),
      xlab = "", ylab = "", zlab = "", theta = -20, phi = 35, scale = FALSE,
      expand = 2, col = "red", border = NA, ltheta = 50, shade = 0.5, main = "(a)")
plot(pers.diag.5[["diagram"]], main = "(b)")
```

Figure 5(a) shows the DTM function of the point cloud. The function has a peak in the middle of the plot, surrounded by a rough circle of local minima (since the original point cloud is from the unit circle and DTM is a smoothed distance function) Figure 5(b) shows the persistence diagram. The dots in the persistence diagram are the birth-death points of 0-th homology groups and the

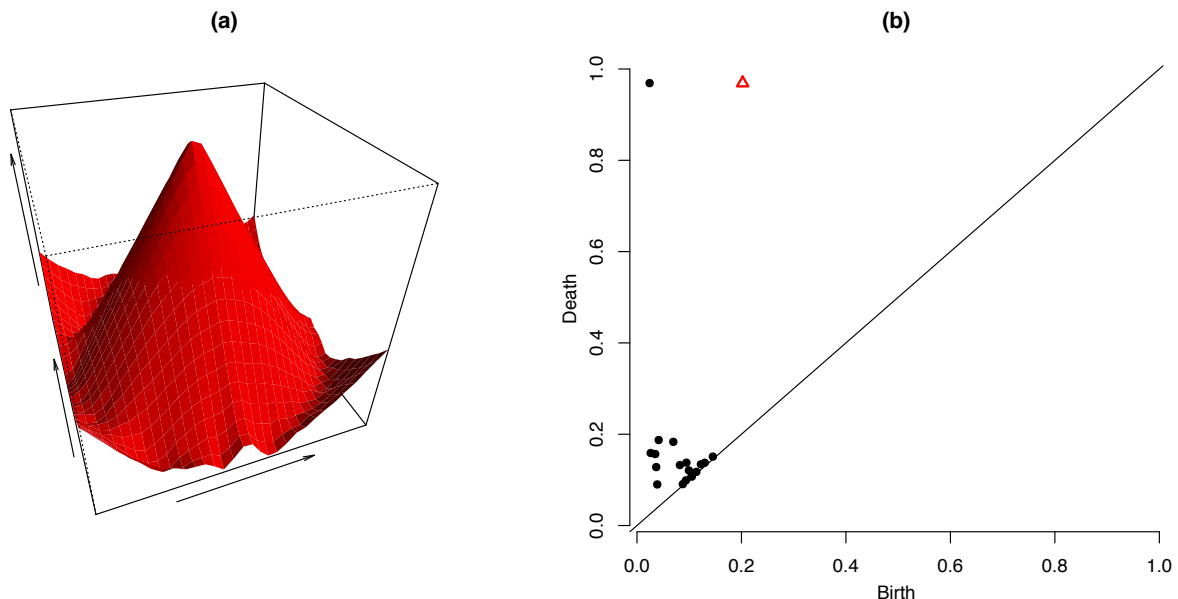


Figure 5: Distance-to-Measure (DTM) function and the persistence diagram.

triangle denotes the birth-death point of 1-th homology group. There are only 20 birth-death points of 0-th homology groups instead of 60 as we saw in Example 2.1, because the DTM smooths the distance function, as mentioned earlier, resulting in a different output from those in Examples 2.1 and 2.2. The point at (0.2, 0.97) indicates a big circle in the data (representing the topology of the underlying unit circle).

3.2 TDA of Time Series via Frequency Domain Functions

Section 3.2.1 discusses TDA starting from variations of the Fourier transform for continuous-valued time series, while Section 3.2.2 shows how to build persistence diagrams based on Walsh Fourier transforms for categorical time series.

3.2.1 Discrete Fourier Transforms to Persistence Diagrams

In this section, we look at topological properties of time series through their frequency domain representations such as second-order spectra. We construct a persistence diagram using sublevel set filtration on the smoothed tapered estimate of the second-order spectrum of the time series $\{x_t, t = 1, \dots, T\}$. The modified DFT (Stoffer, 1991) and corresponding periodogram with tapering are defined as

$$\begin{aligned} d_h(\omega_j) &= T^{-1/2} \sum_{t=1}^T h_t x_t e^{-2\pi i \omega_j t} \text{ and} \\ I_h(\omega_j) &= |d_h(\omega_j)|^2, \end{aligned}$$

for $t = 1, 2, \dots, T$, where h_t is a taper function. In Example 3.3, we show the use of the R function `gridDiag` to construct the persistence diagram starting from a smoothed version of $I_h(\omega_j)$, using the Daniel window for smoothing.

Example 3.3. Smoothed Tapered Second-order Spectrum to Persistence Diagram. We use the R function `gridDiag` to construct the persistence diagrams for the same three periodic time series signals shown in Example 2.4. For Case 1, `spc.t1` denotes the smoothed tapered periodogram (taper = 0.1 as default in the R function `spec.pgram`, and smoothing via the modified Daniel window (0.25, 0.5, 0.25)) of the time series `ts1`.

```
x.1 = 1:length(ts1); ts1 = lm(ts1~x.1)$residuals; ts1 = ts1/sd(ts1) #Step 1
spc.t1=spec.pgram(ts1, kernel=kernel("modified.daniell", c(1)), plot = F)#Step 2
#Step 3
PD.t1=gridDiag(FUNvalues=spc.t1$spec,location = FALSE, sublevel=TRUE)$diagram
```

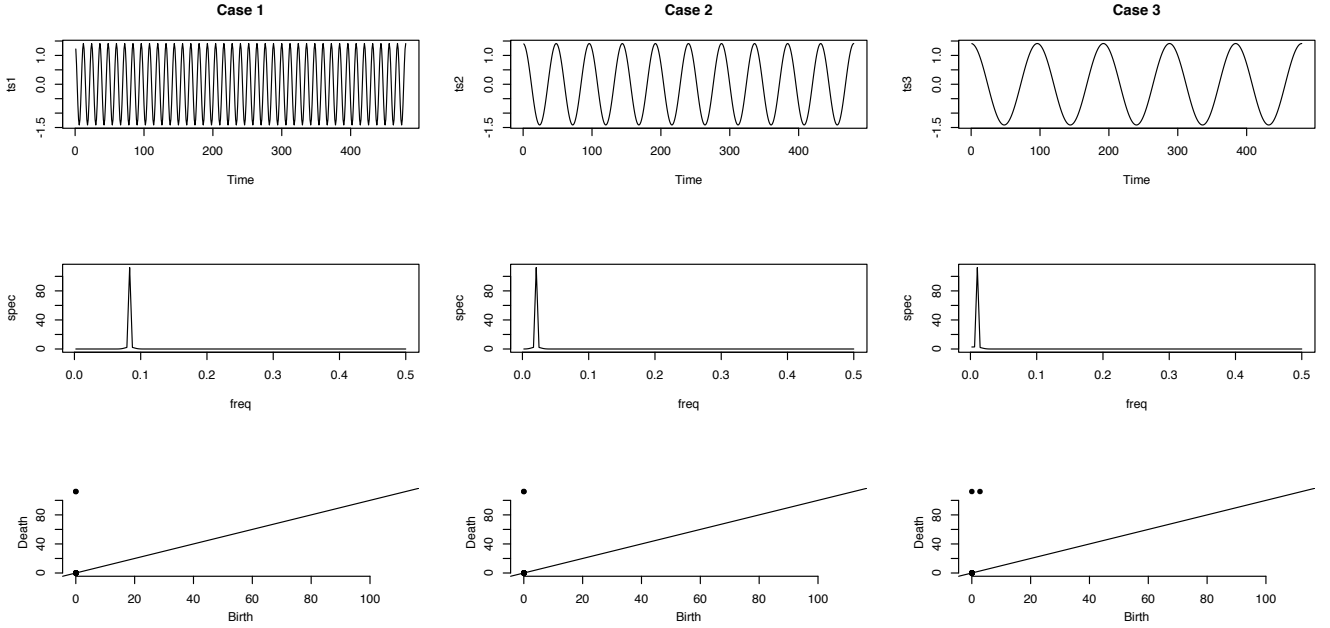


Figure 6: Persistence diagrams using second-order spectrum.

In Figure 6, the top row shows the time series signals, the middle row shows the second-order spectra and the bottom row shows the persistence diagrams. The peaks in the spectra occur at different frequencies and correspond to the contributions at these frequencies to the total variance of x_t . The three persistence diagrams in the bottom row are similar to each other, since this method is insensitive to differences in the periodicity of the time series.

In contrast with the persistence diagrams in Figure 3 for the same signals, we no longer see red triangles for 1-th homology groups, indicating that there is a difference between constructing persistence diagrams from point clouds versus the spectrum.

We compute the bottleneck distances between the three persistence diagrams. The distance between Case 1 and Case 2 is 0.01, which is much smaller than the distance between Case 1 and Case 3 which is 2.77, or the one between Case 2 and Case 3 which is 2.76. The code for computing the distances is similar to the one shown in Example 2.4.

3.2.2 Walsh-Fourier Transforms to Persistence Diagrams

Stoffer (1991) suggested that Walsh spectral analysis is suited to the analysis of discrete-valued and categorical-valued time series, and of time series that contain sharp discontinuities. The fast

Walsh-Fourier Transform construction uses the method of Shanks (1969) to decompose a time series $\{x_t, t = 1, \dots, T\}$ into a sequence of Walsh functions, each representing a distinctive binary sequence pattern. If the time series length T is not a power of 2, let T_2 denote the next power of 2. For example, if $T = 1440$, then $T_2 = 2^{11} = 2048$. We use zero-padding to obtain a time series of length T_2 by setting $x_{T+1}, x_{T+2}, \dots, x_{T_2} = 0$.

For $j = 0, \dots, T_2 - 1$, let $\lambda_j = j/T_2$ denote the j th sequence. Let $W(t, j)$ denote the t -th Walsh function value in sequence λ_j . Walsh functions are iteratively generated as follows (Shanks, 1969):

$$\begin{aligned} W(0, j) &= 1, j = 0, 1, \dots, T_2 - 1, \\ W(1, j) &= \begin{cases} 1 & j = 0, 1, \dots, (T_2)/2 - 1 \\ -1 & j = (T_2)/2, (T_2)/2 + 1, \dots, T_2 - 1 \end{cases} \\ W(t, j) &= W([t/2], 2j) \times W(t - 2[t/2], j), \\ t &= 2, \dots, T_2 - 1, \quad j = 0, 1, \dots, T_2 - 1, \end{aligned} \tag{5}$$

where $[a]$ denotes the integer part of a . For more details on Walsh functions, refer to Stoffer (1991). The Walsh-Fourier Transform (WFT) of the time series is computed as

$$d_T(\lambda_j) = \frac{1}{\sqrt{T_2}} \sum_{t=1}^{T_2} x_t W(t, j), \quad 0 \leq j \leq T_2 - 1. \tag{6}$$

The computational complexity is $O(T \log(T))$ (Shanks, 1969). In Example 3.4, we illustrate the construction of a persistence diagram for a categorical time series with two levels.

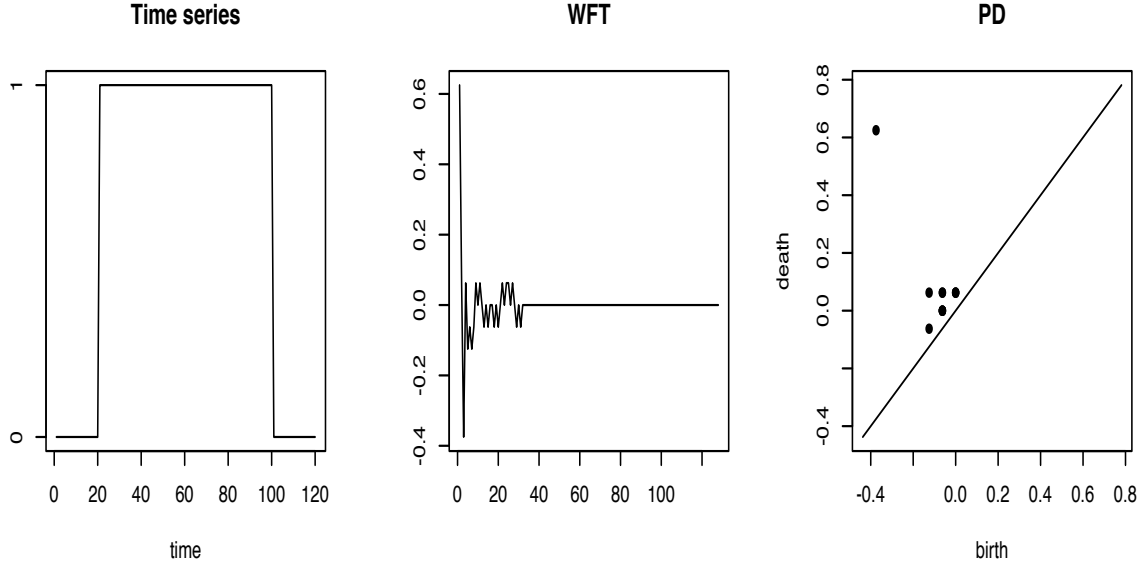


Figure 7: Persistence diagrams using Walsh-Fourier Transforms.

Example 3.4. Walsh-Fourier Transform to Persistence Diagram. In Figure 7, a simulated categorical time series of length $T = 120$ with two levels, 0 or 1, is shown in the first column. Level 1 only occurs in the period between $t = 21$ and $t = 100$. The middle column shows the WFT of

the time series, while the third column shows its persistence diagram. There is one point in the diagram away from the diagonal line, which is a significant birth-death point of the 0-th homology groups.

The R code for simulating the time series and converting the WFT into a persistence diagram is shown below.

```
x.ts = c(rep(0, 20), rep(1, 80), rep(0, 20))
# create WFT using C++ code
x.diag=gridDiag(FUNvalues = x.WFTs, location = FALSE, sublevel = TRUE)$diagram
```

4 Feature Construction Using TDA

Unlike a vector space, the space of persistence diagrams is not easy to work with. For instance, a set of persistence diagrams may not have a unique mean (Mileyko et al., 2011). The bottleneck or Wasserstein distances are also more complicated than the Euclidean distance in practice. This section discusses an alternative.

4.1 Persistence Landscapes - A Basic Review

Bubenik (2015) introduced persistence landscapes as useful statistical summaries which build topological features and are easy to combine with tools from statistics and machine learning. This section reviews persistence landscapes while Section 4.2 describes their construction for time series, with examples.

The ν th order persistence landscape of \tilde{p} -th homology groups is defined as

$$PL_{\tilde{p},\nu}(\ell) = \{\min(\ell - \lambda_{\tilde{p},k,1}, \lambda_{\tilde{p},k,2} - \ell)_+ : k = 1, 2, \dots\}_{(\nu)} \quad (7)$$

where $\lambda_{\tilde{p},k,1}$ and $\lambda_{\tilde{p},k,2}$ were introduced under Step 3 of Section 2.1, $\ell \in \mathcal{R}$, $\min(a, b)_+$ denotes the smaller value if both a and b are positive, or zero if neither value is positive, and $\{A\}_{(\nu)}$ is the ν -th order statistic of the set A . Bubenik (2015) proved that a set of persistence landscapes admits a unique mean and preserves statistical stability of the data distribution. If we assume that the observed data \mathbf{X} is a random draw from an underlying space \mathcal{X} equipped with a probability measure, and assume multiple copies $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_N$, then the mean of their persistence landscapes $\overline{PL}_{\tilde{p},\nu}^{(N)}(\ell) = \sum_{i=1}^N PL_{\tilde{p},\nu}^{(i)}(\ell)/N$ would converge almost surely (as $N \rightarrow \infty$) to the expectation of the persistence landscapes $E(PL_{\tilde{p},\nu}^{\mathcal{X}}(\ell))$ if and only if

$$E||PL_{\tilde{p},\nu}^{\mathcal{X}}(\ell)|| = \int_{\ell} |PL_{\tilde{p},\nu}^{\mathcal{X}}(\ell)| d\ell < \infty,$$

i.e., $\int_{\ell} |PL_{\tilde{p},\nu}^{\mathcal{X}}(\ell)| d\ell < \infty$ for all \tilde{p} . This means that the set of persistence landscapes from the observed data is a good representative for the underlying distribution of true persistence landscapes and it is possible to do statistical inference for these using the sample persistence landscapes. Another important statistical property is the stability in terms of using persistence landscapes versus using persistence diagrams. Suppose there are two persistence diagrams $\tilde{\Omega}_1$ and $\tilde{\Omega}_2$ together with the corresponding persistence landscapes $PL_{\tilde{p},\nu}^{(1)}(\ell)$ and $PL_{\tilde{p},\nu}^{(2)}(\ell)$. Then, $||PL_{\tilde{p},\nu}^{(1)}(\ell) - PL_{\tilde{p},\nu}^{(2)}(\ell)||_q = [\int_{\nu} |PL_{\tilde{p}}^{(1)} - PL_{\tilde{p}}^{(2)}|^q]^{1/q}$ is no larger than a function of the q -Wasserstein distance $\mathbf{W}_{q,\tilde{p}}(\tilde{\Omega}_1, \tilde{\Omega}_2)$. Intuitively, it means that using persistence landscapes could preserve differences in the persistence diagrams.

The following steps describe the construction of persistence landscapes of \tilde{p} -th homology groups in increasing order starting from the first-order landscape.

Step 1. Extract the 1-th homology groups $\tilde{\tau}_{\tilde{p},k}$ ($k = 1, 2, \dots, k_{\tilde{p}}$) from the persistence diagram $\tilde{\Omega}$ and create a set of computation grids, $\ell = M_1, M_1 + \delta, M_1 + 2\delta, \dots, M_2$, where the lower and upper bounds M_1 and M_2 are usually set as $\min_k \lambda_{\tilde{p},k,1}$ and $\max_k \lambda_{\tilde{p},k,2}$ respectively. Here, δ depends on the degree of resolutions. The R function `landscape` uses 500 grid points by default so that $\delta = (M_2 - M_1)/500$.

Step 2. Use each $\tilde{\tau}_{\tilde{p},k}$ to compute $PL_{\tilde{p},k}(\ell) = \min(\ell - \lambda_{\tilde{p},k,1}, \lambda_{\tilde{p},k,2} - \ell)_+$ for all values of ℓ .

Step 3. Fixing ℓ , sort $pl_{\tilde{p},k}(\ell)$ in decreasing order, calling them $pl_{\tilde{p}}^{(1)}(\ell), pl_{\tilde{p}}^{(2)}(\ell), \dots, pl_{\tilde{p}}^{(k_{\tilde{p}})}(\ell)$.

Step 4. Output the ν th order persistence landscape of the \tilde{p} -th homology groups, $PL_{\tilde{p},\nu}(\ell) = (pl_{\tilde{p}}^{(\nu)}(\ell))_+$, where $\nu = 1, 2, \dots$ and set $pl_{\tilde{p}}^{(\nu)}(\ell) = 0$ when $\nu > k_{\tilde{p}}$.

Step 3 primarily determines the computational cost of constructing persistence landscapes. Higher order persistence landscapes require more values sorted for each ℓ , which could be costly when $k_{\tilde{p}}$ (number of \tilde{p} -th homology groups) is large.

Example 4.1. Persistence Landscapes from Persistence Diagrams. The R function `landscape` can be used to compute persistence landscapes. We illustrate on the the persistence diagram from Example 2.3.

```
Diag = pers.diag.3$diagram; Land <- c(); k=1; threshold=1
while(threshold>0){
  Land <- cbind(Land, landscape(Diag = Diag, dimension = 0, KK = k,
    tseq = seq(min(Diag[,2:3]), max(Diag[,2:3]), length=500)))
  threshold = sum(abs(Land[, ncol(Land)]))
}
```

	[,1]	[,2]	[,3]	[,4]
[1,]	0.0000000000	0	0	0
[2,]	0.003006012	0	0	0
[3,]	0.006012024	0	0	0
.....				
[499,]	0.003006012	0.003006012	0	0
[500,]	0.0000000000	0.0000000000	0	0

In the code above, the function `landscape` takes several arguments. When `dimension=0`, it takes 0-th homology groups of the persistence diagram `pers.diag.3` to compute landscape functions. The `KK` argument specifies the order of landscapes to be computed. The `tseq` argument specifies the range of the landscape functions. It uses a while-loop to compute persistence landscapes of all orders (here, four) over 500 grids ranging from the minimum of the birth time to the maximum of the death time.

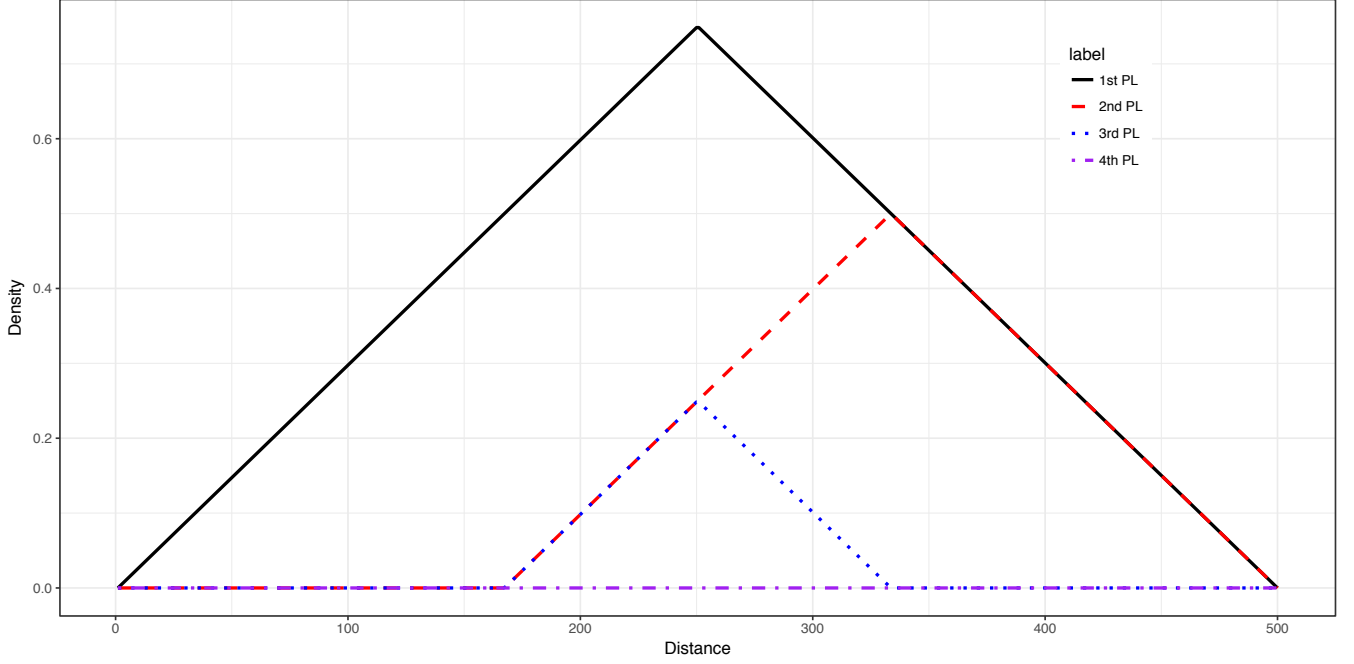


Figure 8: Persistence landscapes of the persistence diagram `pers.diag.3`.

Persistence landscapes of any order can be used as features. Lower order persistence landscapes contain information about important topological features than higher order persistence landscapes which are closer to zero and handle topological noise. Therefore, selecting the order of the persistence landscapes to serve as features requires a delicate balance between missing important signals and introducing too much noise.

4.2 TDA of Time Series via Persistence Landscapes

TDA on functions can be used to construct feature representations for time series analysis, and persistence landscapes are useful as topological representations for similarity/dissimilarity analysis on time series. In the literature, different representations of time series have been used, such as the weighted Fourier transform in Wang et al. (2018) or the Walsh-Fourier transform in Chen et al. (2019). We describe these situations in the following sections.

4.2.1 Persistence Landscapes for Continuous Time Series

Wang et al. (2018) proposed TDA to measure structural changes in electroencephalogram (EEG) time series. They first constructed Fourier transforms of the time series, then applied an exponential weighting scheme on the Fourier transforms to focus on the more important low frequency components of EEG. They further smoothed the weighted Fourier transform in order to make it a Morse function (Palais, 1963).

The smoothed weighted Fourier series of a time series $\{x_t, t = 1, \dots, T\}$ has the form

$$\hat{\mu}_{T_u}^k(t) = \sum_{j \in \mathbf{I}_1} e^{-(2j\pi/T)^2 \sigma} a_j \cos(2j\pi t/T) + \sum_{j \in \mathbf{I}_2} e^{-(2j\pi/T)^2 \sigma} b_j \sin(2j\pi t/T),$$

where $\mathbf{I}_1 = \{j = 0, 1, 2, \dots, k : |a_j| > T_u\}$, $\mathbf{I}_2 = \{j = 1, 2, \dots, k : |b_j| > T_u\}$, $T_u = s\sqrt{2 \log(n)}$, $a_j = \frac{2}{T} \sum_{t=1}^T x_t \cos(2j\pi t/T)$, $b_j = \frac{2}{T} \sum_{t=1}^T x_t \sin(2j\pi t/T)$ and $a_0 = \sum_{t=1}^T x_t/T$. Here, k is the degree

deciding the highest frequency $[k/T]$ to be included in the representation (for $T = 500$, they used $k = 99$), n is the number of data points in each phase and s is the median of the absolute deviation (MAD) of the Fourier coefficients:

$$\begin{aligned} a^{(m)} &= \text{median}\{|a_i|, i = 1, 2, \dots, k\} \\ b^{(m)} &= \text{median}\{|b_i|, i = 1, 2, \dots, k\} \\ s &= \text{median}\{|a_i - a^{(m)}|, |b_j - b^{(m)}|, i, j = 1, 2, \dots, k\} \end{aligned} \quad (8)$$

Using this finite sum of weighted sinusoidal functions, they argued that $\hat{\mu}_{T_u}^k(t)$ becomes a Morse function. They used this Morse function representation to construct persistence landscapes of all orders as features to detect possible structural changes. A main contribution of this paper is to show via simulation studies that the proposed TDA framework is robust to topology-preserving transformations such as translation and amplitude and frequency scaling, while being sensitive to topology-destroying transformations. They argued the topological change happens only if there is a structural change in the time series.

4.2.2 Persistence Landscapes for Categorical Time Series

Chen et al. (2019) described TDA of categorical time series via their Walsh-Fourier transforms (which are not Morse functions). They constructed first order persistence landscapes based on Walsh-Fourier transforms of categorical time series, which they then used as features for clustering. They applied this analysis to a large travel-activity data set, carrying out computations in parallel. They showed that construction of the first order persistence landscape only involves a linear transformation of the Walsh Fourier transform.

Given a sequence of WFT $d_T(n, \lambda_j), j = 0, 1, \dots, T_2 - 1$ of the time series $x_{n,t}, n = 1, 2, \dots, N$, denote the minimum and maximum of the WFT values of the time series $x_{n,t}$ by

$$d_{n,\min} = \min_j d_T(n, \lambda_j) \text{ and } d_{n,\max} = \max_j d_T(n, \lambda_j).$$

Let

$$D_{\min} = \min_n d_{n,\min} \text{ and } D_{\max} = \max_n d_{n,\max}$$

denote the minimum and maximum values of the WFTs across all N time series. The first-order persistence landscape of $x_{n,t}$ is obtained for $\ell = 1, 2, \dots, L$ as

$$PL(n, \ell) = \min(V_1(n, \ell), V_2(n, \ell))_+ \quad (9)$$

where

$$\begin{aligned} V_1(n, \ell) &= D_{\min} + \frac{(\ell - 1)(D_{\max} - D_{\min})}{L - 1} - d_{n,\min}, \\ V_2(n, \ell) &= d_{n,\max} - D_{\min} - \frac{(\ell - 1)(D_{\max} - D_{\min})}{L - 1}, \end{aligned}$$

and $(a)_+$ denotes the positive part of a real number a . For $\ell = 1, 2, \dots, L$ and $n = 1, \dots, N$, the $PL(n, \ell)$ are piecewise linear functions that constitute features constructed for each of the N time series and useful for clustering. Our C++ code is available here: <https://github.com/bluemarlon/TDA-of-K-means-on-1st-PL>.

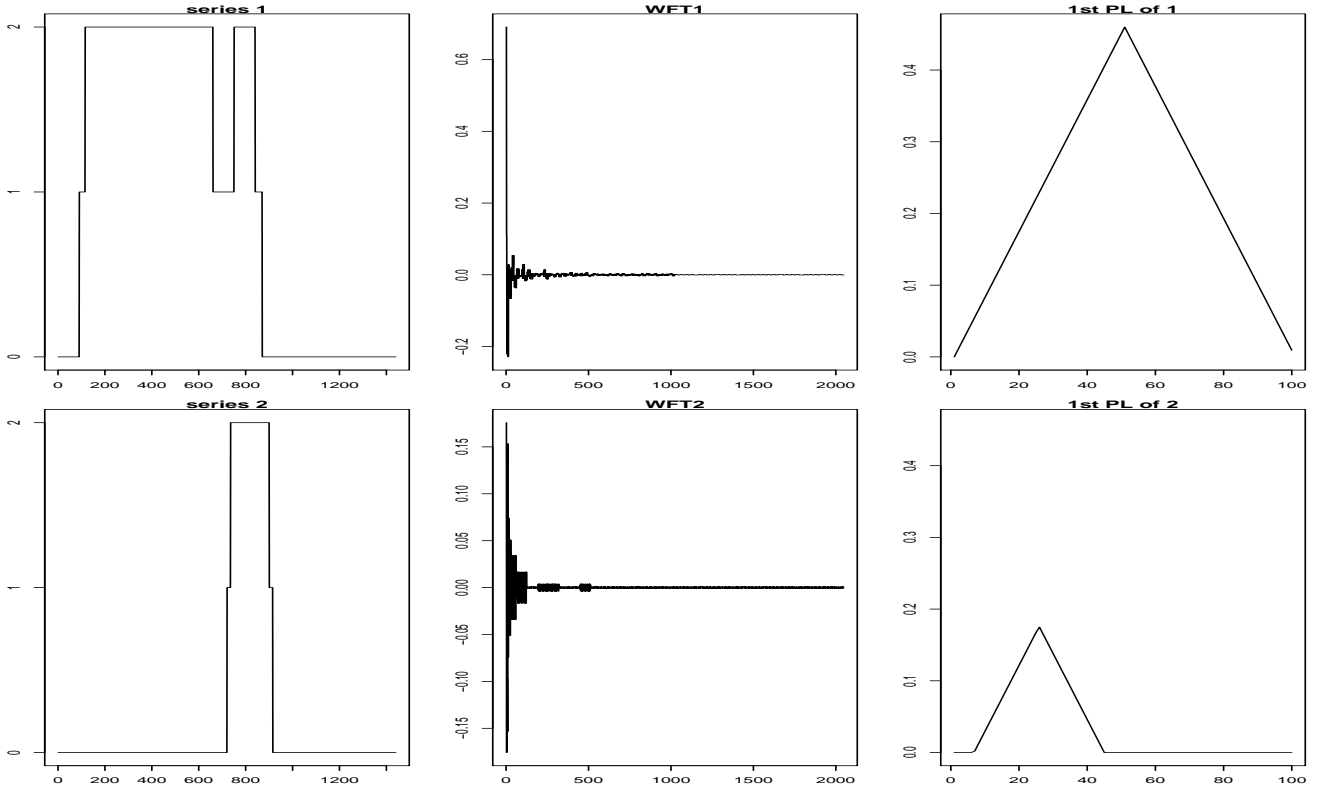


Figure 9: Persistence diagrams using Walsh-Fourier Transforms.

In Figure 9, the first column shows categorical time series on activity-travel behavior of two randomly chosen adults from the National Household Travel Survey (Chen et al., 2019). The length of each time series is $T = 1440$, corresponding to the number of minutes in a day. The response has three levels for each adult: 0 for staying at home, 1 for travel and 2 for being out of the home. The middle column shows the WFT of the time series, each of length $T_2 = 2048$. The last column shows first order persistence landscapes which are quite distinct for the two series.

After the first order persistence landscapes are constructed, Chen et al. (2019) used a divide and combine K-means approach for clustering a large number of subjects and identified three distinct temporal patterns among them. The main contribution of this paper was to implement clustering of a large set of activity-travel time series through TDA of non-Morse functions.

4.3 Other TDA Based Approaches

Numerical summaries other than persistence landscapes have been used for clustering (Berwald et al., 2013; Pereira and de Mello, 2015; Seversky et al., 2016), classification (Umeda, 2017), and break detection (Gidea, 2017; Gidea and Katz, 2018) of time series. We give a brief review in the following sections.

4.3.1 Clustering

TDA based feature construction may be used with classical clustering methods, like K-means clustering for time series. For instance, Pereira and de Mello (2015) considered the model analyzed by Costantino et al. (1995) for evaluating two regimes of adult *Tribolium* flour population growth

(numerical measures) under stable equilibrium and aperiodic oscillations. They simulated a total of 400 time series from the model for a period of 240 weeks (2 weeks per unit), consisting of 200 time series for the stable equilibrium regime and the other 200 for the aperiodic oscillations. A brief description of their approach follows.

Step 1. They pre-processed the data using the Empirical Mode Decomposition (EMD) (Huang et al., 1998) to denoise the data.

Step 2. They constructed point clouds using Takens' embedding with $d = 2$ and $\tau = 3$, taking the time series $\{x_t, t = 1, 2, \dots, T\}$ to $\{\mathbf{v}_i = (x_i, x_{i+3}) | i = 1, 2, \dots, T - 3\}$. The point cloud, therefore, contains a total of $N = T - 3$ number of points.

Step 3. They used the Witness complex (De Silva and Carlsson, 2004) for doing persistent homology. Witness complex is essentially a different simplicial complex, which is most useful when dealing with large data sets.

Step 4. After acquiring the persistence diagram $\tilde{\Omega} = \{\tilde{\tau}_{\tilde{p},k} | k = 1, 2, \dots, k_{\tilde{p}}, \tilde{p} = 0, 1\}$ ($\tilde{\tau}_{\tilde{p},k} = (\lambda_{\tilde{p},k,1}, \lambda_{\tilde{p},k,2})$), they constructed the following set of features:

- (i) The number of points on each dimension \tilde{p} , i.e., $(k_0, k_1, k_2, \dots, k_{\tilde{p}})$;
- (ii) The maximum lifetime of each \tilde{p} , i.e., $\max_k(\lambda_{\tilde{p},k,2} - \lambda_{\tilde{p},k,1})$;
- (iii) The number of relevant points for each \tilde{p} , namely $\#\{\tilde{\tau}_{\tilde{p},k} | \lambda_{\tilde{p},k,2} - \lambda_{\tilde{p},k,1} \geq 0.5 \max_k(\lambda_{\tilde{p},k,2} - \lambda_{\tilde{p},k,1})\}$, where $\#$ denotes the cardinality of the set;
- (iv) The average lifetime of all homology groups on each \tilde{p} , $\sum_{k=1}^{k_{\tilde{p}}} (\lambda_{\tilde{p},k,2} - \lambda_{\tilde{p},k,1}) / k_{\tilde{p}}$;
- (v) The sum of the lifetimes of all homology groups on each \tilde{p} , $\sum_{k=1}^{k_{\tilde{p}}} (\lambda_{\tilde{p},k,2} - \lambda_{\tilde{p},k,1})$.

Step 5. Apply all features from Step 4 to do K-means clustering.

The implementation was done in **Java**. They compared results with true labels using different metrics from the confusion matrix, and showed that their method achieved a high F1-score of 94%. In sum, they used EMD to filter out data noise, constructed point cloud using Takens' embedding method and used persistent homology to construct features for unsupervised learning. This logic is applicable to all other research papers.

4.3.2 Classification

Features constructed using TDA can be applied for classification of time series as well. Umeda (2017) described an example with motion sensor data of daily and sports activities used in Altun and Barshan (2010); Altun et al. (2010), including both chaotic and non-chaotic time series data. Each of 19 activities was performed by 8 subjects and 60 signals were obtained for each activity and each subject, yielding 9120 time series. The sensor frequency was 512Hz and each signal was collected for 5 minutes. The pipeline of their method which was implemented in **MATLAB** is shown below.

Step 1. Construct a point cloud via Takens' embedding, $\{x_t, t = 1, 2, \dots, T\} \rightarrow \{\mathbf{v}_i\} \subset \mathcal{R}^3$, for $i = 1, 2, \dots, T - 2$ and $\mathbf{v}_i = (x_i, x_{i+1}, x_{i+2})$.

Step 2. Convert the point cloud to the persistence diagram $\tilde{\Omega} = \{\tilde{\tau}_{\tilde{p},k} | k = 1, 2, \dots, \tilde{p}, \tilde{p} = 0, 1, 2\}$ using Rips complex and persistent homology.

- Step 3.** Compute features using Betti sequences of \tilde{p} -th homology groups $\tilde{\Delta}_{\tilde{p}}(\lambda) = \#\{\lambda_{\tilde{p},k,1} \leq \lambda \leq \lambda_{\tilde{p},k,2} | k = 1, 2, \dots, k_{\tilde{p}}\}$ from the persistence diagram, discretized into 300 points for each \tilde{p} , and connected into one feature vector of length 900.
- Step 4.** Apply a one-dimensional convolutional neural network (CNN) (Krizhevsky et al., 2012) on the feature vector from Step 3 to do the classification.

They concluded that their approach performed better than an approach using support vector machine (SVM) (Hastie et al., 2001).

4.3.3 Structural Break Detection

TDA based features for structural break detection has been discussed in bioinformatics (Wang et al., 2018), financial data analysis (Gidea, 2017; Truong, 2017; Gidea and Katz, 2018; Gidea et al., 2018), etc. Here we describe the approach in Gidea et al. (2018), who used four major daily log-price cryptocurrencies (Bitcoin, Ethereum, Litecoin, and Ripple) between 2016-01-01 to 2018-02-28 independently for break detection of critical transitions:

- Step 1.** They constructed point clouds each with 50 points from each log-price time series $x_t, t = 1, 2, \dots, T$. Each of the $T - 52 = 448$ point clouds had a total of 50 points, the first point cloud being $(v_1, v_2, \dots, v_{50})$ and the last point cloud being $(v_{448}, v_{449}, \dots, v_{497})$, with $\mathbf{v}_i = (x_i, x_{i+1}, x_{i+2}, x_{i+3}) \in \mathcal{R}^4$.
- Step 2.** On each windowing point cloud $\mathcal{P}^{(w)}$, they constructed a persistence diagram $\tilde{\Omega}^{(w)}$ using Rips complex. Since they only focused on the 1-th homology groups, $\tilde{\Omega}^{(w)} = \{\tilde{\tau}_{1,k}^{(w)}, k = 1, 2, \dots, k_1\}$.
- Step 3.** They constructed all persistence landscapes $PL_{1,\nu}^{(w)}(\ell)$ from each $\tilde{\Omega}^{(w)}$, $\nu = 1, 2, \dots$, and then converted the persistence landscapes $PL_{1,\nu}^{(w)}(\ell)$ to its L_1 norm, which is defined as $\|PL_1^{(w)}\|_1 = \sum_{\nu=1}^{\infty} \|PL_{1,\nu}^{(w)}\|_1$, where $\|PL_{1,\nu}^{(w)}\|_1 = \int_{\mathcal{R}} PL_{1,\nu}^{(w)}(\ell) d\ell$.
- Step 4.** On each window, they combined the log-price time series x_w , first difference of the log-price values $x_{w+1} - x_w$ and the L_1 norm $\|PL_1^{(w)}\|_1$ as the feature vector $(x_w, x_{w+1} - x_w, \|PL_1^{(w)}\|_1)$ and used it to do K-means clustering with number of cluster $K = 18$.

They applied the method independently to each daily log-price cryptocurrency time series, and then used the clusters to identify topologically distinct regimes before the crash of each asset. They argued that this method has the potential to automatically recognize approaching critical transitions in the cryptocurrency markets, even when the relevant time series exhibit a highly non-stationary, erratic behavior.

5 Discussion and Summary

This paper gives a comprehensive overview of TDA which consists of a set of powerful tools for measuring topological features of time series and using it for pattern detection, clustering, classification, and structural break detection. Research extensions in several directions are possible. First, TDA for multivariate time series analysis has been studied very recently (Gidea and Katz, 2018; Gidea, 2017; Stolz et al., 2017). A second extension consists of using summary statistics and dissimilarity

measures for TDA. A review of summary statistics and dissimilarity measures refer to Nanopoulos et al. (2001); Fulcher (2017); Aghabozorgi et al. (2015). Third, research into improved computation tools in computational topology and further exploration of statistical properties while using TDA is a rich research area. Ongoing research can be separated into these different scenarios: computational homology (Phillips et al., 2015; De Silva and Carlsson, 2004; Liu et al., 2012; Carlsson and Zomorodian, 2009; Corbet et al., 2019; Ren et al., 2018), study of topological summaries (Bendich et al., 2016; Biscio and MÃžller, 2019; Reininghaus et al., 2015; Kusano et al., 2016; Carrière et al., 2017), and statistical inference (Fasy et al., 2014b; Phillips et al., 2015; Chazal et al., 2018; Alaa and Mohamed, 2017; Robinson and Turner, 2017; BréchetEAU, 2019).

References

- Adams, H. and Carlsson, G. (2015). “Evasion paths in mobile sensor networks.” *The International Journal of Robotics Research*, 34, 1, 90–104.
- Aghabozorgi, S., Seyed Shirkhorshidi, A., and Ying Wah, T. (2015). “Time-series Clustering - A Decade Review.” *Information Systems*, 53, C, 16–38.
- Alaa, H. and Mohamed, S. (2017). “On the Topological Data Analysis extensions and comparisons.” *Journal of the Egyptian Mathematical Society*, 25, 4, 406 – 413.
- Altun, K. and Barshan, B. (2010). “Human activity recognition using inertial/magnetic sensor units.” *International workshop on human behavior understanding*, 38–51.
- Altun, K., Barshan, B., and Tunçel, O. (2010). “Comparative study on classifying human activities with miniature inertial and magnetic sensors.” *Pattern Recognition*, 43, 10, 3605–3620.
- Bendich, P., Marron, J. S., Miller, E., Pieloch, A., and Skwerer, S. (2016). “Persistent homology analysis of brain artery trees.” *The annals of applied statistics*, 10, 1, 198.
- Berwald, J., Gidea, M., and Vejdemo-Johansson, M. (2013). “Automatic recognition and tagging of topologically different regimes in dynamical systems.” *CoRR*, abs/1312.2482.
- Biscio, C. A. N. and MÃžller, J. (2019). “The Accumulated Persistence Function, a New Useful Functional Summary Statistic for Topological Data Analysis, With a View to Brain Artery Trees and Spatial Point Process Applications.” *Journal of Computational and Graphical Statistics*, 0, 0, 1–21.
- Bonis, T., Ovsjanikov, M., Oudot, S., and Chazal, F. (2016). “Persistence-Based Pooling for Shape Pose Recognition.” *Proceedings of the 6th International Workshop on Computational Topology in Image Context - Volume 9667*, 19–29.
- BréchetEAU, C. (2019). “A statistical test of isomorphism between metric-measure spaces using the distance-to-a-measure signature.” *Electron. J. Statist.*, 13, 1, 795–849.
- Bubenik, P. (2015). “Statistical topological data analysis using persistence landscapes.” *The Journal of Machine Learning Research*, 16, 1, 77–102.
- Carlsson, G. (2009). “Topology and data.” *Bulletin of the American Mathematical Society*, 46, 2, 255–308.

- (2014). “Topological pattern recognition for point cloud data.” *Acta Numerica*, 23, 289–368.
- Carlsson, G. and Zomorodian, A. (2009). “The theory of multidimensional persistence.” *Discrete & Computational Geometry*, 42, 1, 71–93.
- Carlsson, G., Zomorodian, A., Collins, A., and Guibas, L. (2004). “Persistence Barcodes for Shapes.” *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, 124–135.
- Carrière, M., Cuturi, M., and Oudot, S. (2017). “Sliced Wasserstein Kernel for Persistence Diagrams.” *Proceedings of the 34th International Conference on Machine Learning*, 664–673.
- Carrière, M., Oudot, S. Y., and Ovsjanikov, M. (2015). “Stable Topological Signatures for Points on 3D Shapes.” *Proceedings of the Eurographics Symposium on Geometry Processing*, 1–12.
- Chazal, F., Cohen-Steiner, D., Guibas, L. J., Mémoli, F., and Oudot, S. Y. (2009). “Gromov-Hausdorff Stable Signatures for Shapes Using Persistence.” *Proceedings of the Symposium on Geometry Processing*, 1393–1403.
- Chazal, F., de Silva, V., and Oudot, S. (2014). “Persistence stability for geometric complexes.” *CoRR*, abs/1207.3885.
- Chazal, F., Fasy, B., Lecci, F., Bertr, Michel, Aless, ro Rinaldo, and Wasserman, L. (2018). “Robust Topological Inference: Distance To a Measure and Kernel Distance.” *Journal of Machine Learning Research*, 18, 159, 1–40.
- Chazal, F. and Michel, B. (2017). “An introduction to Topological Data Analysis: fundamental and practical aspects for data scientists.” *Journal de la Société Française de Statistique*.
- Chen, R., Zhang, J., Ravishanker, N., and Konduri, K. (2019). “Clustering Activity-Travel Behavior Time Series using Topological Data Analysis.” *arXiv preprint arXiv:1907.07603*.
- Corbet, R., Fugacci, U., Kerber, M., Landi, C., and Wang, B. (2019). “A kernel for multi-parameter persistent homology.” *Computers & Graphics: X*, 100005.
- Costantino, R. F., Cushing, J. M., Dennis, B., and Desharnais, R. A. (1995). “Experimentally induced transitions in the dynamic behaviour of insect populations.” *Nature*, 375, 6528, 227.
- De Silva, V. and Carlsson, G. E. (2004). “Topological estimation using witness complexes.” *SPBG*, 4, 157–166.
- De Silva, V. and Ghrist, R. (2007). “Coverage in sensor networks via persistent homology.” *Algebraic & Geometric Topology*, 7, 1, 339–358.
- DeWoskin, D., Climent, J., Cruz-White, I., Vazquez, M., Park, C., and Arsuaga, J. (2010). “Applications of computational homology to the analysis of treatment response in breast cancer patients.” *Topology and its Applications*, 157, 1, 157–164.
- Di Fabio, B. and Landi, C. (2011). “A Mayer-Vietoris Formula for Persistent Homology with an Application to Shape Recognition in the Presence of Occlusions.” *Foundations of Computational Mathematics*, 11, 5, 499.

- (2012). “Persistent homology and partial similarity of shapes.” *Pattern Recognition Letters*, 33, 11, 1445–1450.
- Edelsbrunner, H. and Harer, J. (2010). *Computational Topology. An Introduction..* American Mathematical Society.
- Fasy, B. T., Kim, J., Lecci, F., and Maria, C. (2014a). “Introduction to the R package TDA.” *CoRR*, abs/1411.1830.
- Fasy, B. T., Lecci, F., Rinaldo, A., Wasserman, L., Balakrishnan, S., and Singh, A. (2014b). “Confidence sets for persistence diagrams.” *Ann. Statist.*, 42, 6, 2301–2339.
- Fulcher, B. D. (2017). “Feature-based time-series analysis.” *Computing Research Repository*, abs/1709.08055.
- Gidea, M. (2017). “Topological data analysis of critical transitions in financial networks.” *International Conference and School on Network Science*, 47–59.
- Gidea, M. and Katz, Y. (2018). “Topological data analysis of financial time series: Landscapes of crashes.” *Physica A: Statistical Mechanics and its Applications*, 491, 820–834.
- Gidea, M., Katz, Y. A., Roldan, P., Goldsmith, D., and Shmalo, Y. (2018). “Topological recognition of critical transitions in time series of cryptocurrencies.” *Social Science Research Network*.
- Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc.
- Hegger, R., Kantz, H., and Schreiber, T. (1999). “Practical implementation of nonlinear time series methods: The TISEAN package.” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 9, 2, 413–435.
- Heo, G., Gamble, J., and Kim, P. T. (2012). “Topological analysis of variance and the maxillary complex.” *Journal of the American Statistical association*, 107, 498, 477–492.
- Huang, N. E., Shen, Z., Long, S. R., Wu, M. C., Shih, H. H., Zheng, Q., Yen, N.-C., Tung, C. C., and Liu, H. H. (1998). “The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis.” *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454, 1971, 903–995.
- Kennel, M. B., Brown, R., and Abarbanel, H. D. (1992). “Determining embedding dimension for phase-space reconstruction using a geometrical construction.” *Physical review A*, 45, 6, 3403.
- Khasawneh, F. A. and Munch, E. (2016). “Chatter detection in turning using persistent homology.” *Mechanical Systems and Signal Processing*, 70, 527–541.
- Kovacev-Nikolic, V., Bubenik, P., Nikolić, D., and Heo, G. (2016). “Using persistent homology and dynamical distances to analyze protein binding.” *Statistical applications in genetics and molecular biology*, 15, 1, 19–38.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). “Imagenet classification with deep convolutional neural networks.” *Advances in neural information processing systems*, 1097–1105.

- Kusano, G., Hiraoka, Y., and Fukumizu, K. (2016). “Persistence weighted Gaussian kernel for topological data analysis.” *International Conference on Machine Learning*, 2004–2013.
- Li, C., Ovsjanikov, M., and Chazal, F. (2014). “Persistence-based Structural Recognition.” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, 1995–2002.
- Liu, X., Xie, Z., Yi, D., et al. (2012). “A fast algorithm for constructing topological structure in large data.” *Homology, Homotopy and Applications*, 14, 1, 221–238.
- Mileyko, Y., Mukherjee, S., and Harer, J. (2011). “Probability measures on the space of persistence diagrams.” *Inverse Problems*, 27, 12, 124007.
- Munkres, J. R. (1993). *Elements of Algebraic Topology*. Addison-Wesley.
- Nanopoulos, A., Alcock, R., and Manolopoulos, Y. (2001). “Information Processing and Technology.” chap. Feature-based Classification of Time-series Data, 49–61. Commack, NY, USA: Nova Science Publishers, Inc.
- Nicolau, M., Levine, A. J., and Carlsson, G. E. (2011). “Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival.” *Proceedings of the National Academy of Sciences of the United States of America*, 108 17, 7265–70.
- Palais, R. S. (1963). “Morse theory on Hilbert manifolds.” *Topology*, 2, 4, 299–340.
- Perea, J. A., Deckard, A., Haase, S. B., and Harer, J. (2015). “SW1PerS: Sliding windows and 1-persistence scoring; discovering periodicity in gene expression time series data.” *BMC bioinformatics*, 16, 1, 257.
- Perea, J. A. and Harer, J. (2015). “Sliding windows and persistence: An application of topological methods to signal analysis.” *Foundations of Computational Mathematics*, 15, 3, 799–838.
- Pereira, C. M. and de Mello, R. F. (2015). “Persistent Homology for Time Series and Spatial Data Clustering.” *Expert Syst. Appl.*, 42, 15, 6026–6038.
- Phillips, J. M., Wang, B., and Zheng, Y. (2015). “Geometric Inference on Kernel Density Estimates.” *31st International Symposium on Computational Geometry (SoCG 2015)*, 34, 857–871.
- Reininghaus, J., Huber, S. M., Bauer, U., and Kwitt, R. (2015). “A stable multi-scale kernel for topological machine learning.” *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4741–4748.
- Ren, S., Wu, C., Wu, J., et al. (2018). “Weighted persistent homology.” *Rocky Mountain Journal of Mathematics*, 48, 8, 2661–2687.
- Robinson, A. and Turner, K. (2017). “Hypothesis testing for topological data analysis.” *Journal of Applied and Computational Topology*, 1, 2, 241–261.
- Seversky, L. M., Davis, S., and Berger, M. (2016). “On time-series topological data analysis: New data and opportunities.” *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1014–1022.
- Shanks, J. L. (1969). “Computation of the Fast Walsh-Fourier Transform.” *IEEE Trans. Comput.*, 18, 5, 457–459.

- Silva, V. D. and Ghrist, R. (2007). “Homological sensor networks.” *Notices Amer. Math. Soc.*, 10–17.
- Stoffer, D. S. (1991). “Walsh-Fourier Analysis and Its Statistical Applications.” *Journal of the American Statistical Association*, 86, 414, 461–479.
- Stolz, B. J., Harrington, H. A., and Porter, M. A. (2017). “Persistent homology of time-dependent functional networks constructed from coupled time series.” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27, 4, 047410.
- Takens, F. et al. (1981). “Detecting strange attractors in turbulence.” *Lecture notes in mathematics*, 898, 1, 366–381.
- Truong, P. (2017). “An exploration of topological properties of high-frequency one-dimensional financial time series data using TDA.” Ph.D. thesis, KTH Royal Institute of Technology.
- Umeda, Y. (2017). “Time Series Classification via Topological Data Analysis.” *Transactions of the Japanese Society for Artificial Intelligence*, 32, 3.
- Van de Weygaert, R., Vegter, G., Edelsbrunner, H., Jones, B. J. T., Pranav, P., Park, C., Hellwing, W. A., Eldering, B., Kruithof, N., Bos, E. G. P. P., Hidding, J., Feldbrugge, J., ten Have, E., van Engelen, M., Caroli, M., and Teillaud, M. (2011). “Transactions on Computational Science XIV.” chap. Alpha, Betti and the Megaparsec Universe: On the Topology of the Cosmic Web, 60–101. Berlin, Heidelberg: Springer-Verlag.
- Wang, Y., Ombao, H., and Chung, M. K. (2018). “Topological data analysis of single-trial electroencephalographic signals.” *The annals of applied statistics*, 12, 3, 1506.