Comparing TDA Methods of Time Series Analysis

1 Background & Motivation

Time series analysis is a well-explored field of interest, with a focus on studying the seasonality and topological structures of such data. Time series with repeating patterns of measures are sometimes called seasonal, and the number of times a given pattern is repeated in one time interval is referred to as the periodicity or frequency of the time series.

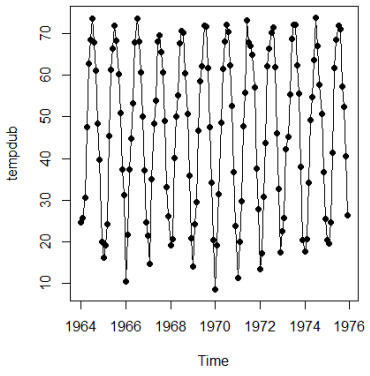


Figure 1

For some series, periodicity is easy to identify and expected. For instance, consider the collection of average monthly temperatures in Dubuque Iowa from 1964 to 1975 in Figure 1 (Cryer and Chan [2008]). This time series is seasonal in the sense that a pattern of low-to-high-to-low temperatures repeats through the 4 seasons of each year a total of 12 times. Hence, this data is periodic with frequency of 12 cycles over 12 years. Not all time series have periodicities that are easy to identify if they exist, however. Because of this, we review and compare several methods that have been constructed to study the periodicity of such time series.

The topology of time series, however, is a more recently studied field. We summarize and apply a variety of methods constructed to analyze the topology of time series. We pull most of these methods from Ravishanker and Chen [2019]. We implement these pipelines on three synthetic time series with three increasing frequencies on the same domain. These constructions are inspired by Perea et al. [2015] and Ravishanker and Chen [2019]. Our functions are given by:

$$\text{Low frequency: } x(t) = sin\Big(2\pi(\tfrac{2}{T})t\Big) \text{ (2 cycles every } T \text{ time points)}$$
$$\text{Mid frequency: } y(t) = sin\Big(2\pi(\tfrac{4}{T})t\Big) \text{ (4 cycles every } T \text{ time points)}$$
$$\text{High freqency: } z(t) = sin\Big(2\pi(\tfrac{8}{T})t\Big) \text{ (8 cycles every } T \text{ time points)}$$

2 Converting Time Series to Point-Clouds

One common approach for studying the topology of time series is the use of persistence homology. This involves varying some parameter within a given set of data and summarizing which structural features (i.e. connected components, holes, voids, tunnels, etc.) appear and disappear throughout the variation. However, using this approach requires that data have point cloud structure (i.e. are sampled from the underlying surface of some n-dimensional manifold). Time series do not inherently have point-cloud structure, and because of this, cannot be used directly for topological data analysis (TDA). One common approach to resolve this issue is to map time series data points onto d-dimensional point clouds. We discuss one such mapping, Takens embedding. This embedding was constructed by Floris Takens (Takens [1981]), and involves mapping each time measure in $\{x(t)|t = 1, 2, \dots T\}$ to a point cloud in d dimensions using a time delay parameter τ. The embedding is given by:

$$x_t \to v_t = \big(x_t, x_{t+\tau}, x_{t+2\tau}, \dots, x_{t+(d-1)\tau}\big)^T.$$

Hence the resulting point cloud will have $N = T-(d-1)\tau$ d-dimensional points. The embedding dimension d and delay (or lag) parameter τ are user-chosen. Several methods including False Nearest Neighbors (FNN)

Tests have been implemented to select choices of d (Truong [2017], Khasawneh and Munch [2016], Seversky et al.), while some just assume $d = 2$ or $d = 3$ (Pereira and de Mello [2015]). Kennel's and Cao's tests are two such FNN tests commonly utilized (Krakovská et al. [2015]). Methods using the autocorrelation function (Khasawneh and Munch [2016], Truong [2017]) and mutual information (Pereira and de Mello [2015]) of a time series have also been implemented to select choices of τ. For our implementation, we select $d = 2$ and $\tau = 30$ for easy visualization, and so we have a sufficient number of points in our point cloud. Observe our resulting series, embedded point clouds, and persistence diagrams after this embedding below. Generally, the more periodic the series (i.e. the higher the frequency of repeating patterns), the more circular or rounded the point cloud is. Hence in a given Vietoris-Rips filtration, as we increase our filter parameter around points, circles in the point cloud will live longer. This is shown in Figure 2 with point clouds becoming increasingly rounded and the red triangles appearing farther away from the diagonal as we move from $x(t)$ to $z(t)$. One feature to note about Takens embedding is that it preserves topological properties of time series across different dimensions.

3 Comparing Distances Between Diagrams

As well as studying the persistence homology of time series, many are also interested in comparing homology in different dimensions. One method used to do this is to quantify any differences between diagrams by computing distances between their corresponding points (Ravishanker and Chen [2019]). In our context, we will use Wasserstein distance between pairs of persistence diagrams to show the preservation of topological features for increasing embedding dimensions.

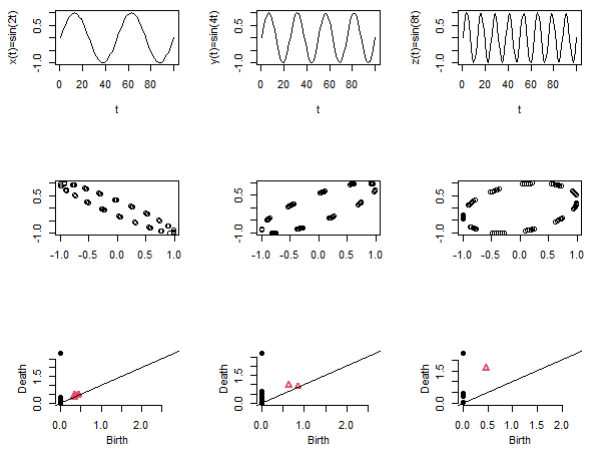


Figure 2

Let's first define Wasserstein distance. Let $\sigma_{p,k}$ be the k-th p homology class produced in a persistence diagram Γ_1 for a given series. Suppose we have another persistence diagram Γ_2 produced from the same time series. For all possible bijections f between points in Γ_1 and Γ_2, the Wasserstein distance between both diagrams is given as:

$$W_{q,p}(\Gamma_1, \Gamma_2) = \Big(inf_f \sum_{\sigma_{p,k} \in \Gamma_1} |\sigma_{p,k} - f(\sigma_{p,k})|_\infty^q\Big)^{1/q} \quad \text{for } q = 1, 2, \ldots$$

When our Wasserstein dimension q is ∞, we are just minimizing the maximum distance between any two points in Γ_1 and Γ_2 (Ravishanker and Chen [2019]). This is the bottleneck distance:

$$W_{\infty,p}(\Gamma_1, \Gamma_2) = inf_f\Big(max_{\sigma_{p,k} \in \Gamma_1}\{|\sigma_{p,k} - f(\sigma_{p,k})|\}\Big)$$

Observe the persistence diagrams of $x(t)$,$y(t)$, and $z(t)$ in columns 1, 2, and 3 of Figure 3, respectively. We compute these diagrams for embedding dimensions 2,3,and 4 in rows 1, 2, and 3, respectively. Notice that visually we can see that the topological properties are preserved for each series for each dimension. That is, the low-frequency diagrams (column 1) are the most different compared to the middle and high-frequency diagrams (columns 2 & 3) in terms of distances between their 0 and 1-homology groups. As well, the middle and high-frequency diagrams are the most similar. These patterns are shown in our table of Wasserstein and Bottleneck distances between all pairs of diagram points. In all three cases of dimension, the largest distances computed were between the low-frequency and mid or high-frequency diagrams, and the lowest distances computed were between the low and mid-frequency or mid and high-frequency diagrams.

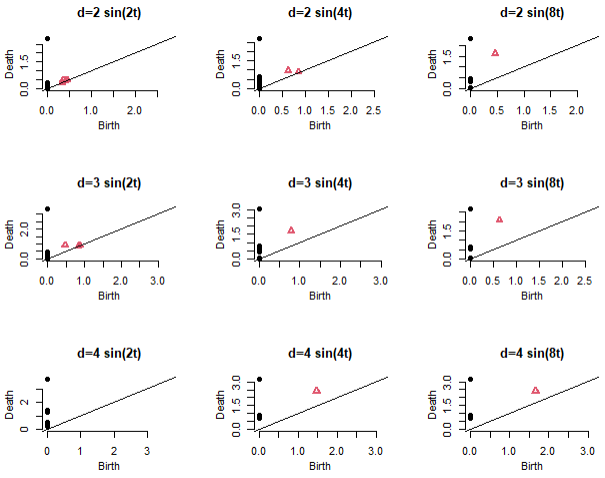
4 Quantifying Periodicity

SW1PerS (Sliding Windows and 1-Persistence Scoring) is a method developed by Perea et al. [2015] to quantify the periodicity of a time series by measuring how often distinct patterns in its signal repeat. Most of our code is borrowed from Ravishanker and Chen [2019]. We summarize the steps below:

Simulate Noisy Series. We sample 216 evenly-spaced time units on $[0, 2\pi]$ and evaluate our three series x, y, and z at these times. We then add Gaussian noise to each with mean 0 and constant variance 0.5. We explain the steps used to obtain a periodicity score for $x(t)$. We repeat these methods for y and z.

De-Noise Series. To de-noise our data, we obtain a locally averaged version of $x(t)$, $x_{avg}(t)$, using a simple moving average with window size one third of the selected embedding dimension. We select Pereira and de Mello [2015]'s recommended embedding dimension $d = 15$.

Fit a Cubic Spline to Series. We then construct a continuous approximation of $x_{avg}(t)$ by mapping its time units $0, \ldots, T-1$ linearly to the interval $[0, 2\pi]$. We fit a cubic spline to $x_{avg}(t)$ to obtain its continuous version $x_{ct}(t) : [0, 2\pi] \to \mathbb{R}$ such that $x_{ct}(0) = x_{avg}(0)$ and $x_{ct}(2\pi) = x_{avg}(T-1)$.



q=1	d=2	d=3	d=4
W(low,high)	2.741	4.046	4.095
W(low,mid)	2.371	3.573	4.145
W(mid,high)	2.408	1.796	0.212

q=infty	d=2	d=3	d=4
W(low,high)	0.500	0.678	0.379
W(low,mid)	0.281	0.321	0.359
W(mid,high)	0.379	0.564	0.050

Figure 3

Embed the Series Using Sliding Windows. We then embed $x_{ct}(t)$ to a point cloud using sliding windows of length τ. That is, we fix $\tau \in (0, 2\pi)$ and restrict $x_{ct}(t)$ to the window $[t, t+\tau]$.

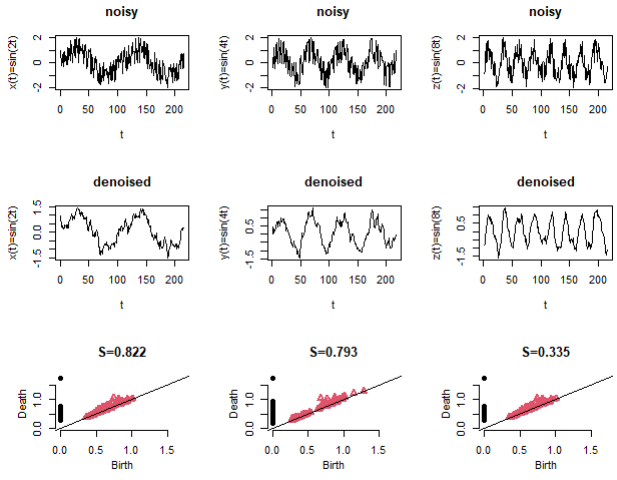


Figure 4

As we vary t from 0 to 2π, we obtain several snapshots of $x_{ct}(t)$. Each of these snapshots corresponds to a point in the embedded point cloud. The sliding windows embedding of $x_{ct}(t)$ is formally defined as $x_{ct}(t) \to v_t = \Big(x_{ct}(t), x_{ct}(t+\tau), \ldots, x_{ct}\big(t + (d-1)\big)\tau\Big)$.

Normalize & Centralize Points. We lastly normalize and center each of our embedded points v_t so that they lay on the surface of the unit sphere and the distance between any two points is less than $\frac{\pi}{16}$ (Perea et al. [2015]). Define the average of each d-dimensional point as $avg(v_t) = \frac{1}{d}\sum_{i=1}^{d} v_{t,i}$, for $v_{t,i}$ the i-th component in v_t. Then each normalized and centered point v_t^{norm} is given by $v_t^{norm} = \frac{v_t + avg(v_t)\mathbf{1}}{\sqrt{\sum_{i=1}^{d}\big(v_{t,i} - avg(v_t)\big)^2}}$.

Compute Periodicity Score. As previously mentioned, each point v_t^{norm} in the sliding windows point cloud P represents one snapshot of x_{ct}. The more similar any two snapshots are, the closer together these points lay on P. In addition, the greater the number of distinct patterns (snapshots) in x_{ct}, the greater the hole in P is. We obtain b and d such that $0 \leq b \leq d \leq \sqrt{3}$ from the 1-Persistence Homology Algorithm (Perea et al. [2015]). The parameter b corresponds to the maximum distance between any point $v_t^{norm}(t)$ in P and its nearest neighbor, while d measures how rounded and wide P is. Then the 1-Persistence Score of $x(t)$ is given by $S = 1 - \frac{d^2-b^2}{3}$. Intuitively, the more periodic $x(t)$ is, the wider and more round P gets and hence $d \to \sqrt{3}$. At the same time, as $x(t)$ becomes more periodic, its snapshots become more similar and hence closer together on P, so $b \to 0$. Thus as the periodicity increases, $S \to 0$, and vice verse for decreasing periodicity ($S \to 1$).

See Figure 4 that displays our three noisy series x,y, and z (row 1), their de-noised counterparts after local averaging (row 2), and their persistence diagrams after filtering their sliding windows embeddings (row 3). Observe that the series increase in periodicity from left to right, and as expected, the scores decrease.

5 Sublevel Set Filtration

We introduce another method used to compute the persistence homology of a time series. For this method, persistence diagrams are produced using sublevel set filtration on the distance to measure (DTM) function of a given series (Ravishanker and Chen [2019], Chazal et al. [2018]). We first review the basics of sublevel set filtration. Consider a continuous function $f : \mathbb{R}^d \to \mathbb{R}$ that is defined on a set of d-dimensional points from a point cloud S. Then the sublevel set of f at level λ is considered to be the set of all d-dimensional points whose function value is less than or equal to λ, $L_\lambda(f) = \{\mathbf{x} \in S \subseteq \mathbb{R}^d | f(\mathbf{x}) \leq \lambda\}$.

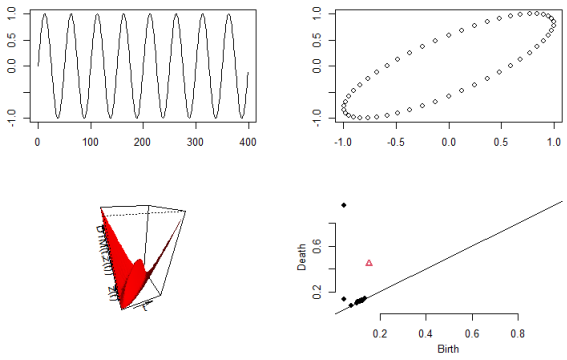


Figure 5

Our filtration involves varying λ and outputting a persistence diagram of all p-dimensional features that appear and disappear throughout. We specifically select our continuous function f to be the DTM function given by:

$$f(x) = \sqrt{\frac{1}{k} \sum_{x_i \in N_k(x)} ||x_i - x||^2}$$

where $N_k(x)$ is the subset of k nearest neighbors $\{x_i\}_{i=1}^k$ of each point x in the point cloud S. As an example, we perform sublevel set filtration on points in the embedded point cloud S of $z(t)$. We define the frequency of z to be 8 cycles for every 400 units of time. We implement the following pipeline (most of our code is borrowed from Ravishanker and Chen [2019]):

Observed Time Series. We simulate 400 evenly spaced time points on $[0, 399]$ and store the corresponding values of $z(t)$. We then add Gaussian noise to z with 0.3 variance. See our plot of $z(t)$ before adding noise in Figure 5 (top left).

Convert to Point Cloud. We map $z(t)$ to 395, 2-dimensional points $(z(t), z(t + \tau))$ using Takens embedding with $\tau = 5$. Visualize the point cloud in Figure 5 (top right).

Apply DTM Function to Points in S. We apply the DTM function f to each point in the point cloud, i.e. $f(\mathbf{x}) = f\Big(\big(z(t), z(t + 5)\big)\Big)$. Hence $f : \mathbb{R}^2 \to \mathbb{R}$ is a 3-dimensional plot. See the graph of f in Figure 5 (bottom left).

Sublevel Set Filtration. We apply sublevel set filtration on f to produce a persistence diagram. This diagram reveals all p-dimensional features that appear and disappear as we vary our values of $f(\mathbf{x})$ between 0 and ∞. We use the R package **TDA** and the function **gridDiag()** to do this (Ravishanker and Chen [2019]). See Figure 5 for the persistence diagram (bottom right).

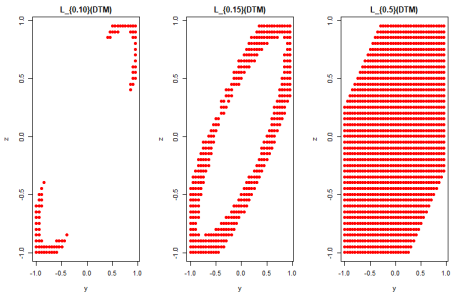


Figure 6

Visualize Sublevel Sets.

In our persistence diagram, we observe that a 1-dimensional hole is born when $f \approx 0.15$ and dies when $f \approx 0.44$. This corresponds to the hole produced from the horizontal cross section of our red DTM plot at $f = 0.15$ and continuing until $f = 0.44$. We visualize this hole appearing and disappearing by plotting these cross sections at $f = 0.10$ (before the hole appears), at $f = 0.30$ (after the hole appears), and at $f = 0.50$ (after the hole disappears). For instance, the sublevel set at 0.3 is the subset $L_{0.3}(f)$ of all points $\big(z(t), z(t+5)\big)$ from the point cloud whose DTM value $f\Big(\big(z(t), z(t+5)\big)\Big)$ is less than or equal to 0.3. See Figure 6 for these cross sections. Cross sections at 0.10, 0.30, and 0.50 are shown in red on the left, middle, and right respectively.

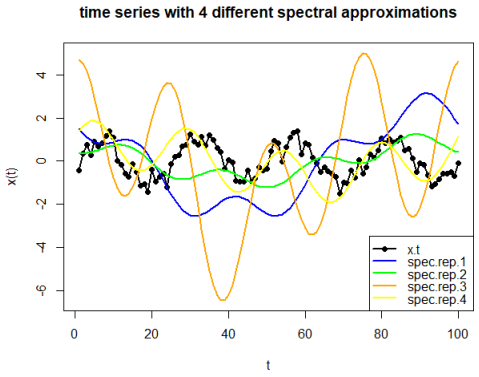
6 Approximations of Time Series

6.1 Fourier Approximations

Periodicity of time series has also been studied using approximations by sinusoidals (Stoffer [1991]). These sinusoidals are called spectral representations of a given time series. That is, for more complicated series whose periodicity is not able to be easily identified or assumed, one can instead study the frequency of their sinusoidal approximations (sums of sine and cosine functions).

Given a time series $\{x(t)|t = 0, \ldots, 99\}$, with constant sample mean such that the covariance between any two time-measures is a function of the lag τ (time difference) between them, we can approximate $x(t)$ using the spectral representation $x(t) = \sum_{j=1}^{q}[a(j)cos(2\pi\lambda_j t)+b(j)sin(2\pi\lambda_j t)]$ for q frequencies $\lambda_1, \ldots, \lambda_q$, and q pairs of mutually uncorrelated amplitudes $(a(j), b(j))$ with mean 0 and constant variance σ_j^2 , $j = 1, \ldots, q$.

As an example, we compute Fourier approximations of $y(t) = x(t)$, with a periodicity of 4 cycles for every 100 units of time. We begin by sampling 100 evenly spaced time points from 0 to 99. We then compute time measures $y(t) = sin(2\pi \cdot \frac{4}{100}t)$ for $t = 0, \ldots, 99$. Next we add Gaussian noise with 0.3 variance, $\epsilon_t \sim N(0, 0.3)$. We plot the original noisy series in black, along with 4 different Fourier representations in Figure 7. We also include a table of parameters for each of these approximations.



Representation	q	variance sigma	lambda 1	lambda 2	lambda 3	lambda 4	color
1	4	1	1/100	1/100	1/100	4/100	blue
2	4	0.5	1/100	1/100	1/100	4/100	green
3	4	1	1/100	1/100	2/100	4/100	orange
4	3	1	1/100	1/100	4/100	N/A	yellow

Figure 7

We next compute a periodogram to identify the best frequency approximation of y. We do so by first constructing the sine and cosine representations of y separately as functions of varying frequencies $\lambda_j = \frac{j}{T}$ (j cycles per every T time units). Secondly, we use the summation of the resulting sinusoidal values squared to plot a periodogram for frequency analysis (Stoffer [1991]). We define the cosine and sine Fourier representations of $y(t)$ to be $C(\lambda_j) = \frac{1}{\sqrt{T}}\sum_{t=0}^{T-1} y(t)cos(2\pi\lambda_j t)$ and $S(\lambda_j) = \frac{1}{\sqrt{T}}\sum_{t=0}^{T-1} y(t)sin(2\pi\lambda_j t)$, respectively.

The periodogram is a collection of points on the graph of $I(\lambda_j) = C^2(\lambda_j) + S^2(\lambda_j)$ v.s. λ_j that produces peaks of varying size. The tallest peak in the periodogram, $\big(\lambda_j^*, I(\lambda_{j^*})\big)$, corresponds to the best approximation λ_j^* of the true frequency of y. Here, we compute various frequencies λ_j as $\lambda_j = j/T$ for $1 \le j \le \frac{T}{2}, T = 100$. Observe our resulting periodogram to the left. As shown in Figure 8, the highest peak

occurs at a frequency of 0.04, which corresponds to the true frequency of $y(t)$ ($\frac{4}{100}$). Hence, Fourier analysis provides another way to study the periodicity of a time series by analyzing the periodicity of sinusoidals that approximate it.

6.2 Walsh-Fourier Approximations

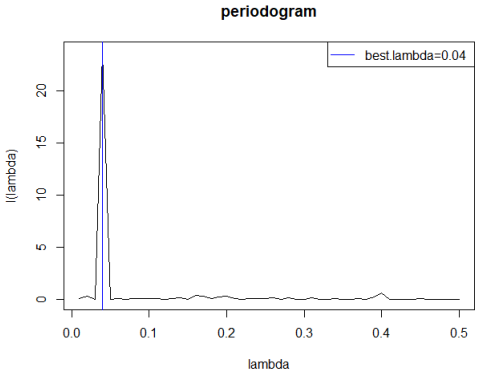


Figure 8

Not all time series maintain periodic behavior. Some series display a more box-like wave structure. For these series, we can apply similar methods to approximate their seasonal behavior using square-wave functions called Walsh functions (Stoffer [1991]). Using Walsh functions to analyze time series is called Walsh-Fourier analysis.

For instance, observe the time series in Figure 9. This series looks to be better approximated using square waves rather than sinusoidal waves. Note that Walsh functions can also be used to approximate sinusoidal series as well. For the purpose of comparing methods, we will demonstrate the use of Walsh functions to approximate the same series $y(t) = x(t)$ with a periodicity of 4 cycles per every T units of time, and then perform sublevel set filtration on its second order spectra to produce a persistence diagram. We will be using **gridDiag()** to do this.

We define a Walsh function to take on two values, either 1 or -1. Then, for each time unit t of an observed series $\{f(t)|t = 0, \dots T-1\}$, we can convert the series to a sequence of Walsh functions $W(t, \lambda_j)$, where t is the number of zero-crossings (moments where the Walsh function switches from -1 to 1 or vice versa) in a given period of time T, and j is a given input of the Walsh function defined over the time sequence $t = \{0, \dots, T-1\}$. Intuitively, for a given value of t and j, the j-th value of the Walsh function with t zero-crossings over T units of time is given as the value $W(t, \lambda_j) = \pm 1$, and $\lambda_j = \frac{j}{T}$ refers to the j-th sequency.

More formally, we construct Walsh functions using the mapping $W(t, \lambda_j) = \{(-1)^k | j \in \{k \cdot \frac{T}{t+1}, \dots, (k+1) \cdot \frac{T}{t+1} - 1\}\}$ for $k = 0, \dots, t$, $t = 0, \dots, T-1$. This is our construction, but an alternative definition of Walsh functions that requires the order of the time series to be a power of 2 can be found in Shanks [1969], Ravishanker and Chen [2019], as well as in Cooley and Tukey [1964]. We also apply an example of this in section 7.1. Our definition simply assigns Walsh values based on values of j that fall within $t+1$ intervals. These intervals are computed by dividing the length of the period T by the number of zero-crossings plus 1, $(t+1)$. The Walsh transform of a time series is defined as $d_w = \frac{1}{\sqrt{T}} \sum_{t=0}^{T-1} f(t) W(t, \lambda_j)$ for $j = 0, \dots, T-1$.

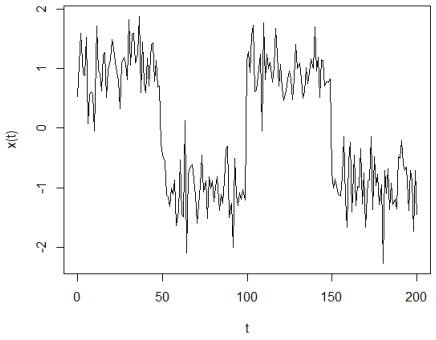


Figure 9

The corresponding Walsh periodogram is given by $I_w(j) = |d_w|^2$. The highest peak in the plot of $I_w(\lambda_j)$ v.s. $\lambda_j = \frac{j}{T}$ corresponds to the closest approximation of the sequency of f (the number of zero-crossings per T time units). For our series $y(t)$ with 7 crossings for every 400 time units, the periodogram should produce an approximation close to $\frac{7}{400}$.

We first plot four examples of Walsh functions for $t = 2, 4, 6$ and 7 in the first two rows of Figure 10. We then plot a periodogram for the given series and observe the highest spectra to occur at $j/T = 0.02$ (see the bottom left image in Figure 10). Hence, our best Walsh-approximation of the sequency of y (0.02) is fairly close to the true sequency (0.0175). We lastly perform sublevel set filtration on our Walsh periodogram to compute a persistence diagram (the bottom right image in Figure 10). See Ravishanker and Chen [2019] for another application of sublevel set filtration on the periodogram of a modified Walsh transform.

7 Persistence Landscapes

The construction of persistence landscapes provides a way to study the underlying distribution of persistence diagrams for a given time series (Ravishanker and Chen [2019]). Moreover, landscapes are typically used to cluster the data into meaningful topological parts (Ravishanker and Chen [2019]). That is, Bubenik [2015] was able to show that given a sample of n persistence diagrams, one can construct their corresponding persistence landscapes, and the sample mean of these landscapes converges to the true mean of the underlying distribution of diagrams. Hence persistence landscapes, under the right conditions, model the underlying structure of the diagrams for an observed time series (see Bubenik [2015] for a more in-depth explanation).

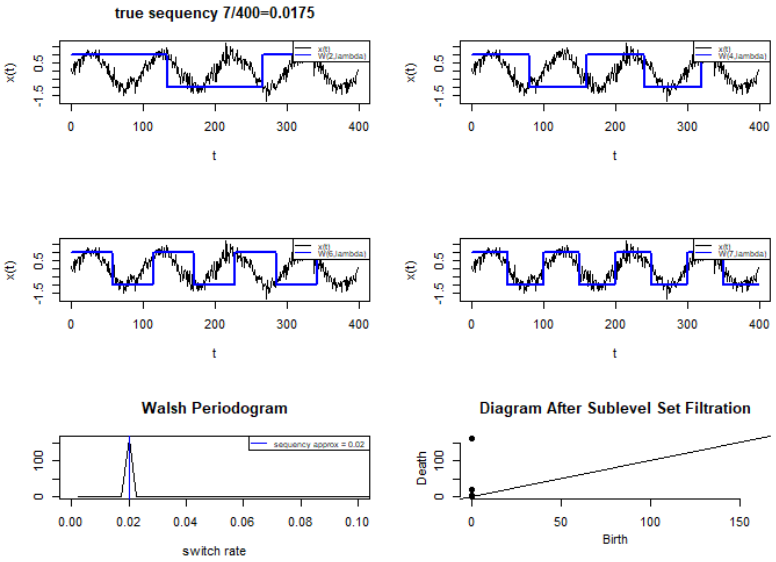


Figure 10

Suppose we wish to study the persistence landscapes of all p-homology groups in a given persistence diagram. Then the v-th order persistence landscape of these groups is given by:

$$PL_{p,v}(l) = \begin{cases} min\{l - \sigma_{p,k,1}, \sigma_{p,k,2} - l\} & \text{if } l - \sigma_{p,k,1}, \sigma_{p,k,2} - l > 0 \\ 0 & \text{o.w.} \end{cases}$$

for all k_p p-homology groups, $k = 1, \dots k_p$, each with a lifetime $[\sigma_{p,k,1}, \sigma_{p,k,2})$. Here l is a real number.

As an example, we compute the persistence landscapes of all 1-homology groups produced from our time series $z(t)$ with a frequency of 8 cycles per time period. We first apply Taken's Embedding to z with dimension 2 and lag 160 to generate a point cloud with 40, 2-dimensional points. We then implement the steps described in Ravishanker and Chen [2019] below to produce persistence landscapes of the resulting diagram. See Figure 11 for the persistence diagram of only the 1-homology groups of z (on the left) and its corresponding persistence landscapes (on the right).

STEP 1. We obtain all 1-homology groups from the persistence diagram of the embedded points of z. We have 4 1-homology groups, so $k_1 = 4$.

STEP 2. We construct a grid of real-valued inputs l to plug into the persistence landscape function $PL_{1,v}(l)$. We sample 501 evenly-spaced values of l from the minimum birth time $M_1 = min_{k=1,\dots,4}\{\sigma_{1,k,1}\}$ to the maximum death time $M_2 = max_{k=1,\dots,4}\{\sigma_{1,k,2}\}$. Here, the step-size for our points l is given by $\delta = \frac{M_2 - M_1}{500}$. Thus our inputs are $l = M_1, M_1 + \delta, M_1 + 2\delta, \dots, M_2$.

STEP 3. For each input l, we compute the persistence landscape $PL_{1,k}(l)$ for $k = 1, \dots, 4$. Hence, for each of our 501 inputs l, we should have a list of 4 outputs $PL_{1,k}$, one for each order k.

STEP 4. We lastly sort the landscape values corresponding to each value of l in decreasing order (from largest to smallest). We denote the ordering for each set of outputs as $PL_{1,k}(l) = PL_1^{(1)}(l), PL_1^{(2)}(l), PL_1^{(3)}(l), PL_4^{(4)}(l)$, where the v-th order persistence landscape of the 1-homology groups is the set of all 501 points $PL_{1,v}(l) = PL_1^{(v)}(l)$ for $v = 1, \dots, k_1 = 4$.

The idea here is that the peak of each landscape represents an important topological feature in the diagram. The lower the order v, the greater the significance of the 1-dimensional topological feature (i.e.

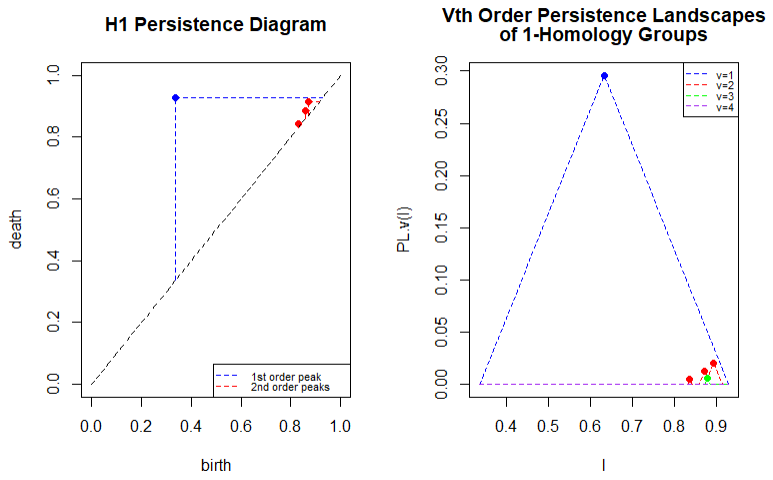


Figure 11

the longer it survived in the filtration). For instance, notice that in the 1st order landscape, the blue peak represents the longest-surviving 1-homology group (denoted as a blue point in our persistence diagram). On the other hand, the 2nd order landscape depicts three peaks (shown as red points). These three peaks correspond to the remaining three 1-homology groups in our diagram (denoted as red points). These red points are closer to the diagonal, and hence represent 1-homology groups that didn't survive as long during the filtration.

7.1 More Applications of Persistence Landscapes

Using a Smoothed Weighted Fourier Transform. This method is used to generate persistence landscapes of continuous time series. Particularly, Wang et al. [2018] uses the smoothed weighted Fourier transform $\hat{\mu}_{T_\mu}(t)$ of an EEG time series to construct persistence landscapes of various orders. This transform is summarized by Ravishanker and Chen [2019] as:

$$\hat{\mu}_{T_\mu}(t) = \sum_{j\in I_1} e^{-(\frac{j}{T}2\pi)^2\sigma} a_j cos(\tfrac{j}{T}2\pi t) + \sum_{j\in I_2} e^{-(\frac{j}{T}2\pi)^2\sigma} b_j sin(\tfrac{j}{T}2\pi t).$$

Here, $a_j = \frac{2}{T}\sum_{t=1}^{T} x_t cos(\frac{j}{T}2\pi t)$ and $b_j = \frac{2}{T}\sum_{t=1}^{T} x_t cos(\frac{j}{T}2\pi t)$ are the regular Fourier coefficients that are normally distributed with mean 0 and constant variance σ^2. The variance is user chosen.

The Fourier sum a_0 is defined as the sample mean of all time series points, $a_0 = \frac{1}{T}\sum_{t=1}^{T} x_t$. The list of possible frequencies for cosine and sine transforms are given respectively as $I_1 = \{j = 0, \ldots, k : |a_j| > T_u\}$ and $I_2 = \{j = 1, \ldots, k : |b_j| > T_u\}$ for $T_u = s\sqrt{2log(n)}$. The parameter k is user-chosen as the maximum number of cycles for which to construct the Fourier sum. Wang et al. [2018] selected a maximum frequency of $k = 99$ cycles for their $T = 500$ EEG points. The parameter n is the number of points in each phase of the series. Lastly, to compute the median of absolute deviation, s, we first compute $a^{(m)} = median\{|a_i| : i = 1, \ldots, k\}$ and $b^{(m)} = median\{|b_j| : i = 1, \ldots, k\}$. Then $s = median\{|a_i - a^{(m)}|, |b_j - b^{(m)}| : i, j = 1, \ldots, k\}$. We apply this method to our original time series $y(t)$ with a frequency of 4 cycles for every T units of time. We select $T = 500$ evenly spaced time points so that our series is $\{y(t)|t = 1, \ldots, 500\}$ and choose the maximum number of cycles to be $k = 99$. We select the period to be 500 so that $n = 500$ (there are 500 points for every phase of $y(t)$). We then compute the Fourier Transform of these points using our Morse Function, $\hat{\mu}_{T_\mu}$. We plot the original series in black and the transformed series in blue in Figure 12. Wang et al. [2018] performed sublevel set filtration on the Morse function to compute a persistence diagram for landscape computation. Instead, we implement Takens embedding to compute $N = 477$, 2-dimensional points ($d = 2$, $\tau = 23$). We plot the resulting point cloud in Figure 12. We next compute the persistence diagram using Vietoris-Rips filtration on the point cloud with the function **gridDiag()**.

We then use the diagram to compute the persistence landscape of the 1-homology groups presented throughout the filtration (13 total). The resulting 13 persistence landscapes of orders $v = 1, \ldots, 13$, along with the persistence diagram of y are shown in Figure 12, as well. We can see that the highest peak presented in the 1st order landscape $PL_1^{(v)}(l)$ corresponds to the longest-surviving 1-homology group displayed as the red triangle farthest away in the diagram.

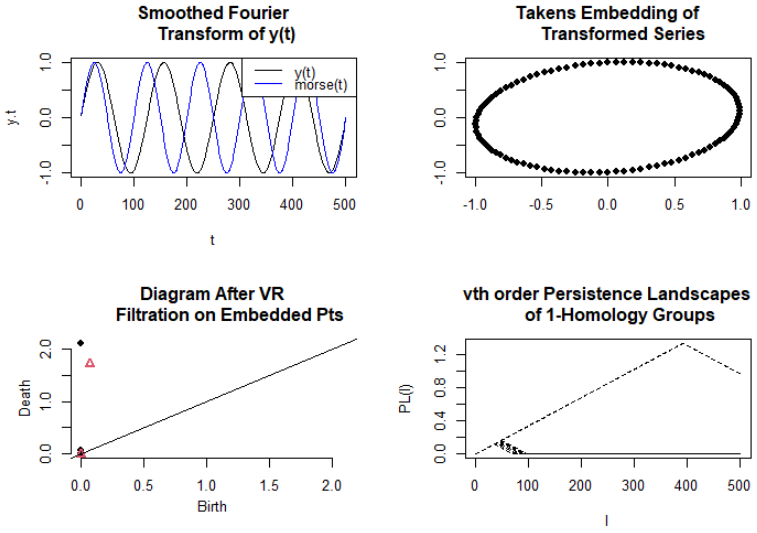


Figure 12

Using a Fast Walsh-Fourier Transform. This method is used to generate persistence landscapes of discontinuous time series. We will provide an application of computing the persistence landscape of a Walsh-Fourier (square-wave) transform of three categorical time series similar to $x(t)$, $y(t)$, and $z(t)$ (Ravishanker and Chen [2019]). As previously defined in section 6.2, Walsh functions are typically used to approximate time series with square-like signals for which frequency is analagously defined as sequency (number of zero-crossings per every T time units).

We denoted the j-th value of the Walsh function with t crossings as $W(t, \lambda_j)$, and the corresponding Walsh transform of the series $x(t)$ as $d_w = \frac{1}{\sqrt{T}} \sum_{t=0}^{T-1} x(t) W(t, \lambda_j)$. We introduce a fast Walsh Fourier transform (FFT) using the Cooley-Tukey algorithm (Cooley and Tukey [1964]). Cooley & Tukey define the FFT of a series $\{x(t) | t = 0, \ldots, T_2 - 1\}$ as follows:

$$x(j) = \sum_{t=0}^{T_2-1} A(t) W^{jk} \text{ for } j = 0, \ldots T_2 - 1$$

where the Walsh values W are defined as complex numbers $W = e^{2\pi i/T}$ with complex coefficients $A(t)$. Here, $T_2 = T$ if T is a power of 2. Otherwise, we define the next power of 2 greater than T to be $T_2 = 2^p$ for some natural number p. We then add a zero-padding of $T_2 - T$ zeros to the original series $x(t)$ (Ravishanker and Chen [2019], Shanks [1969]).

The Cooley-Tukey algorithm works by splitting the series $x(t)$ in half, with one half $x_e(t)$ being all even-indexed measures and the other half $x_o(t)$ being all odd-indexed measures. Then the Cooley-Tukey FFT is applied to both half-series:

$$W_e(j) = \sum_{t=0}^{T_2-1} A(t) e^{2\pi i j/T}, \; j = 0, 2, \ldots, T_2 - 2$$
$$W_o(j) = \sum_{t=0}^{T_2-1} B(t) e^{2\pi i j/T}, \; j = 1, 3, \ldots, T_2 - 1$$

Then the FFT of the entire series $x(t)$, $W(j)$, is given as $x(j) = W_e(j) + \sum_{j=0}^{\frac{T_2}{2}-1} (e^{2\pi i T_2})^j W_o(j)$ for $j = 0, \ldots, \frac{T_2}{2} - 1$ and $x(j) = W_e(j-2) + \sum_{j=\frac{T_2}{2}}^{T_2-1} (e^{2\pi i T_2})^j W_o(j-2)$ for $j = \frac{T_2}{2}, \ldots, T_2 - 1$ (Nabeel [2023]). We implement Cooley-Tukey using the **fft()** function in R (a similar function is also available in Python using the SciPy package). We specifically apply Cooley-Tukey transformations to three categorical series $x(1,t)$, $x(2,t)$, and $x(3,t)$ with respective zero-switches 2, 4, and 8 (between ± 1). We then compute the 1st order persistence landscapes of these three transforms using Chen et al. [2019]'s method. We plot the three categorical series as well as their FFT's in rows 1 and 2 of Figure 13, respectively. We now have three FFTs:

$$W(2, \lambda_j) \text{ of } x(1,t)$$
$$W(4, \lambda_j) \text{ of } x(2,t)$$
$$W(8, \lambda_j) \text{ of } x(3,t)$$

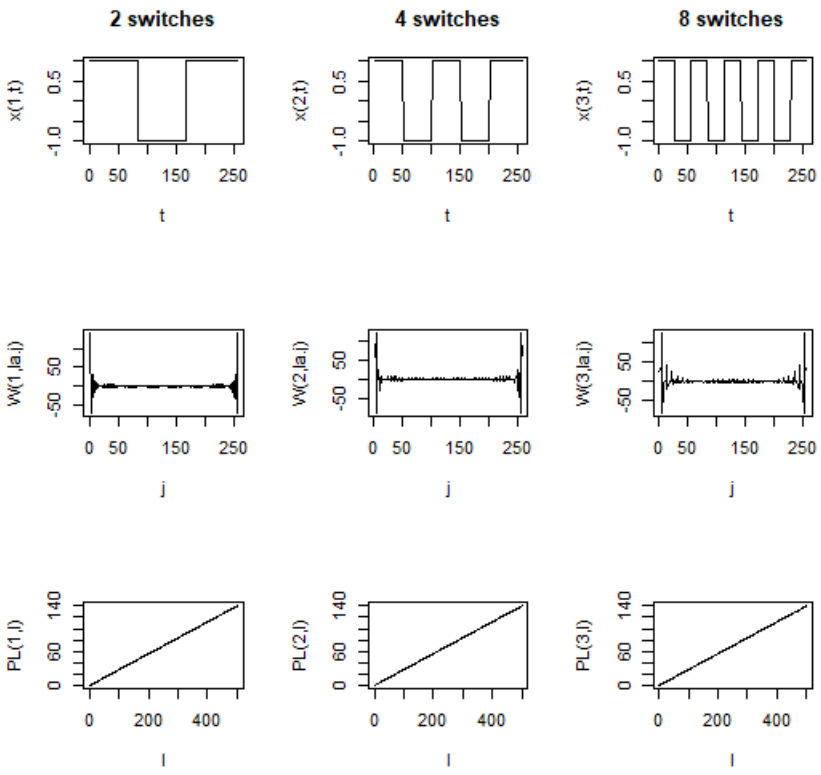


Figure 13

The 1st order persistence landscape of the n-th transform is given by $PL(n,l) = min(V_1(n,l), V_2(n,l))_+$ for $n = 1, \ldots, N$ and $l = 1, \ldots, L$, where N is the number of series (3 in our case) and L is the number of inputs (we select $L = 500$). The parameters of this function are defined as follows:

$V_1(n,l) = W_{min} - \frac{(l-1)(W_{max}-W_{min})}{L-1} - W_{n,min}\}$

$V_2(n,l) = W_{min} - \frac{(l-1)(W_{max}-W_{min})}{L-1} - W_{n,max}\}$

$W_{n,min} = min_j\{W(n,\lambda_j)\}$ (the minimum value of the nth FFT)

$W_{n,max} = max_j\{W(n,\lambda_j)\}$ (the maximum value of the nth FFT)

$W_{min} = min_n\{W_{n,min}\}$ (take the smallest value among the N Walsh minimums)

$W_{max} = max_n\{W_{n,max}\}$ (take the largest value among the N Walsh maximums)

$(a)_+$ is the positive part of a.

See the third row of Figure 13 for the 1st order persistence landscapes of all three Cooley-Tukey transforms. Notice that all of our landscapes are linear, i.e. there are no peaks. This could be due to the fact that landscapes of Fast Fourier Transforms are typically used to summarize unique topological features in the data rather than in their persistence diagrams, and our three categorical series all have evenly-spaced sequencies (i.e. no left or right skewed zero-switches) so any uniqueness of the features presented in the transform may not be revealed in their landscapes. Another FFT is defined by Shanks [1969]. It is defined as follows for $t = 0, \ldots, T_2 - 1$ ($T_2 = 2^p$):

$$W(0,j) = 1 \text{ for } j = 0, \ldots, T_2 - 1$$

$$W(1,j) = \begin{cases} 1 & \text{for } j = 0, \ldots, \dfrac{T_2}{2} - 1 \\ -1 & \text{for } j = \dfrac{T_2}{2}, \ldots, T_2 - 1 \end{cases}$$

$$W(t,j) = W(\left[\tfrac{t}{2}\right], 2j) \cdot W(t - 2\left[\tfrac{t}{2}\right], j)$$
$$j = 0, \ldots, T_2 - 1,\ t = 2, \ldots, T_2 - 1.$$

where $\left[\frac{t}{2}\right]$ is the integer part of $\frac{t}{2}$ (i.e. $\left[\frac{3}{2}\right] = 1$).

8 Other Uses of Topological Summaries

Topological summaries of time series have also been utilized for clustering, classification, and signal break detection as discussed below (Ravishanker and Chen [2019]). One common pipeline for feature construction in the context of clustering begins with de-noising a time series, embedding it into a point cloud, constructing a persistence diagram, and building features from the diagram for K-means clustering. Takens Embedding and Vietoris-Rips filtrations are two methods that have been used for this process. Pereira and de Mello [2015] applied this pipeline to cluster topological features of two sets of flower growth series, one with periodic structure, and the other with aperiodic.

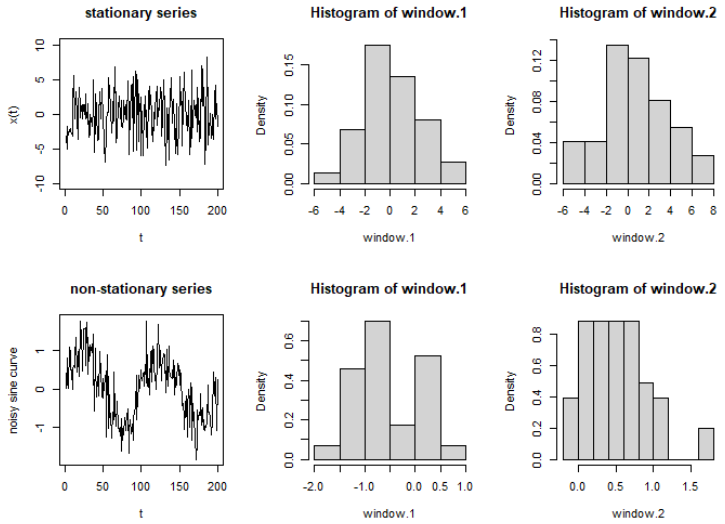


Figure 14

Constructing topological features from time series has also been used for building predictive models (classifiers). This methodology typically begins by embedding a time series, producing a persistence diagram, constructing topological features from the diagram, and feeding these features into machine learning models to make predictions about the series. Altun and Barshan [2010] and Altun et al. [2010] used this methodology to build classifiers for 1-dimensional homology groups of a series of daily and sports activities.

Feature construction has also been used to detect significant changes or breaks in time series data. Common steps for this approach begin by embedding a time series, producing a diagram, using the diagram to build persistence landscapes, using these landscapes to construct an L1-norm (Ravishanker and Chen [2019]), and using the L1-norm to construct feature vectors for k-means clustering. The clusters intend to detect critical transitions in the time series. Gidea et al. [2018] applied this pipeline to detect clusters of log-price cryptocurrency data occurring just before the crash of their assets.

9 Stability of Current Topological Summaries of Time Series

A topological summary of a time series is considered stable if its features and expected behaviors are preserved when subject to changes in the parameters involved within the TDA approach itself. Typically, stability is shown using distance metrics between topological features. Current stable TDA approaches include persistence diagrams and persistence barcodes. Naturally, it is important to ensure that TDA approaches for studying time series are stable in nature. We hence review existing stability results for such TDA methods.

9.1 Takens Embedding

Takens embedding, under the right assumptions, has been shown to preserve distances between distinct states of the attractor of a dynamical system (Yap et al. [2014]). Let the time series $s(t)$ be a dynamical system and $s(t_0)$ a given state of this system at time t_0. Then $s(t)$ is the image of its observed state space containing all states $x(t) \in \mathcal{M} \subset \mathbb{R}^N$ for some manifold $\mathcal{M}$. The mapping that produces this image is defined as the function $h : \mathbb{R}^N \to \mathbb{R}$ given by $h(x(t)) = s(t) \forall t$. Given an interval τ of $s(t)$, we define the flow of the state space from one state to the next by the flow function $\phi : \mathcal{M} \to \mathcal{M}$, where $\phi(x(t)) = x(t+\tau)$ for $x(t), x(t+\tau) \in \mathcal{M}$. If the flow of a given state space only moves in one direction, this is called dissipative flow, and it implies that the states of $s(t)$ converge to a submanifold of $\mathcal{M}$, called the attractor of $s(t)$. That is, $x(t) \to \mathcal{A} \subset \mathcal{M}$ as $t \to \infty$. We define Takens embedding as a coordinate mapping of the states in $\mathcal{A}$ to M-dimensional points in a reconstructed state space. More specifically, Takens embedding of the state space of a dynamical system $s(t)$ is given by $F : \mathbb{R}^N \to \mathbb{R}^M$ such that $F(x(t)) = \big(s(t), s(t+\tau), \ldots, s(t+(M-1)\tau)\big)^T$. Observe that if h is the identity map, then $s(t) = x(t)$, and we define the mapping F just as we do in section 2 given $M = d$. Takens embedding preserves the topology of the attractor, but not its geometry (Takens [1981]). That is, F ensures that the mapping between $\mathcal{A}$ and $F(\mathcal{A})$ is one-to-one (maps any 2 points in $\mathcal{A}$ to 2 distinct points in $F(\mathcal{A})$), but does not guarantee that points close together (far apart) in $\mathcal{A}$ remain close (far) in its image under F. Yap et al. [2014] shows the geometric stability of Takens embedding for linear observation functions h that map points in $\mathcal{A}$ to its trajectory manifold under F. Consider the basis of linear observation functions $h_p : \mathbb{R}^N \to \mathbb{R}$ for $p = 1, \ldots, P$ given by $h_p(x(t)) = s(t)$ for all states $x(t) \in \mathcal{A}$. Let h_α be one such mapping and define Takens embedding as $F_\alpha(x(t)) = \Big(h_\alpha(x(t)), \ldots, h_\alpha(\phi^{-M+1}(x))\Big)^T$ for smooth flow function ϕ. Then define the trajectory vector of state $x(t) \in \mathcal{A}$ as $\tilde{g}(x(t)) = \Big(x(t), \phi^{-1}(x(t)), \ldots, \phi^{-M+1}(x(t))\Big)^T \in \mathbb{R}^M$. Hence the trajectory of $\mathcal{A}$ is given as $\tilde{g}(\mathcal{A})$. Yap et al. [2014] was able to show that F_α with M delays is a stable embedding of $\tilde{g}(\mathcal{A})$:

$$(1-\delta)||\tilde{g}(x(t)) - \tilde{g}(y(t))||_2^2 \leq ||F_\alpha(x(t)) - F_\alpha(y(t))||_2^2 \leq (1+\delta)||\tilde{g}(x(t)) - \tilde{g}(y(t))||_2^2$$

for all distinct points $x(t)$ and $y(t)$ on $\mathcal{A}$. This result only follows the assumption that the infemum over all soft ranks of the differences $G_{x(t)} - G_{y(t)}$ is bounded below by a required threshold, where $G_{x(t)}$ and $G_{y(t)}$ are the matrix versions of the trajectory vectors $\tilde{g}(x(t))$ and $\tilde{g}(y(t))$, respectively. See Yap et al. [2014] for more detailed descriptions.

9.2 SW1PerS Embedding

SW1PerS embedding remains stable with regards to computing distances between embedded points in a sliding windows point cloud, producing scores when input series are subject to noise, as well as producing point clouds containing bounded distance between any pair of their points. Firstly, SW1Pers necessarily obtains a distance measure between points in a point cloud of a given signal that correspond to the similarity between their respective snapshots. Applying cubic splining and computing a moving average on any input series allows SW1Pers to handle particularly noisy series. As well as this, normalizing and centering points in the point cloud enables SW1PerS to guarantee the distance between any two points remains bounded above by $\frac{\pi}{16}$, as well as to handle input series with trends, damping, and a variety of amplitudes. These aspects of SW1Pers maintain its stability in terms of producing periodicity scores when subject to several changes in its input parameters.

9.3 Persistence Landscapes

Persistence Landscapes are shown to preserve distances between pairs of functions defined on a topological space X, and hence are stable. Bubenik [2015] was able to prove stability of persistence landscapes using Wasserstein and Bottleneck distance as defined in section 3. Recall that a persistence module of a filtered simplicial complex at a given level of the filtration X_r is the collection of all homology classes $H(X_r)$ paired with the set of all linear maps induced between $H(X_r)$ and $H(X_{r'})$ for $r \leq r'$, $H(l_r^{r'})$. Here, l is the induced linear map from X_r to $X_{r'}$. Let M and M' be a pair of persistence modules of a given filtration, and λ and

λ' be their corresponding persistence landscapes. Then for $1 \leq p \leq \infty$, the p-landscape distance between M and M' is given by the p-norm of the difference between both landscapes: $\Lambda_p(M, M') = ||\lambda - \lambda'||_p$. Equivalently, the p-landscape distance between the corresponding diagrams of M and M', D and D', is given as $\Lambda_p(D, D') = ||\lambda - \lambda'||_p$. Bubenik [2015] was able to prove the stability of the persistence landscape with respect to the supremum norm by showing that for some functions $f, g : X \to \mathbb{R}$ defined on a topological space X, the ∞-landscape distance between modules $M(f)$ and $M(g)$ is bounded above by the supremum norm of the difference between f and g:

$$\Lambda_\infty(M(f), M(g)) \leq ||f - g||_\infty$$

Bubenik [2015] was also able to show stability for finite p using persistence weighted p-Wasserstein distance between two diagrams D and D'. Let $x_j = (b_j, d_j) \in D$ and $\phi(x_j) = x'_j = (b'_j, d'_j) \in D'$ for bijection $\phi : D \to D'$. Then $l_j = d_j - b_j$ is the persistence of topological feature $x_j \in D$ and $\epsilon_j = ||x_j - x'_j||_\infty$ is the maximum distance between x_j and any other topological point x'_j in D'. The persistence weighted p-Wasserstein distance between D and D' is given as the following infemum over all possible bijections ϕ: $\bar{W}_p(D, D') = inf_{\phi:D\to D'}\Big[\sum_j l_j \epsilon_j^p\Big]^{\frac{1}{p}}$. Bubenik [2015] hence shows that the persistence landscape is stable with respect to the p-landscape distance if $p > k$, whenever topological space X implies bounded degree-k total persistence. They do so by proving that a preserving upper bound on this finite landscape distance exists:

$$\Lambda_p(D(f), D(g))^p \leq C||f - g||_\infty^{p-k}$$

$\forall p \geq k$ and tame Lipshitz functions f and g with $C = C_{X,k}||f||_\infty\big(Lip(f)^k + Lip(g)^k\big) +$
$C_{X,k+1}\frac{1}{p+1}\big(Lip(f)^{k+1} + Lip(g)^{k+1}\big)$. Here, $C_{X,k}$ is the constant that satisfies the bounded degree-k total persistence of X.

10 Discussion

A common historical point of interest when studying time series has been the analysis of its topological features, as well as how these features change in relation to periodicity. Topological feature analysis follows a common pipeline across most methods we've discussed. One begins with a (possibly noisy) time series, denoises the series, embeds the series into a point cloud, performs a filtration on the point cloud, and produces a persistence diagram that represents the topological and periodic features of the time series. Common methods we've discussed within this pipeline include the study of time series data, its Fourier or Walsh approximations, the use of Takens and Sliding Windows embeddings, as well as the use of Vietoris-Rips or sublevel set filtrations of periodograms or Morse Functions for diagram construction. Another aspect of topological analysis that we've discussed is the distribution of persistence diagrams for a given series. This includes the comparison of similarities and dissimilarities between diagrams using Wasserstein & Bottleneck distances, as well as computing topological summaries of these diagrams using persistence landscapes. We've seen that persistence diagrams of embedded time series, as well as its Fourier and Walsh representations have both been used to construct persistence landscapes. Periodicity (sequency) of time series has also been studied through both its quantification using a score function (SW1PerS), as well as through its estimation using spectral analysis of Fourier (Walsh) approximations. Along with the development of several topological summaries of time series (i.e. Takens, SW1PerS, & landscapes), stability results have also been proven.

11 Future Work

While hundreds of machine learning models have been developed to classify time series data, many models have focused on prediction of univariate time series (Faouzi [2023]). Of those multivariate models that have been developed (i.e. TOTOPO, Franceschi, 1-NN DTW (Dynamic Time Warping), 1-NN Euclidean, etc.), even these have limitations with classification. For instance, TOTOPO does not perform as well on time series with classes of data that are too similar (Pilyugina et al. [2020]). As well, several methods of cluster analysis on time series data have also been done (i.e. Temporal-based, Representation-based, Model-based, etc.) and

also pose difficulties when applied to high-dimensional time series (Heka.ai [2022]). Among these limitations, one pertains to knowing which is the optimal choice of similarity (distance) measure that defines the clustering of time series data points. Popular measures include Hausdorff distance, DTW, Euclidean distance, etc. Most often, deciding which measure to use is a process of trial and error, as well as making educated decisions based on the nature of the data and the type of questions one is trying to answer (Heka.ai [2022]). Stability has been shown for persistence landscapes and Takens embedding, however stability for Takens embedding has only been shown for linear observation functions h on trajectory manifolds of Riemannian attractors $\mathcal{A}$. Yap et al. [2014] asserts that that stability of these embeddings has yet to be explored for nonlinear observation functions, trajectory manifolds of non-Riemannian attractors, and for submanifolds $F(\mathcal{A})$ rather than trajectory manifolds $\tilde{g}(\mathcal{A})$. SW1PerS shows stability when subject to changes in parameters and in terms of bounded distance, however I have yet to find literature formalizing the notion of preserving distance between points in a given series.

My work aims to contribute and hopefully address some of these limitations posed. With regards to time series classification, I hope to construct another alternative machine learning approach that is able to provide stronger predictions on data containing similar classes, and can be applied to both univariate and multivariate time series. I also plan to obtain a clustering method that is able to be used on high-dimensional data. Most literature has not presented cluster analysis of high-dimensional series using Mapper Algorithm, so this is one method for which I am interested in applying multivariate series. In addition, I hope to develop an efficient method for selecting an optimal choice of similarity measure for both clustering and classification. I also aim to further explore the stability of Takens embedding subject to different parameters as previously mentioned. Lastly, I would like to formalize and prove the stability of SW1PerS embedding in terms of preserving distances between points in a given time series. Through these processes, I hope to contribute to the somewhat sparse field of multivariate topological time series analysis, address the setbacks regarding the similarity measures and high-dimensionality that comes with this added complexity, as well as prove more stability results for embeddings of time series.

References

K. Altun and B. Barshan. Human activity recognition using inertial magnetic sensor units. *International workshop on human behavior understanding*, page 38–51, 2010.

K. Altun, B. Barshan, and O. Tunçel. Comparative study on classifying human activities with miniature inertial and magnetic sensors. *Pattern Recognition*, 43:3605–3620, 10 2010. doi: 10.1016/j.patcog.2010.04.019.

P. Bubenik. Statistical topological data analysis using persistence landscapes. 2015.

F. Chazal, B. Fasy, F. B. Lecci, A. Michel, R. Ro, and L. Wasserman. Topological inference: Distance to a measure and kernel distance. 18:1–40, 2018.

R. Chen, J. Zhang, N. Ravishanker, and K. Konduri. Clustering activity-travel behavior time series using topological data analysis. 2019.

J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex fourier series. 1964. URL `https://www.ams.org/journals/mcom/1965-19-090/S0025-5718-1965-0178586-1/S0025-5718-1965-0178586-1.pdf`.

J. D. Cryer and K.-S. Chan. *Time Series Analysis With Applications in R*. Springer, 2 edition, 2008. URL `https://mybiostats.files.wordpress.com/2015/03/time-series-analysis-with-applications-in-r-cryer-and-chan.pdf`.

J. Faouzi. Time series classification: A review of algorithms and implementations. *Machine Learning (Emerging Trends and Applications)*, 2023. ISSN 978-1-8381524-1-3. doi: ffhal-03558165f.

M. Gidea, Y. A. Katz, P. Roldan, D. Goldsmith, and Y. Shmalo. Topological recognition of critical transitions in time series of cryptocurrencies. *Social Science Research Network*, 2018.

Heka.ai. Time series clustering. *Medium*, 2022.

F. A. Khasawneh and E. Munch. Chatter detection in turning using persistent homology. *Mechanical Systems and Signal Processing, 70, 527-541*, 2016.

A. Krakovská, K. Mezeiová, and H. Budáčová. Use of false nearest neighbours for selecting variables and embedding parameters for state space reconstruction, 2015. URL `https://doi.org/10.1155/2015/932750`.

M. Nabeel. What is fft (cooley tukey algorithm)? *educative*, 2023. URL `https://www.educative.io/answers/what-is-fft-cooley-tukey-algorithm`.

J. A. Perea, A. Deckard, S. B. Haase, and J. Harer. Sw1pers: Sliding windows and 1-persistence scoring; discovering periodicity in gene expression time series data. *BMC bioinformatics 16, 1, 257*, 2015.

C. M. Pereira and R. F. de Mello. Persistent homology for time series and spatial data clustering. 2015.

P. Pilyugina, R. Rivera-Castro, and E. Burnaev. Totopo: Classifying univariate and multivariate time series with topological data analysis. 2020.

N. Ravishanker and R. Chen. Topological data analysis (tda) for time series. 2019.

L. M. Seversky, S. Davis, and M. Berger. On time series topological data analysis: New data and opportunities. pages 1014–1022. doi: 10.1109/CVPRW.2016.131.

J. Shanks. Computation of the fast walsh-fourier transform. *IEEE Transactions on Computers*, C-18(5): 457–459, 1969. doi: 10.1109/T-C.1969.222685.

D. Stoffer. Walsh-fourier analysis and its statistical applications. *Journal of the American Statistical Association*, 86:414, 461–479, 1991.

F. Takens. Detecting strange attractors in turbulence. *Lecture notes in mathematics*, 1981.

P. Truong. *An exploration of topological properties of high-frequency one-dimensional financial time series data using TDA*. PhD thesis, 2017.

Y. Wang, H. Ombao, and M. K. Chung. Topological data analysis of single-trial electroencephalographic signals. *The annals of applied statistics*, 12:3, 1506, 2018.

H. L. Yap, A. Eftekhari, M. B. Wakin, and C. J. Rozell. A first analysis of the stability of takens' embedding. *GlobalSIP: Information Processing for Big Data*, 2014. URL `https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7032148`.