Myers et al. *EPJ Data Science* (2023) 12:6
https://doi.org/10.1140/epjds/s13688-023-00379-5

EPJ Data Science
a SpringerOpen Journal

EPJ.org

REGULAR ARTICLE Open Access

Temporal network analysis using zigzag persistence

Audun Myers[1], David Muñoz[2], Firas A Khasawneh[1*] and Elizabeth Munch[2,3]

*Correspondence:
khasawn3@msu.edu
[1]Department of Mechanical Engineering, Michigan State University, 428 S. Shaw Lane, 48824, East Lansing, USA
Full list of author information is available at the end of the article

Abstract

This work presents a framework for studying temporal networks using zigzag persistence, a tool from the field of Topological Data Analysis (TDA). The resulting approach is general and applicable to a wide variety of time-varying graphs. For example, these graphs may correspond to a system modeled as a network with edges whose weights are functions of time, or they may represent a time series of a complex dynamical system. We use simplicial complexes to represent snapshots of the temporal networks that can then be analyzed using zigzag persistence. We show two applications of our method to dynamic networks: an analysis of commuting trends on multiple temporal scales, e.g., daily and weekly, in the Great Britain transportation network, and the detection of periodic/chaotic transitions due to intermittency in dynamical systems represented by temporal ordinal partition networks. Our findings show that the resulting zero- and one-dimensional zigzag persistence diagrams can detect changes in the networks' shapes that are missed by traditional connectivity and centrality graph statistics.

Keywords: Zigzag persistence; Temporal graph; Dynamical network; Topological data analysis; Persistent homology; Transportation network

1 Introduction

Network data, that is, the encoding of connections between objects, is a natural form of information representation in many fields [1]. While the analysis of static networks or graphs is already a broad field of research in itself, there is often much information ignored. In particular, we are interested in the case of *temporal networks* [2, 3]; that is, the case of a dynamical system represented by a network evolving over time. These networks can arise in many different cases, such as social networks [4], disease spread dynamics [5], manufacturer-supplier networks [6], power grid network [7], and transportation networks [8]. Many important characteristics of a dynamical network can be extracted from the data. These include source and rate of disease spread as well as predictions on future infections [9], weak branches in supply chains and possible failures [6, 10], changes in infrastructure to avoid cascade failures in power grids [7, 11], transportation network optimal routing (finding an optimal minimum time route between) [12], fault analysis (detecting transportation disruptions) in transportation networks [13], and flow pattern analysis (visualization) [14].

Springer

Temporal graph data is commonly represented using attributed information on the edges for the time intervals or instances in which the edges are active [15, 16]. Using this attributed information, we can represent the graph in several ways including edge labeling, snapshots, and static graph representations [17]. In this work we will first represent the data in the standard attributed (labeled) temporal graph structure and then use the graph snapshots approach, where our input is a sequence of static graphs $G_0, G_1, \ldots, G_n$.

The available tools for analyzing such data is often inspired by the tools from the static network community; for instance, centrality or flow measures [18]; temporal clustering for event detection [19–21]; and connectedness [22]. However, these tools do not account for higher-dimensional structures (e.g., loops as a one-dimensional structure). It may be important to account for evolving higher-dimensional structures in temporal networks to understand the changing structure better. For example, a highly connected network may only have one connected component with no clear clusters, but the number of loops within the network may detect the change.

For this reason, we introduce a method for incorporating ideas from Topological Data Analysis (TDA) [23, 24] to encode more complex structure than can be seen in the standard graph tools. The mainstay of TDA is persistent homology, colloquially referred to as persistence, which encodes structure by analyzing the changing shape of a simplicial complex (a higher dimensional generalization of a network) over a filtration (a nested sequence of subcomplexes $K_1 \subseteq K_2 \subseteq \cdots \subseteq K_n$). This shape is measured via homology, a vector space encoding information about topological structure of the space. Different dimensions of homology measure different things: 0-dimensional homology encodes information about connected components; 1-dimensional homology encodes loops; 2-dimensional homology encodes voids. How these structures change over the filtration (e.g. when a loop appears and then subsequently fills in) can be stored as a persistence diagram; that is, a collection of points in the upper half plane where a point at (i, j) means that a structure appeared at K_i but was lost at K_j.

However, a major limitation of standard persistence for the temporal network input data is the requirement that inclusions only go one way. Over the evolution parameter, our temporal graphs might add or remove edges, and we would like to build a system that can account for this. Thus, to study the evolving higher dimensional structures within a temporal network, we will leverage zigzag persistence [25], which allows for insertions and deletions of simplices at every step. The same mathematical theorems that make it possible to represent the information of standard persistence in a persistence diagram or barcode can be used in the case of a zigzag filtration. Zigzag persistence tracks the formation and disappearance of homological structures through a persistence diagram as a two-dimensional summary diagram.

The tools we employ will treat the graph input itself as a topological structure by using it to extract distance information between the vertices. For the work discussed here, we will assume the input networks are unweighted graphs and thus use the shortest path distance between vertices. However, this is by no means the only way to incorporate graph data for persistent homology; see [26] for a survey. Further, because we generally treat our networks as a metric space, our intuition is that the tools developed here are most applicable to data where vertices have a geometric component (such as networks where nodes come from geo-spatial data) rather than only combinatorial data (such as human

proximity networks, even if this has to do with physical proximity), but we look forward to being proven wrong.

We thus apply our methods to two sample data sets. The first is transportation data from Great Britain [27], where we can use the zigzag persistent homology for understanding higher order structures in the system, and see behavior such as daily periodicity represented automatically in the diagram output. The second data comes from input data from time series analysis. We have previously used zigzag persistence to detect Hopf bifurcations by taking as input a point cloud approximation of the underlying attractor to the time series [28]. However, the computational limitations of zigzag persistence on large point clouds makes its use in this setting unwieldy. Thus, we combine these ideas with recently available network representations of time series [29] to instead encode the structure of the time series in a graph. Previous work shows that measuring this structure using standard persistent homology can be used to differentiate between behaviors in the underlying dynamical system [30, 31].

1.1 Organization

We will start in Sect. 2 with an introductory background on persistent homology and zigzag persistence. Next, in Sect. 3, we overview the general pipeline for applying zigzag persistence to temporal graph data. We couple this explanation with a demonstrative toy example. In Sect. 4 we introduce the two systems we will study. The first is a dataset collected over a week of the Great Britain transportation system [27]. The second is an intermittent Lorenz system simulation, where we generate a temporal network through complex networks of sliding windows. Then we apply zigzag persistence to our two examples and show how the resulting persistence diagrams help visualize the underlying dynamics in comparison to standard temporal network analysis techniques.

2 Background

2.1 Persistent homology

Persistent homology, the flagship tool from the field of Topological Data Analysis (TDA), is used to measure the shape of a dataset at multiple dimensions. For example, it can measure connected components (dimension zero), loops (dimension one), voids (dimension two), and higher dimensional analogues. Persistent homology measures these shapes using a parameterized filtration to detect when the structures are born (appear) and die (disappear). We give the basic ideas in this section and direct the interested reader to more complete introductions to standard homology [32, 33] and persistent homology [23, 24, 34].

The required input for persistence is a filtration of a simplicial complex K. Specifically, a filtration $\{K_{a_i}\}_i$ is a parameterized sequence of simplicial complexes which are nested; i.e.

$$K_{\alpha_0} \subseteq K_{\alpha_1} \subseteq K_{\alpha_2} \subseteq \cdots \subseteq K_{\alpha_n}. \tag{1}$$

Under this notation, there are $n+1$ simplicial complexes; and it is often the case that K_{α_0} is either the empty complex or the complex consisting of only the vertex set. We can then calculate the homology of dimension p for each complex, $H_p(K_{a_i})$, which is a vector space representing the p-dimensional structure of the space such as loops, voids, etc. Surprisingly, there is further information to be used, namely that the inclusions on the simplicial

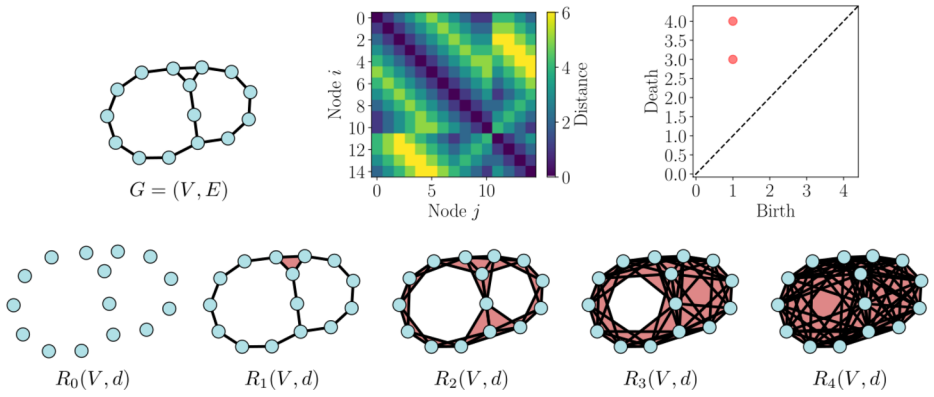


Figure 1 An example of a Rips filtration where the input is the shortest path distance on an example graph G. The persistence diagram in the top right is for standard persistence rather than zigzag as will be discussed later in the paper

complexes induce linear maps on the vector spaces resulting in a construction called a persistence module:

$$H_p(K_{\alpha_0}) \to H_p(K_{\alpha_1}) \to H_p(K_{\alpha_2}) \to \cdots \to H_p(K_{\alpha_n}). \tag{2}$$

The appearance and disappearance of classes in this object can be tracked, resulting in a representation of the information known as a persistence diagram. For each class which appears at K_{b_i} and disappears at K_{d_i}, we draw a point in the plane at (b_i, d_i). Taken together, the collection of points (also called persistence pairs), all of which are above the diagonal $\Delta = \{(x, y) \mid x = y\}$, is called a persistence diagram.

One common approach to obtain this setup is in the case where the input data is a finite metric space (M, d); i.e. a finite set M with distances given by $d(m_1, m_2)$. For example, M might be a finite point cloud $\chi \subset \mathbb{R}^k$ with d given by Euclidean distance. In the case of a graph G as input data, we can have $M = V$ the vertex set, and $d(u, v)$ as the number of edges in the shortest path between vertex u and vertex v.

From this metric space data, we fix a value $\alpha \geq 0$ and construct the Vietoris–Rips (VR) complex, denoted $R_\alpha(M)$. In the VR construction, we have a vertex set M and a simplex $\sigma \subseteq M$ is included in the abstract simplicial complex $R_\alpha(M)$ whenever $d(v, w) \leq \alpha$ for all $v, w \in \sigma$. Then by definition, $R_\alpha(\chi) \subseteq R_{\alpha'}(\chi)$ whenever $\alpha \leq \alpha'$. In this particular context, the axes of the persistence diagram correspond to distances between points. So, for example, points in the diagram that are far from the diagonal of the 1-dimensional persistence diagram represent large loop structures in the input data.

See Fig. 1 for an example of the filtration in this setting. The graph G is shown at the top left, and the distance defined on the nodes is given in the matrix at the top middle. The bottom row shows the Rips complexes for different choices of α parameter. In particular, note that $R_1(V, d)$ has the graph G as the 1-skeleton, but includes an additional triangle not present in the graph. The top right shows the 1-dimensional persistence diagram. Note that the two large loops in the graph are encoded as points in the diagram; while the small triangle is immediately filled in and thus not represented.

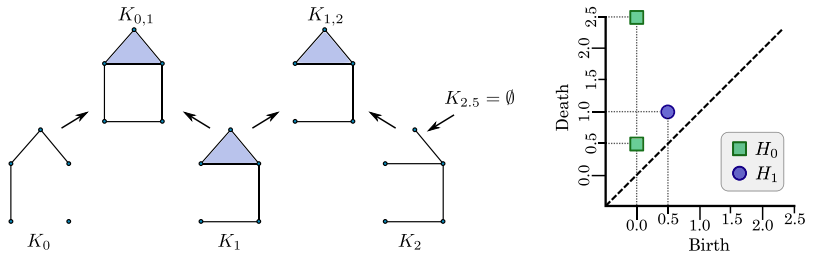


Figure 2 Example application of zigzag persistence to study changing topology of simplicial complex sequence. This example shows the sequence of simplicial complexes with intermediate unions and corresponding time stamps on the left and on the right the resulting zigzag persistence diagram on the right for dimension 0 and 1 as H_0 and H_1, respectively

2.2 Zigzag persistence

A limitation of the standard setup of persistent homology is that it requires each simplicial complex to be a subset of the next, as shown in Eq. (1). This means that at each step, we are only allowed to add new simplices to the previous complex to build this filtration. However, temporal graphs have no such behavior. Thus, this issue can be alleviated through zigzag persistence [25, 35], which allows for inclusions which can go either way at each step. This is often written as

$$K_{\alpha_0} \leftrightarrow K_{\alpha_1} \leftrightarrow K_{\alpha_2} \leftrightarrow \cdots \leftrightarrow K_{\alpha_n}, \tag{3}$$

where $\leftrightarrow$ denotes one of the two inclusions $\hookrightarrow$ and $\hookleftarrow$.

A common special case of this definition is where the left and right inclusions alternate, which can arise by taking a sequence of simplicial complexes, and interleaving them with either unions or intersections of the adjacent complexes. Focusing on the case of the union, denote $K_{i,i+1} = K_i \cup K_{i+1}$ to get the zigzag filtration

$$K_{\alpha_0} \hookrightarrow K_{\alpha_0,\alpha_1} \hookleftarrow K_{\alpha_1} \hookrightarrow K_{\alpha_1,\alpha_2} \hookleftarrow K_{\alpha_2} \hookrightarrow \cdots \hookleftarrow K_{\alpha_{n-1}} \hookrightarrow K_{\alpha_{n-1},\alpha_n} \hookleftarrow K_{\alpha_n}.$$

The same algebra that makes it possible for standard persistence to be represented in a diagram allows for computation of when homology features are born and die based on the zigzag persistence, however one must take care as some of the intuition from standard persistence is lost. We can again track this with a persistence diagram consisting of persistence pairs (b_i, d_i). In the case of a class appearing or disappearing at the union complex $K_{\alpha_i,\alpha_{i+1}}$, we draw the index at the average $(\alpha_i + \alpha_{i+1})/2$. If a topological feature persists through the last simplicial complex we set its death as the end time of the last window or index $n + 0.5$.

To demonstrate how zigzag persistence tracks the changing topology in a sequence of simplicial complexes we will use a simple example shown in Fig. 2. The sequence of simplicial complexes are shown as $[K_0, K_1, K_2]$, with unions given by $K_{0,1}$ and $K_{1,2}$. The persistence diagram then tracks where topological features of various dimensions of H_p (dimension 0 and 1 for this example) form and disappear. For example, for H_0 there are two components in K_0. At the next simplicial complex $K_{0,1}$ the two 0-dimensional features combine signifying one of their deaths which is tracked in the persistence diagram as the persistence pair $(0, 0.5)$ since 0.5 is the average of 0 and 1. The component that persists

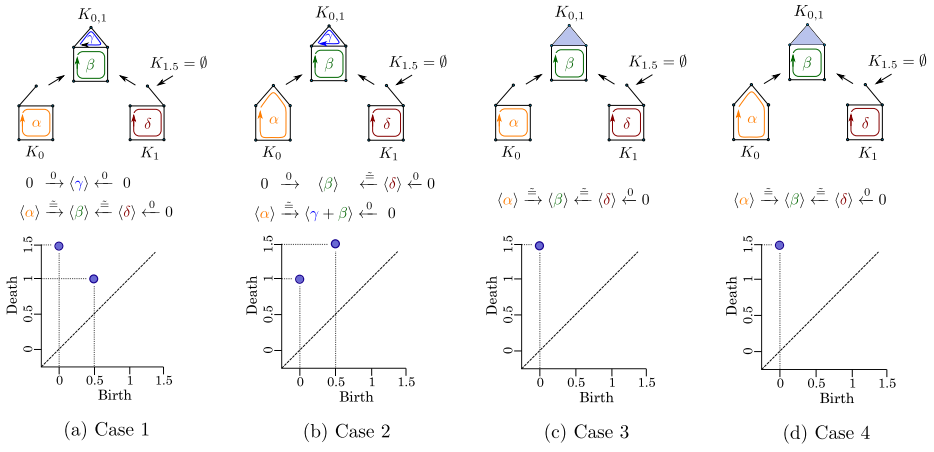


Figure 3 Several examples showing how the zigzag persistence diagram can be affected by the existence of small loops. The first and third; and the second and fourth examples have the same filtration with the addition of a single triangle in the later case. In each case, the generators of homology are shown with the interval decomposition drawn below. The resulting persistence diagrams show that the third and fourth examples show a single bar representing the persisting loop. However in the first two intervals are quite different, where the long lived bar is split into two pieces in the second example

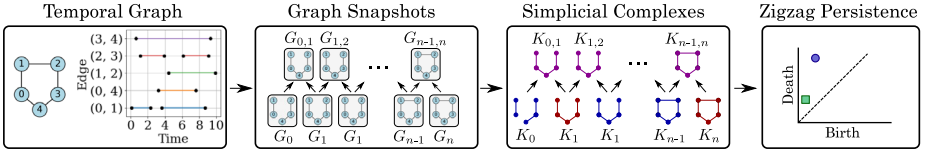


Figure 4 Pipeline for applying zigzag persistence to temporal networks. Begin with an unweighted and undirected *temporal graph* where each edge is on at a point or interval of time. Create *graph snapshots* using a sliding window interval over the time domain. Create a sequence of *simplicial complexes* from the graphs and apply *zigzag persistence* to the union zigzag simplicial complexes

remains throughout all of the simplicial complexes. Therefore, we set its death as the last time stamp plus 0.5 and record the persistence pair as $(0, 2.5)$. On the other hand, there is a single loop (a 1-dimensional feature) which is shown only at $K_{0,1}$. For technical reasons, this is drawn as a point with birth time 0.5 (the average of 0 and 1) and death time 1 since it dies entering K_1.

It should be noted that sometimes zigzag persistence results are counter-intuitive to those used to standard persistence interpretation. For example, consider the example of Fig. 3. The first two filtrations each appear to have a circular structure which lasts through the duration of the filtration, however, the zigzag persistence diagram sees the first case as a class living throughout, while the second breaks this class into two individual bars. In this case at least, the issue can be mitigated by including a triangle at the top of the house filling in the short cycle, resulting in both filtrations having only a single point in the persistence diagram representing the loop in the house.

Secondly, we also note that the axes in the zigzag diagram correspond to indexing in the zigzag diagram. This means that a point far from the diagonal means there is a loop structure that is present for a large portion of the index set, rather than being a measurement of size of that same loop.

3 Method

To apply zigzag persistence for studying temporal graphs, we use the pipeline shown in Fig. 4 which we describe more specifically here. A temporal graph is a graph structure that incorporates time information on when edges and/or nodes are present in the graph. We will only be using the case of temporal information attributed to the edges in this work and assume nodes are included as part of an edge-induced subgraph. Thus, our starting data is a graph $G = (V, E)$ where each edge e has a collection of closed intervals $\mathcal{I}_e = \{I_{e,1}, \ldots, I_{e,k_e}\}$ associated to it for when that edge is active. Fitting with the previous section, we fix a collection of times $t_0 \leq t_1 \leq \cdots \leq t_n$ and a choice of window size w. Then $G_i = (V_i, E_i)$ is the graph induced by edges present within a window $[t_i - w/2, t_i + w/2]$. Please note that to ease notation, we are using subscripts i for the graphs, but the graphs should be thought of as encoding information for the interval centered at t_i and thus when computing persistence, the birth and death times are associated to the t_i's rather than the i's. Finally, we define $G_{i,i+1} = G_i \cup G_{i+1}$ to be the union of the two adjacent graphs.

While we can construct a zigzag filtration of graphs

$$G_0 \hookrightarrow G_{0,1} \hookleftarrow G_1 \hookrightarrow G_{1,2} \hookleftarrow G_2 \hookrightarrow \cdots \hookleftarrow G_{n-1} \hookrightarrow G_{n-1,n} \hookleftarrow G_n$$

this is not actually the zigzag we will be using since, as noted earlier, it is not flexible enough to find some of the loop structures needed in the later analysis. So, for any graph G_i with vertex set $V_i \subseteq V$, we let d_i be the shortest path distance on that graph. That is, $d_i : V_i \times V_i \to \mathbb{R}$ where $d_i(u, v)$ is the number of edges needed to get from vertex u to vertex v. Fixing an $r \geq 0$, let $K_i^r := R_r(V_i, d_i)$ be the Rips complex constructed from this distance information. First, we note that K_i^0 is the simplicial complex with only the vertex set V_i. When $r = 1$, K_i^1 is the clique complex of the original graph G_i; where the 1-simplices are the same as that of the graph, but higher dimensional simplices are filled in when available. Then for higher r, we have more and more edges added to the original graph, so this construction is more complicated than simply treating the graph itself as a simplicial complex.

Assuming we have similar notation for the union graph information ($d_{i,i+1}$, $K_{i,i+1}^r$, etc), we form a zigzag filtration by replacing G_i with its Vietoris–Rips complex,

$$K_0^r \hookrightarrow K_{0,1}^r \hookleftarrow K_1^r \hookrightarrow K_{1,2}^r \hookleftarrow K_2^r \hookrightarrow \cdots \hookleftarrow K_{n-1}^r \hookrightarrow K_{n-1,n}^r \hookleftarrow K_n^r.$$

Finally, we compute the p-dimensional homology at each step and from there compute the zigzag persistence diagram. Our code uses the `Dionysus2` package [36] for this last step. Again, be aware that the indices on the zigzag persistence points correspond to the t_i value associated to the given complex.

3.1 Example

In the following simple example shown in Fig. 5, we describe the method in more detail and show how to interpret the resulting zigzag persistence diagram. In this example, we measure the changing structure of a simple 5-node cycle graph as edges are added and removed based on the temporal information. Fixing $r = 1$, the simplicial complexes are exactly the same as the graphs $K_i = G_i$ as there are no cliques in this particular example. The bottom left of Fig. 5(a) shows the times associated to each edge, and the resulting

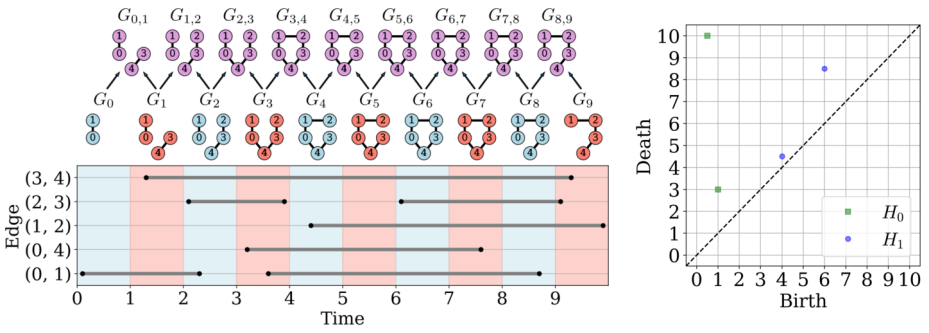


Figure 5 Example zigzag persistence applied to a simple temporal graph with temporal information stored for each edge as intervals. (Left) Edge intervals with sliding windows highlighted (alternating blue-red) with corresponding graphs and union graphs above. (Right) Zigzag persistence diagram for both H_0 and H_1. The birth and death of a feature is encoded as the midpoint of the interval where the event happened

graphs are shown above. In this notation, the subscripts correspond to the interval of time used to build the graphs. Then we have $w = 0.5$, and the centers of intervals are $t_0 = 0.5$, $t_1 = 1.5, \ldots, t_9 = 9.5$. This results in intervals for graph G_i of the form $(i, i+1)$ and intervals for the union graphs $G_{i,i+1}$ as $(i, i+2)$. At the end of the sliding windows, we consider the graph empty and set the death of any remaining homology features as the end time of the last window (i.e., $t = 10$ for this example).

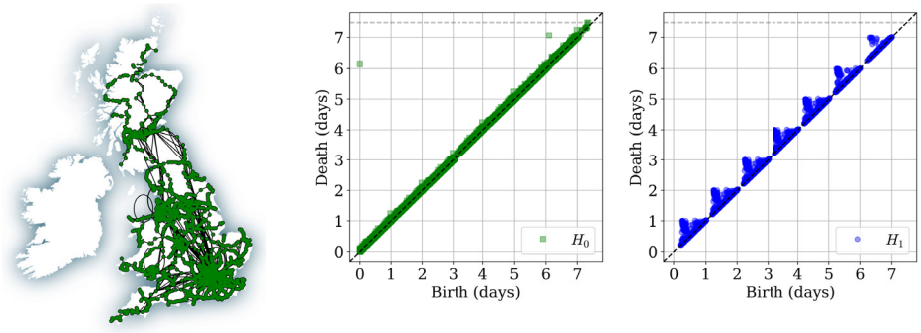
The resulting zigzag persistence diagram is shown in Fig. 5(b). This persistence diagram shows the zero-dimensional and one-dimensional features as H_0 and H_1, respectively. There are two zero-dimensional features at persistence pairs $(1, 3)$ and $(0.5, 10)$. The later represents the connected component which appears at the first graph and lasts throughout the filtration. The other component is the piece consisting of vertices 3 and 4 which appears at union graph $G_{(0,2)}$, thus is associated to a birth at the midpoint of this interval occurring at 1.

The one-dimensional feature (the cycle represented in H_1) is present twice in the persistence diagram. This is due to it first appearing in $G_{(3,5)}$ and then disappearing at $G_{(4,5)}$ with corresponding persistence pair $(4, 4.5)$. The cycle then reappears at $G_{(5,7)}$ and disappears at $G_{(8,9)}$ resulting in persistence pair at $(6, 8.5)$.

This example demonstrates how zigzag persistence captures the changing structure of temporal graphs at multiple dimensions. It is possible to also capture higher-dimensional structures using higher-dimensional homology, although we do not investigate this direction in this work. In particular, it is not clear what higher dimensional homology would represent in the context of data coming from 1-dimensional graph structures.

4 Results

To demonstrate the functionality of zigzag persistence for analyzing temporal graphs, we will use two examples. The first is an analysis of transportation data from Great Britain in Sect. 4.1. The second is a simulated dataset from the Lorenz system that exhibits intermittency, a dynamical system phenomenon where the dynamic state transition from periodic to chaotic in irregular intervals with results in Sect. 4.2. We study this signal using the temporal ordinal partition network framework as described in Sect. 4.2.1. We



(a) Full Rail Travel Network.

(b) Zero-dimensional zigzag persistence.

(c) One-dimensional zigzag persistence.

Figure 6 Zigzag persistence diagrams of the rail transportation network of Great Britain

compare our results for both examples to some standard networks tools to analyze temporal networks. Namely, we will compare two connectivity statistics and three centrality statistics.

The two connectivity statistics analyze the Connected Components (CCs). The first CC statistic is the number of connected components N_{cc}, which provides a simple shape summary of the graph snapshots by understanding the number of disconnected subgraphs. The second statistic is the average size (number of nodes) of the connected components $\bar{S}_{cc}$. This statistic provides insight into how significant the components are for each graph snapshot.

The second statistic type is on centrality measures. The three centrality measures we use are the average and standardized degree centrality $\bar{C}_d$, betweenness centrality $\bar{C}_b$, and closeness centrality $\bar{C}_c$. The degree centrality measures the number of edges connected to a node, the betweenness centrality measures how often a node is used in all possible shortest paths, and the closeness centrality measures how close the node is to all other nodes through the shortest path. For details on the implementation of each centrality measure, we direct the reader to [37].

4.1 Great Britain temporal transportation network

We use temporal networks created from the Great Britain (GB) temporal transportation dataset [27] for the air, rail, and coach transportation methods. This data provides the destinations (nodes) and connections (edges) for public transportation in GB. Additionally, the departure and arrival times are provided to allow a temporal analysis. This temporal data was collected for one week, Monday through Sunday. In this section, we use both the rail and coach data; similar calculations for air data are included in Appendix A. The rail graph constructed without the inclusion of temporal information is shown at left in Fig. 6 where the destinations are overlaid with a GB map outline. Figures for the similar air and coach graphs are included in Appendix A. In all three cases, we set the sliding windows to have width $w = 20$ minutes. Because the average wait time was 7 minutes and 7 seconds with a standard deviation of 7 minutes and 24 seconds from a collected sample [38], this ensures that we retain connectivity. Additionally, we used an overlap of 50% between adjacent windows.

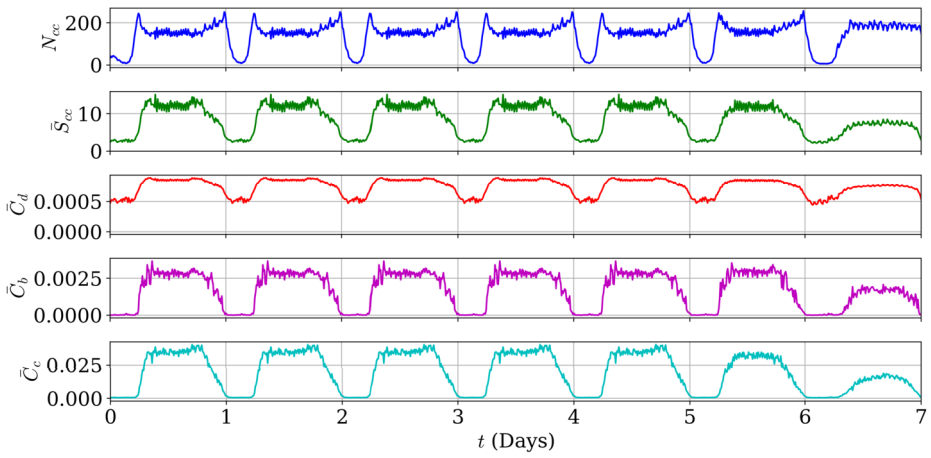


Figure 7 Connectivity and centrality analysis on temporal Great Britain rail network

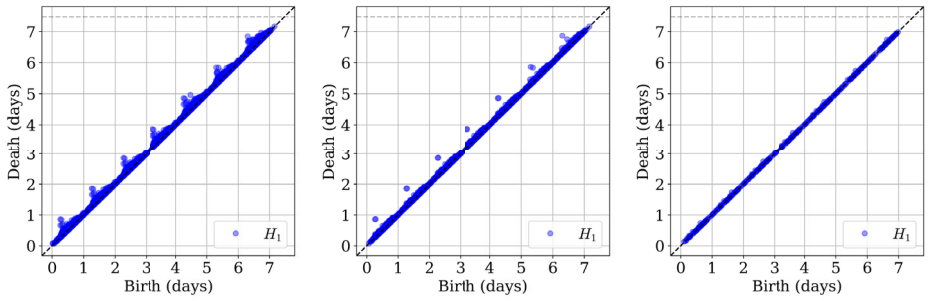
4.1.1 Rail

We first focus on the rail data, for which we use the VR complex with $r = 1$. The 0- and 1-dimensional zigzag diagrams are shown in Fig. 6. The first noticeable feature is that the 1-dimensional diagram has a clear daily pattern of points, implying that there are many loops in the rail network that persist during the day, but become disconnected in the evenings when the trains are no longer running. However, there is still an overarching connectivity of the rail network seen during the week, as noted by the high persistence point at approximately $(0, 6)$ in the 0-dimensional diagram. We note that this does not mean that the entire graph is connected for the entirety of the days Monday through Saturday, but instead that there is some connected area that remains running even overnight. We suspect this would be tied to some of the more urban areas such as London, but further work would be needed to implement code to obtain generators of the zigzag diagram. This lower connectivity on Saturday and Sunday can also be seen in the additional sparsity in the day 6 and day 7 peaks in the 1-dimensional diagram.

We compare these interpretations to the standard graph analysis tools [3]. The standard centrality and connectivity statistics for the same data are shown in Fig. 7. We can clearly see that there is a daily periodicity in the data from these standard tools as was seen in the zigzag persistence. Specifically, all the connectivity and centrality measures increase during peak travel hours. What is lost, and which can be augmented with the zigzag viewpoint, is the ability to connect the clustering and centrality measures from one time step to the next. There is no way to determine, say, from the N_{cc} graph that there is some portion of the graph that remains connected for 6 out of the 7 days. For this reason, we believe that the zigzag setting can be used along side the standard measures in order to strengthen analysis of given temporal graph information.

4.1.2 Coach

Now, we use the coach network data to illustrate using the VR complexes with different r values. In Fig. 8, 1-dimensional zigzag persistence diagrams are displayed. The leftmost diagram is computed using the graphs directly as input to the zigzag persistence without taking the VR complex. The second and third were obtained using the graph and VR complexes with $r = 1$ and $r = 2$ respectively.



(a) One-dimensional zigzag persistence using original graph.

(b) One-dimensional zigzag persistence using VR complex with $r = 1$.

(c) One-dimensional zigzag persistence using VR complex with $r = 2$.

Figure 8 Zigzag persistence diagrams of the coach transportation network of Great Britain

Both diagrams (a) and (b) show periodic structure during the day which is to be expected as the number of buses running diminishes overnight. However, in diagram (b) we can clearly see a daily trend. While unfortunately difficult to see on the diagram, this is in fact points in the diagram overlaid, implying we have two loops of more than three edges that persist during the day. If the loops had length 3, they would be filled in by triangles when replacing the graph with the VR complex. This further implies that the noise seen in (a) is the existence of a great deal of triangles; not surprising in a highly connected local system like a bus network. But then, we can also compare persistence diagram Fig. 8(b) to the analogous Fig. 6(c) from the rail network. The more locally connected nature of the bus system means that many of these small loops are not included when using the VR complex; however the rail system is more spread out, thus resulting in longer loops that are not filled in by a choice of $r = 1$.

Finally, we observe that in the persistence diagram of Fig. 8(c), where there is very little of the periodic structure remaining. From this, we can assume that the many small loops seen in the $r = 1$ diagram have been filled in by our choice of $r = 2$, thus meaning that the loops are of length at most 6 (See the text surrounding Fig. 6 of [30] for a discussion of lifetime of loops by length in this setting).

4.2 Temporal ordinal partition network for intermittency detection

In this section we apply the zigzag methods to study time series data encoded using complex networks; namely, the ordinal partition network. Ordinal partition networks [29] are a graph representation of time series data based on permutation transitions. As such, they encapsulate the state space structure of the underlying system. While we only use the ordinal partition network in this work, there are several other transitional complex networks from time-series data that a similar analysis could be done. These include k-nearest-neighbors [39], epsilon-recurrence [40], and coarse-grained state-space networks [41, 42]. We begin by giving a brief introduction to the construction of the network from time series data, followed by the results of analysis using zigzag persistence.

4.2.1 Temporal ordinal partition network

Given a time series $x = [x_0, x_1, x_2, \ldots, x_n]$ for a sequence of times $t = [t_0, \ldots, t_n]$, the ordinal partition network is formed by first generating a sequence of permutations using a

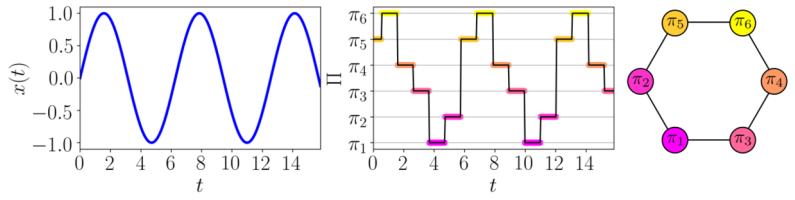


Figure 9 Example formation of an ordinal partition network for a sinusoidal signal $x(t) = \sin(t)$ with permutations of dimension $n = 3$. The resulting permutation sequence Π shows how these permutations transition, which is captured by the ordinal partition network on the right

fixed permutation dimension m and delay τ. We generate a sequence of permutations by assigning each vector embedding

$$v_i = [x_i, x_{i+\tau}, x_{i+2\tau}, \ldots, x_{i+(m-1)\tau}] = \big[v_i(0), v_i(1) \ldots, v_i(m-1)\big] \tag{4}$$

to one of the $m!$ possible permutations. We give an arbitrary ordering to the permutations for labeling purposes, and assign the permutation $\pi_j = [\pi_j(0), \ldots \pi_j(m-1)] \in \mathbb{Z}^m$ based on the ordinal pattern of v_i such that $v_i(\pi_j(0)) \leq v_i(\pi_j(1)) \leq \cdots \leq v_i(\pi_j(m-1))$.

Using the chronologically ordered sequence of permutations Π, we can form a graph $G(E, V)$ by setting the vertices V as all permutations used and edges for transitions from π_a to π_b with $a, b \leq m!$ and $a \neq b$ (no self-loops). We will not add weight or directionality to the graph for this formation. However, we will track the index i for the corresponding time x_i when the edge is activated as the temporal data for the graph.

In Fig. 9 we demonstrate the ordinal partition network formation procedure for a simple example signal as $x(t) = \sin(t)$, where $t \in [0, 15]$ sampled at a rate of $f_s = 25$ Hz. Using the method of multi-scale permutation entropy we selected $\tau = 52$ and set $n = 3$ for demonstrative purposes. The corresponding permutations to delay embedding vector v_i is shown as a sequence Π in the middle subfigure of Fig. 9. This sequence captures the periodic nature of the signal which is then summarized as the ordinal partition network on the right with each permutation as a vertex and edges added for permutation transitions in Π. For more details and examples of the ordinal partition network, we direct the reader to [29, 30].

4.2.2 Ordinal partition network results

Using a sliding window technique, we can represent ordinal partition networks as temporal graphs. However, instead of each edge having a set of intervals associated with it as in the example in Sect. 3.1, they have time instances where the edge is active based on when a transition between unique permutations occur. For example, the transition from π_i to π_{i+1} occurring at time t_j would be active for the moment in time t_j. If the sliding window contains an edge's activation instance, we add that edge to the sliding window graph.

We will show how this procedure can be used to detect chaotic and periodic windows in a signal exhibiting intermittency (i.e., the irregular transitions from periodic to chaotic dynamics). The signal used here is the z solution to the simulated Lorenz system defined as

$$\frac{dx}{dt} = \sigma(y - x), \qquad \frac{dy}{dt} = x(\rho - z) - y, \qquad \frac{dz}{dt} = xy - \beta z \tag{5}$$

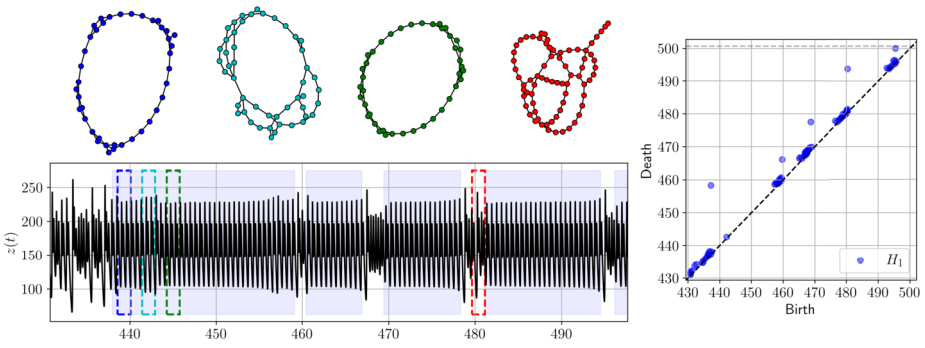


Figure 10 On the left, the z solution of the intermittent Lorenz system described in Eq. (5) is shown, along with four different graphs obtained from the corresponding ordinal partition networks in the windows of matching color. On the right, the one-dimensional zigzag persistence diagram

with system parameters $\sigma = 10$, $\beta = 8/3$, and $\rho = 166.18$ for a response with type 1 intermittency [43]. We simulated the system with a sampling rate of 100 Hz for 500 seconds with only the last 70 seconds used. To construct the ordinal partition network, we choose $m = 6$ and τ using the multi-scale permutation entropy method as suggested in [44]. We set the sliding windows for generating graph snapshots to have a width of 5τ and 80% overlap between adjacent windows.

The resulting signal $z(t)$ from simulating the Lorenz system in Eq. (5) is shown in Fig. 10, with four examples of ordinal partition networks generated at several window locations. These sample graph snapshots show that the structure of the ordinal partition network significantly changes depending on the dynamic state of the window's time-series segment. At right in the figure, we see the 1-dimensional zigzag diagram for the data. Of particular note is that there are several high persistence points in the diagram. Recalling that the coordinates of these points are associated to time of appearance of loops rather than size of the loops themselves, we have marked the regions of the times series associated to these points in blue at left. From visual inspection, it appears that these regions correspond to periodic behavior in the time series. The remaining chaotic windows characteristically have many low-lifetime persistence pairs seen close to the diagonal in the persistence diagram. This is in line with the results in [30] that showed ordinal partition networks from chaotic signals tend to have persistence diagrams with many features in H_1 when compared to their periodic counterpart. The fact that this labeling is done with only the user's choice of threshold for what is considered a high-persistence point makes this a potentially exciting avenue for future work to understand how it can be used in the case of labeling intermittency in time series.

Additionally, note in the persistence diagram in Fig. 10 the point near the diagonal, at around second 442, representing a loop in the network that quickly disappears, as seen also in the first three ordinal partition networks shown at left of Fig. 10. This tells us that low-lifetime loops may show up in the middle of periodic regions while the main periodic trend is preserved, confirming that we can still trust the coordinates of high persistence points as the bounds of periodic regions. Thus, in general, we can use the persistence diagram to identify periodic regions by looking for intervals where there are persistent loops over relatively long periods and very few (or no) other shorter-living loops, which is easily identified from the persistent points coordinates.

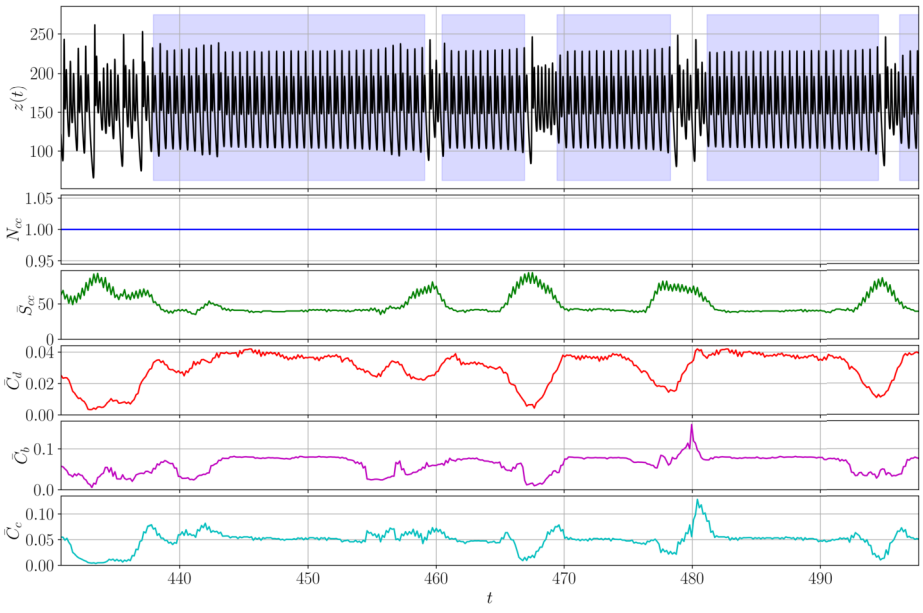
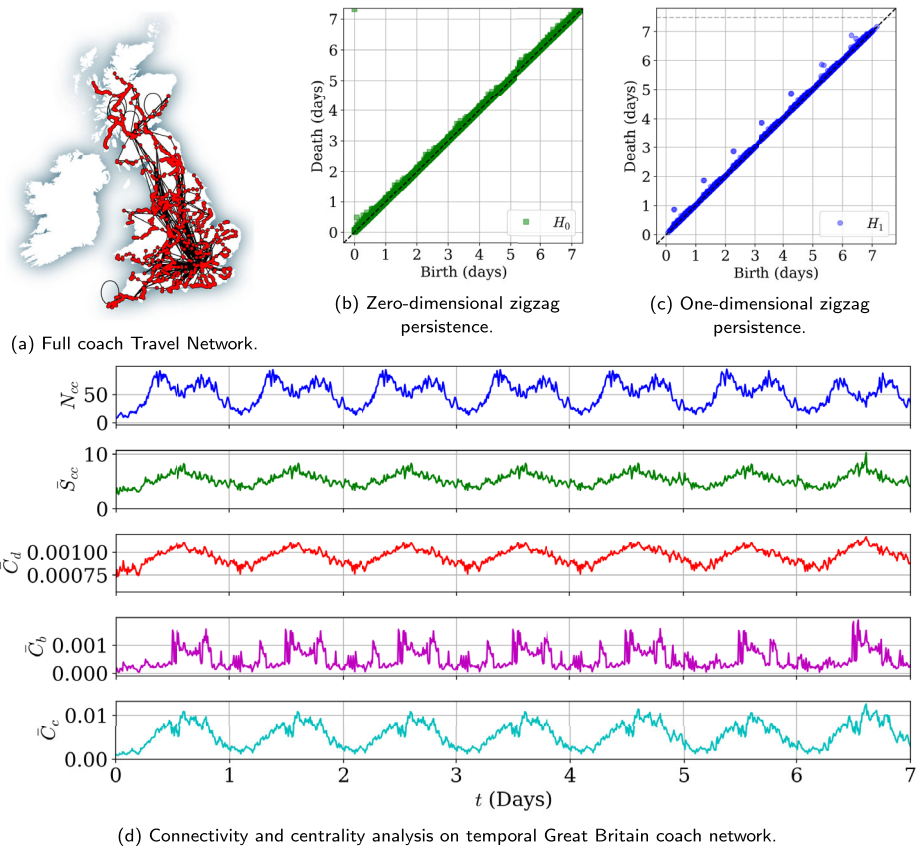


Figure 11 Connectivity and centrality analysis on temporal ordinal partition network with periodic regions as labeled with the zigzag persistence points highlighted in blue

To compare with the standard tools from temporal network analysis, we show the connectivity and centrality measures of the graph snapshots in Fig. 11. The number of components N_{cc} is constant due to the nature of the ordinal partition network, where the sequence of permutation transitions creates a chain of connected edges. As such, there is no structural information in the number of components. However, the size of the components does increase during the chaotic windows. This increase is due to, in general, more unique permutations and thus nodes used in a chaotic signal compared to periodic. Of the centrality statistics, only the average closeness centrality shows an apparent increase during chaotic regions. The increase in centrality is most likely due to the chaotic regions causing a more highly connected graph as demonstrated in the chaotic window and corresponding network of Fig. 11. While these statistics do provide some insight into the changing dynamics, they do not show how the higher-dimensional structure of the graph evolves through the sliding windows and graph snapshots, in contrast to zigzag persistence.

5 Conclusion

In this work we investigated one method for applying zigzag persistence to temporal graphs and discussed interpretation in several settings. We treated these graphs as inputs to create a metric space, and then studied the zigzag constructed by the changing graph over time for a fixed connectivity parameter. Zigzag persistence provides a unique perspective when studying the evolving structure of a temporal graph by tracking the standard lower-dimensional features (e.g., connected components), but also higher-dimensional features (e.g., loops and voids) through a sequence of simplicial complexes. This allows for an understanding of the evolving topology of a temporal graph which can be used in addition to more standard techniques for their analysis. We showed interpretation for these tools on two examples: the Great Britain transportation network and temporal ordinal partition networks. Our results showed that the informative zero and one-dimensional



(a) Full coach Travel Network.

(b) Zero-dimensional zigzag persistence.

(c) One-dimensional zigzag persistence.

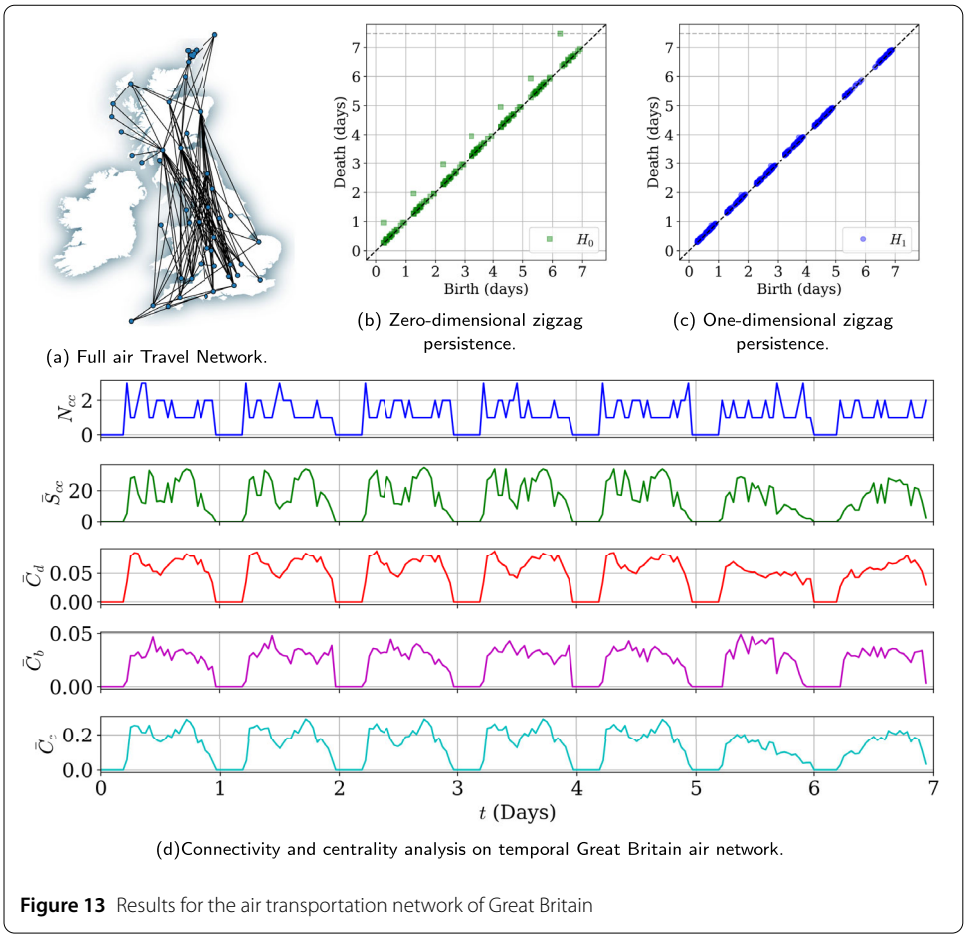
(d) Connectivity and centrality analysis on temporal Great Britain coach network.

Figure 12 Results for the coach transportation network of Great Britain

zigzag persistence provided insights into the structure of the temporal graph that were not easily gleaned from standard centrality and connectivity statistics.

It should be noted that the work here is essentially functioning as a more computationally tractable version of what we would like to do, namely two-parameter persistence. Because multiparameter persistence presents mathematical barriers to simplified representations in the spirit of persistence diagrams [35], many other workarounds have sprung up. These include vineyards [45], where we study a one parameter family of persistence diagrams; or CROCKER plots [46, 47], where we can study a similarly evolving Betti curve. There has been previous work investigating the use of zigzag persistence in graph based applications in the broader theoretical framework of *formigrams* [48, 49] with additional results on stability [50]. These include applications to dynamic metric spaces [51, 52] and brain networks [53], although this setting is restricted to 0-dimensional persistence due to its tight connection to time-varying clustering. Even though in theory the running time of zigzag persistence should be similar to standard persistence [54], in practice it has not seen the flurry of optimizations available in the regular case [55, 56]. However, recent work [57–59] promises substantial improvements in the potentially available code, which should further make the tools discussed in this paper more accessible to a wide array of data sets.

We believe zigzag persistence could also be leveraged to study other temporal graphs including flock behavior models (e.g., Viscsek model) and the emergence of coordinated



(a) Full air Travel Network.

(b) Zero-dimensional zigzag persistence.

(c) One-dimensional zigzag persistence.

(d)Connectivity and centrality analysis on temporal Great Britain air network.

Figure 13 Results for the air transportation network of Great Britain

motion, power grid dynamics with the topological characteristics of a cascade failures, and supplier-manufacture networks through the effects of trade failures on production and consumption. Future work could involve an analysis on deciding an optimal window size and overlap, a method to incorporate edge weight and directionality, and temporal information on both the nodes and edges. It would also be worth investigating higher-dimensional features (e.g., voids through H_2), although more likely because of the 1-dimensional structure of the input data, it is less clear what sorts of behavior might arise in the form of interesting behavior in the zigzag diagrams.

Additionally, the method presented here is by no means the only way to incorporate the input temporal graph in a form that could generate a zigzag complex. We direct the interested reader to [26] for a large collection of possible 1-parameter filtrations that could be generated from a fixed graph. Utilizing any of these for some fixed filtration parameter evolving over the temporal graph information could yield a zigzag diagram whose structure may be useful for additional interpretation in other applications.

Appendix A: Great Britain transportation networks results: air and coach travel

For completeness, we include figures and persistence diagrams for the Great Britain transportation data not included in Sect. 4.1.

A.1 Coach travel network analysis

Figure 12 shows the full network for the coach data, and the 0- and 1-dimensional zigzag diagrams in the top row. The standard statistics for temporal graphs are shown in the bottom row. See Sect. 4.1.2 for a full discussion of the coach data.

A.2 Air travel network analysis

Figures 13 shows the zigzag diagrams and standard measures for the air travel data. Like rail and coach, the air travel network clearly has regular daily structure as seen in both diagrams. Interestingly, here we have a daily persistence point in 0-dimensions with splits in the middle, meaning that the air network does not retain any connected regions overnight. This, again, is not surprising as there is more tendency for airports to have absolutely no flights overnight as opposed to late-night bus systems. The lack of high persistence points in H_1 suggests that there are not even many small loops in the network at any given time, which might be caused by having considerably fewer edges in this network than the others.

Acknowledgements
We thank two anonymous reviewers for substantial feedback which greatly improved the state of the paper.

Funding
This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-22-1-0007. The work of EM and DM was supported in part by NSF grant CCF-2106578.

Availability of data and materials
The code used in this paper is freely available in the Python open-source package `teaspoon` [60].

Declarations

Competing interests
The authors declare that they have no competing interests.

Author contributions
AM and DM ran the analysis. FK conceptualized the problem, and supervised the work. EM provided mathematical expertise on zigzag interpretation. All authors wrote, read and approved the final manuscript.

Author details
[1]Department of Mechanical Engineering, Michigan State University, 428 S. Shaw Lane, 48824, East Lansing, USA. [2]Department of Computational Mathematics, Science and Engineering, Michigan State University, 428 S. Shaw Lane, 48824, East Lansing, USA. [3]Department of Mathematics, Michigan State University, 619 Red Cedar Road, 48824, East Lansing, USA.

Publisher's Note

Received: 18 May 2022 Accepted: 6 February 2023 Published online: 02 March 2023

References

1. Porter MA (2020) Nonlinearity + networks: a 2020 vision. In: Emerging frontiers in nonlinear science. Springer, Berlin, pp 131–159. https://doi.org/10.1007/978-3-030-44992-6_6
2. Holme P, Saramäki J (eds) (2013) Temporal networks Springer, Berlin. https://doi.org/10.1007/978-3-642-36461-7
3. Holme P (2015) Modern temporal network theory: a colloquium. Eur Phys J B 88(9):234. https://doi.org/10.1140/epjb/e2015-60657-4
4. Skyrms B, Pemantle R (2000) A dynamic model of social network formation. Proc Natl Acad Sci 97(16):9340–9346. https://doi.org/10.1073/pnas.97.16.9340
5. Husein I, Mawengkang H, Suwilo S, Mardiningsih (2019) Modeling the transmission of infectious disease in a dynamic network. J Phys Conf Ser 1255(1):012052. https://doi.org/10.1088/1742-6596/1255/1/012052
6. Xu M, Radhakrishnan S, Kamarthi S, Jin X (2019) Resiliency of mutualistic supplier-manufacturer networks. Sci Rep 9(1):13559. https://doi.org/10.1038/s41598-019-49932-1
7. Schäfer B, Witthaut D, Timme M, Latora V (2018) Dynamically induced cascading failures in power grids. Nat Commun 9(1):1975. https://doi.org/10.1038/s41467-018-04287-5

8. David Boyce BR (2012) Modeling dynamic transportation networks. Springer, Berlin
9. Enright J, Kao RR (2018) Epidemics on dynamic networks. Epidemics 24:88–97. https://doi.org/10.1016/j.epidem.2018.04.003
10. Nuss P, Graedel TE, Alonso E, Carroll A (2016) Mapping supply chain risk by network analysis of product platforms. Sustain Mater Technol 10:14–22. https://doi.org/10.1016/j.susmat.2016.10.002
11. Soltan S, Mazauric D, Zussman G (2014) Cascading failures in power grids. In: Proceedings of the 5th international conference on future energy systems. ACM, New York. https://doi.org/10.1145/2602044.2602066
12. Bast H, Delling D, Goldberg A, Müller-Hannemann M, Pajor T, Sanders P, Wagner D, Werneck RF (2015) Route planning in transportation networks. arXiv:1504.05140
13. Sugishita K, Asakura Y (2020) Vulnerability studies in the fields of transportation and complex networks: a citation network analysis. Public Transp 13(1):1–34. https://doi.org/10.1007/s12469-020-00247-9
14. Hackl J, Adey BT (2019) Estimation of traffic flow changes using networks in networks approaches. Appl Netw Sci 4(1):28. https://doi.org/10.1007/s41109-019-0139-y
15. Chen X, Zhang C, Ge B, Xiao W (2016) Temporal query processing in social network. J Intell Inf Syst 49(2):147–166. https://doi.org/10.1007/s10844-016-0437-0
16. Huang S, Fu AW-C, Liu R (2015) Minimum spanning trees in temporal graphs. In: Proceedings of the 2015 ACM SIGMOD international conference on management of data. ACM, New York. https://doi.org/10.1145/2723372.2723717
17. Wang Y, Yuan Y, Ma Y, Wang G (2019) Time-dependent graphs: definitions, applications, and algorithms. Data Sci Eng 4(4):352–366. https://doi.org/10.1007/s41019-019-00105-0
18. Borgatti SP (2005) Centrality and network flow. Soc Netw 27(1):55–71. https://doi.org/10.1016/j.socnet.2004.11.008
19. Crawford J, Milenković T (2018) ClueNet: clustering a temporal network based on topological similarity rather than denseness. PLoS ONE 13(5):0195993. https://doi.org/10.1371/journal.pone.0195993
20. You J, Hu C, Kamigaito H, Funakoshi K, Okumura M (2021) Robust dynamic clustering for temporal networks. In: Proceedings of the 30th ACM international conference on information & knowledge management. ACM, New York. https://doi.org/10.1145/3459637.3482473
21. Moriano P, Finke J, Ahn Y-Y (2019) Community-based event detection in temporal networks. Sci Rep 9(1):4358. https://doi.org/10.1038/s41598-019-40137-0
22. Kempe D, Kleinberg J, Kumar A (2002) Connectivity and inference problems for temporal networks. J Comput Syst Sci 64(4):820–842. https://doi.org/10.1006/jcss.2002.1829
23. Dey TK, Wang Y (2021) Computational topology for data analysis. Cambridge University Press, Cambridge
24. Munch E (2017) A user's guide to topological data analysis. J Learn Anal 4(2):47–61. https://doi.org/10.18608/jla.2017.42.6
25. Carlsson G, de Silva V (2010) Zigzag persistence. Found Comput Math 10(4):367–405. https://doi.org/10.1007/s10208-010-9066-0
26. Aktas ME, Akbas E, Fatmaoui AE (2019) Persistence homology of networks: methods and applications. Appl Netw Sci 4(1):61. https://doi.org/10.1007/s41109-019-0179-3
27. Gallotti R, Barthelemy M (2015) The multilayer temporal network of public transport in Great Britain. Sci Data 2(1):140056. https://doi.org/10.1038/sdata.2014.56
28. Tymochko S, Munch E, Khasawneh FA (2020) Using zigzag persistent homology to detect Hopf bifurcations in dynamical systems. Algorithms 13(11):278. https://doi.org/10.3390/a13110278
29. McCullough M, Small M, Stemler T, Iu HH-C (2015) Time lagged ordinal partition networks for capturing dynamics of continuous dynamical systems. Chaos, Interdiscip J Nonlinear Sci 25(5):053101. https://doi.org/10.1063/1.4919075
30. Myers A, Munch E, Khasawneh FA (2019) Persistent homology of complex networks for dynamic state detection. Phys Rev E 100(2):022314. https://doi.org/10.1103/physreve.100.022314
31. Myers A, Khasawneh FA, Munch E (2022) Topological signal processing using the weighted ordinal partition network. arXiv:2205.08349
32. Hatcher A (2002) Algebraic topology. Cambridge University Press, Cambridge
33. Munkres JR (1993) Elements of algebraic topology. Addison-Wesley, Reading
34. Oudot SY (2015) Persistence theory: from quiver representations to data analysis. AMS mathematical surveys and monographs, vol 209. Am. Math. Soc., Providence
35. Carlsson G, de Silva V, Morozov D (2009) Zigzag persistent homology and real-valued functions. In: Proceedings of the 25th annual symposium on computational geometry—SCG 09. ACM, New York. https://doi.org/10.1145/1542362.1542408
36. Morozov D (2019). Dionysus2. http://www.mrzv.org/software/dionysus2/
37. Landherr A, Friedl B, Heidemann J (2010) A critical review of centrality measures in social networks. Bus Inf Syst Eng 2(6):371–385. https://doi.org/10.1007/s12599-010-0127-3
38. van Hagen M (2011) Waiting experience at train stations. PhD thesis, University of Twente
39. Khor A, Small M (2016) Examining k-nearest neighbour networks: superfamily phenomena and inversion. Chaos, Interdiscip J Nonlinear Sci 26(4):043101. https://doi.org/10.1063/1.4945008
40. Jacob R, Harikrishnan KP, Misra R, Ambika G (2019) Weighted recurrence networks for the analysis of time-series data. Proc R Soc A, Math Phys Eng Sci 475(2221):20180256. https://doi.org/10.1098/rspa.2018.0256
41. Small M, Zhang J, Xu X (2009) Transforming time series into complex networks. In: Lecture notes of the institute for computer sciences, social informatics and telecommunications engineering. Springer, Berlin, pp 2078–2089. https://doi.org/10.1007/978-3-642-02469-6_84
42. Small M (2013) Complex networks from time series: capturing dynamics. In: 2013 IEEE international symposium on circuits and systems (ISCAS2013). IEEE Press, New York. https://doi.org/10.1109/iscas.2013.6572389
43. Pomeau Y, Manneville P (1980) Intermittent transition to turbulence in dissipative dynamical systems. Commun Math Phys 74(2):189–197. https://doi.org/10.1007/bf01197757
44. Myers A, Khasawneh FA (2020) On the automatic parameter selection for permutation entropy. Chaos, Interdiscip J Nonlinear Sci 30(3):033130. https://doi.org/10.1063/1.5111719

45. Cohen-Steiner D, Edelsbrunner H, Morozov D (2006) Vines and vineyards by updating persistence in linear time. In: Proceedings of the twenty-second annual symposium on computational geometry—SCG '06, p 119. https://doi.org/10.1145/1137856.1137877
46. Ulmer M, Ziegelmeier L, Topaz CM (2019) A topological approach to selecting models of biological experiments. PLoS ONE 14(3):0213679. https://doi.org/10.1371/journal.pone.0213679
47. Güzel I, Munch E, Khasawneh FA (2022) Detecting bifurcations in dynamical systems with CROCKER plots. Chaos, Interdiscip J Nonlinear Sci 32(9):093111. https://doi.org/10.1063/5.0102421
48. Kim W, Mémoli F (2022) Extracting persistent clusters in dynamic data via Möbius inversion. arXiv:1712.04064
49. Kim W, Mémoli F, Stefanou A (2019) Interleaving by parts: join decompositions of interleavings and join-assemblage of geodesics. https://doi.org/10.48550/ARXIV.1912.04366. arXiv:1912.04366v4
50. Kim W, Mémoli F, Smith Z (2020) Analysis of dynamic graphs and dynamic metric spaces via zigzag persistence. In: Baas NA, Carlsson GE, Quick G, Szymik M, Thaule M (eds) Topological data analysis. Springer, Cham, pp 371–389
51. Kim W (2020) The persistent topology of dynamic data. PhD thesis, The Ohio State University
52. Kim W, Mémoli F (2021) Spatiotemporal persistent homology for dynamic metric spaces. Discrete & Computational Geometry 66(3):831–875. https://doi.org/10.1007/s00454-019-00168-w
53. Chowdhury S, Dai B, Mémoli F (2018) The importance of forgetting: limiting memory improves recovery of topological characteristics from neural data. PLoS ONE 13(9):1–20. https://doi.org/10.1371/journal.pone.0202561
54. Milosavljevic N, Morozov D, Skraba P (2011) Zigzag persistent homology in matrix multiplication time. In: Proceedings of the 27th annual symposium on computational geometry
55. Bauer U (2021) Ripser: efficient computation of Vietoris–rips persistence barcodes. J Appl Comput Topol. https://doi.org/10.1007/s41468-021-00071-5
56. Otter N, Porter MA, Tillmann U, Grindrod P, Harrington HA (2017) A roadmap for the computation of persistent homology. EPJ Data Sci 6(1):17. https://doi.org/10.1140/epjds/s13688-017-0109-5
57. Dey TK, Hou T (2021) Computing zigzag persistence on graphs in near-linear time. arXiv:2103.07353
58. Dey TK, Hou T (2022) Fast computation of zigzag persistence. arXiv:2204.11080
59. Dey TK, Hou T (2021) Updating barcodes and representatives for zigzag persistence. arXiv:2112.02352
60. Myers AD, Yesilli M, Tymochko S, Khasawneh F, Munch E (2020) Teaspoon: a comprehensive python package for topological signal processing. In: NeurIPS 2020 workshop on topological data analysis and beyond. https://openreview.net/forum?id=qUoVqrIcy2P