Persistent homology of complex networks for dynamic state detection

Audun Myers

Department of Mechanical Engineering, Michigan State University, East Lansing, Michigan 48824, USA

Elizabeth Munch

Department of Computational Mathematics, Science, and Engineering and Department of Mathematics,
Michigan State University, East Lansing, Michigan 48824, USA

Firas A. Khasawneh

Department of Mechanical Engineering, Michigan State University, East Lansing, Michigan 48824, USA

In this paper we develop an alternative topological data analysis (TDA) approach for studying graph representations of time series of dynamical systems. Specifically, we show how persistent homology, a tool from TDA, can be used to yield a compressed, multi-scale representation of the graph that can distinguish between dynamic states such as periodic and chaotic behavior. We show the approach for two graph constructions obtained from the time series. In the first approach the time series is embedded into a point cloud which is then used to construct an undirected k-nearest-neighbor graph. The second construct relies on the recently developed ordinal partition framework. In either case, a pairwise distance matrix is then calculated using the shortest path between the graph's nodes, and this matrix is utilized to define a filtration of a simplicial complex that enables tracking the changes in homology classes over the course of the filtration. These changes are summarized in a persistence diagram's two-dimensional summary of changes in the topological features. We then extract existing as well as new geometric and entropy point summaries from the persistence diagram and compare to other commonly used network characteristics. Our results show that persistence-based point summaries yield a clearer distinction of the dynamic behavior and are more robust to noise than existing graph-based scores, especially when combined with ordinal graphs.

I. INTRODUCTION

There has been extensive work on understanding the behavior of the underlying dynamical system given only a time series [1,2]. The revolutionary work of Takens [3] extended by Sauer *et al.* [4] showed that, given most choices of parameters, the state space of the dynamical system can be reconstructed through the Takens embedding. Computationally, this arises as the following procedure. Given a time series $[x_1, \ldots, x_n]$, a choice of dimension d and time lag τ give rise to a point cloud $\chi = \{\mathbf{x}_i := (x_i, x_{i+\tau}, \ldots, (x_{i+(d-1)\tau})\} \subset \mathbb{R}^d$. Then the goal is to analyze this point cloud, which really is a sampling of the full state space, in a way that the dynamics can be understood. Of course, for practical purposes, not all parameter choices are equally desirable. While some effort has gone into mathematical justification of "best" choices [5], we are largely left with heuristics that work quite well in practice [6,7].

A first method for analyzing this point cloud arose in the form of a recurrence plot [8,9]. Fixing ϵ, this is a binary, symmetric matrix $R = R(\epsilon)$ where R_{ij} is 1 if $\|\mathbf{x}_i - \mathbf{x}_j\| \leqslant \epsilon$, and 0 otherwise. Of course, this can be equivalently viewed as the adjacency matrix of a network [10], often called the ϵ-recurrence network in this literature [11–14]. From this observation, a large literature grew on methods to convert time series into networks; see Donner *et al.* [15] for an extensive survey.

In this paper, we focus on two of these options. First, given the point cloud χ, we can construct the (undirected) k-nearest-neighbor graph, commonly called the k-NN graph. This is built by adding a vertex to represent each $\mathbf{x}_i \in \chi$, and for each $\mathbf{x}_i$, adding an edge to the k closest points $\mathbf{x}_j \in \chi$. This construction, and in particular the investigation of motifs in the resulting graph, has been extensively studied [16–18].

The second network construction method we work with is the recently developed ordinal partition network [19,20]. It can be viewed as a special case of the class of *transition networks* built from time series data, where vertices represent subsets of the state space, and edges are included based on temporal succession. This construction arose as a generalization of the concept of permutation entropy [21]. The basic idea of the construction is to replace each $\mathbf{x} = (x_1, \ldots, x_d) \in \chi \subset \mathbb{R}^d$ with a permutation π of the set $\{1, \ldots, d\}$ so that $\pi(i) = j$ if x_j is the ith entry in the sorted order of the coordinates. That is, π satisfies $x_{\pi(1)} \leqslant x_{\pi(2)} \leqslant \cdots \leqslant x_{\pi(d)}$. Then we build a graph with vertex set equal to the set of encountered permutations, and an edge is included if the ordered point cloud passes from one permutation to the other [22]. Utilizing the transition network vantage point, each permutation π represents a subspace of $\mathbb{R}^d$ given by the intersection of $\binom{d}{2}$ inequalities, and an edge is included based on passing from one of these subspaces to the other in one time step. What
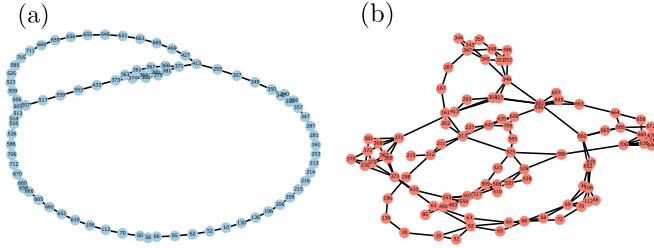
FIG. 1. Comparison between ordinal partition networks generated from x solution of the Rössler system for both (a) periodic and (b) chaotic time series.

makes this construction particularly useful is its robustness to noise [23]. See Ref. [24] for a more extensive introduction.

Many have observed qualitatively that these networks encode the structure of the underlying system [15]. In particular, periodic time series tend to create networks with overarching circular structure, while those arising from chaotic systems have more in common with a hairball (see, e.g., Fig. 1). However, quantification of this behavior is lacking. Much of the literature to date has focused on using standard quantification methods from network theory such as local measures like degree, closeness, and betweenness centrality, or the local clustering coefficient. Global measures are also used, e.g., the global clustering coefficient, transitivity, and average path length. However, these measures can only do so much to measure the overarching structure of the graph.

It was for this reason that topological data analysis (TDA) [25–29] has proven to be quite useful for time series analysis. TDA is a collection of methods arising from the mathematical field of algebraic topology [30,31], which provide concise, quantifiable, comparable, and robust summaries of the shape of data. The main observation is that we can encode higher-dimensional structure than the one-dimensional information of a network by passing to simplicial complexes. Like graphs, simplicial complexes are combinatorial objects with vertices and edges, but they also allow for higher-dimensional analogs like triangles, tetrahedra, etc. To date, the interaction of time series analysis with TDA has focused on a generalization of the ϵ-recurrence network called a Rips complex and its approximation, the witness complex. The Rips complex for parameter ϵ includes a simplex $\sigma = \{\mathbf{y}_0, \dots, \mathbf{y}_k\}$ if and only if $\|\mathbf{y}_i - \mathbf{y}_j\| \leqslant \epsilon$ for all i, j. That is, it is the largest simplicial complex which has the ϵ-recurrence graph as its 1-skeleton.

Unlike the time series analysis literature, where one works hard to find the perfect ϵ to construct a single network, TDA likes to work with the Rips complex over all scales in a construction called a filtration. Then, one can analyze the structure of the overall shape by looking at how long features of interest persist over the course of this filtration. One particularly useful tool for this analysis is one-dimensional (1D) persistent homology [32,33], which encodes how circular structures persist over the course of a filtration in a topological signature called a persistence diagram. This and its variants have been quite successful in applications, particularly for the analysis of periodicity [34–41], including for parameter selection [42,43], data clustering [44], machining dynamics [45–49], gene regulatory systems [50,51], financial

data [52–54], wheeze detection [55], sonar classification [56], video analysis [57–59], and annotation of song structure [60,61].

Unfortunately, while persistence diagrams are powerful tools for summarizing structure, their geometry is not particularly amenable to the direct application of standard statistical or machine learning techniques [62–64]. To circumvent the problem, a common trick to deal with persistence diagrams particularly when we are interested in classification tasks is to choose a method for *featurizing* the diagrams, that is, constructing a map from persistence diagrams into Euclidean space $\mathbb{R}^d$ via some method that preserves enough of the structure of the persistence diagrams to be reasonably useful. Many of these exist in the literature [65–76]; however, in this work, we focus on the simplest of these realizations, namely, point summaries of persistence diagrams which extract a single number for each diagram to be used as its representative. One summary which we use in this paper is persistent entropy. This was defined by Chintakunta *et al.* [77], and later Rucco *et al.* [78,79] proved that the construction is continuous. This construction, a modification of Shannon entropy, has found use in several applications [80–82].

In this paper we move away from the standard application of TDA to time series analysis (namely, the combination of the Rips complex with 1D persistence) to implement the following new pipeline and use it to differentiate between chaotic and periodic systems (see Fig. 2). Given a time series, determine a good choice of embedding parameters, and use these to build an embedding of the time series. Then, obtain a graph either by constructing the k-nearest-neighbor graph for the points of the embedding, or by building the ordinal partition network. Construct a filtration of a simplicial complex using this information, compute its persistence diagram, then return one of several point summaries of the diagram. We show experimentally on both synthetic and real data that this pipeline, particularly using persistent entropy, is quite good at differentiating between chaotic and periodic time series. Further, the resulting simplicial complexes used are considerably smaller than those utilized in the Rips complex setting, providing the potential for faster running times.

II. BACKGROUND

A. Graphs

A graph $G = (V, E)$ is a collection of vertices V with edges $E = \{uv\} \subseteq V \times V$. In this paper, we assume all graphs are simple (no loops or multiedges) and undirected. The complete graph on the vertex set V is the graph with all edges included, i.e., $E = \{uv \mid u \neq v \in V\}$.

We reference a few special graphs. The cycle graph on n vertices is the graph $G = (V, E)$ with $V = \{v_1, \dots, v_n\}$, and $E = \{v_i v_{i+1} \mid 1 \leqslant i < n\} \cup \{v_n v_1\}$; i.e., it forms a closed path (cycle) where no repetition occurs except for the starting and ending vertex. The complete graph on n vertices is the graph $G = (V, E)$ with $V = \{v_1, \dots, v_n\}$, and $E = \{v_i v_j \mid i \neq j\}$. That is, it is the graph with n vertices and all possible edges are included.

We also work with weighted graphs, $G = (V, E, \omega)$, where $\omega : E \to \mathbb{R}$ gives a weight for each edge in the graph. In this
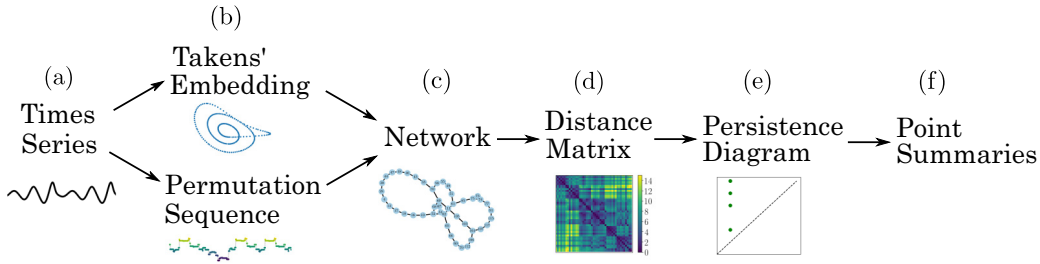
FIG. 2. Outline of method: (a) a time series (b) is embedded into either $\mathbb{R}^n$ space using Takens's embedding or segmenting into a set of permutations. From these two representations, (c) an undirected, unweighted network is formed by either applying a kth-nearest-neighbor algorithm or by setting each permutation state as a node. (d) The distance matrix is calculated using the shortest path between all nodes. (e) The persistence diagram is generated by applying persistent homology to the distance matrix. (f) Finally, one of several point summaries is used to extract information from the persistence diagram.

paper, we assume all weights are non-negative, $\omega : E \to \mathbb{R}_{\geqslant 0}$. Given an ordering of the vertices $V = \{v_1, \ldots, v_n\}$, a graph can be stored in an adjacency matrix $\mathbf{A}$ where entry $\mathbf{A}_{ij} = 1$ if there is an edge $v_i v_j \in E$ and 0 otherwise. This can be edited to store the weighting information by setting $\mathbf{A}_{ij} = \omega(v_i v_j)$ if $v_i v_j \in E$ and 0 otherwise.

A path γ in a graph is an ordered collection of nonrepeated vertices $\gamma = u_0 u_1 \cdots u_k$ where $u_i u_{i+1} \in E$ for every i. The length of the path is the number of edges used, namely, $\text{len}(\gamma) = k$ in the above notation. The distance between two vertices u and v is the minimum length of all paths from u to v and is denoted $d(u, v)$. Given an ordering of the vertices, this information can be stored in a distance matrix $\mathbf{D}$ where $\mathbf{D}_{ij} = d(v_i, v_j)$. Thus an unweighted graph $G = (V, E)$ gives rise to a weighted complete graph on the vertex set V by setting the weight $\omega(uv) = d(u, v)$.

B. k-nearest-neighbor graph

Given a collection of points in $\mathbb{R}^d$, the k-nearest-neighbor graph, or k-NN graph, is a commonly used method to build a graph. Fix $k \in \mathbb{Z}_{\geqslant 0}$. The (undirected) k-NN graph has a vertex set in one-to-one correspondence with the point cloud, so we abuse notation and write v_i for both the point $v_i \in \mathbb{R}^d$, and for the vertex $v_i \in V$. An edge $v_i v_j$ is included if v_i is among the kth nearest neighbors of v_j.

C. Embedding of time series

Takens's theorem forms one of the theoretical foundations for the analysis of time series corresponding to nonlinear, deterministic dynamical systems [3]. It basically states that in general it is possible to obtain an embedding of the attractor of a deterministic dynamical system from one-dimensional measurements of the system's evolution in time. An embedding is a smooth map $\Psi : M \to N$ between the manifolds M and N that diffeomorphically maps M to N.

Specifically, assume that the state of a system is described for any time $t \in \mathbb{R}$ by a point $\mathbf{x}$ on an m-dimensional manifold $M \subseteq \mathbb{R}^d$. The flow for this system is given by a map $\phi^t(\mathbf{x}) : M \times \mathbb{R} \to M$ which describes the evolution of the state $\mathbf{x}$ for any time t. In reality, we typically do not have access to $\mathbf{x}$, but rather have measurements of $\mathbf{x}$ via an observation function $\beta(\mathbf{x}) : M \to \mathbb{R}$. The observation function has a time evolu-

tion $\beta(\phi^t(\mathbf{x}))$, and in practice it is often a one-dimensional, discrete and equispaced time series of the form $\{\beta_n\}_{n \in \mathbb{N}}$.

Although the state $\mathbf{x}$ can lie in a higher dimension, the time series $\{\beta_n\}$ is one dimensional. Nevertheless, Takens's theorem states that by fixing an embedding dimension $d \geqslant 2m + 1$, where m is the dimension of a compact manifold M, and a time lag $\tau > 0$, then the map $\Phi_{\phi, \beta} : M \to \mathbb{R}^d$, given by

$$\Phi_{\phi, \beta} = (\beta(\mathbf{x}), \beta(\phi(\mathbf{x})), \ldots, \beta(\phi^{d-1}(\mathbf{x})))$$
$$= (\beta(\mathbf{x}_t), \beta(\mathbf{x}_{t+\tau}), \beta(\mathbf{x}_{t+2\tau}), \ldots, \beta(\mathbf{x}_{t+(d-1)\tau})),$$

is an embedding of M, where ϕ^{d-1} is the composition of ϕ $d - 1$ times and $\mathbf{x}_t$ is the value of $\mathbf{x}$ at time t.

Theoretically, any time lag τ can be used if the noise-free data are of infinite precision; however, in practice, the choice of τ is important in the delay reconstruction. In this paper we use the first minimum of the mutual information function $I_\epsilon(\tau)$ for determining a proper value for τ [6]. Figure 3 shows pictorially the quantities needed to compute the mutual information function for a fixed τ. Specifically, the range of the data values is divided into equispaced bins with resolution ϵ and a specific delay value τ is chosen and fixed. The joint probability p_{ij} of finding point $\mathbf{x}(t)$ in the ith bin and point $\mathbf{x}(t + \tau)$ in the jth bin is then computed by counting the number of points lying in the cell indexed by ij in Fig. 3 and dividing this count by the total number of transitions. In this example, we see that, for instance, $p_{2 \to 5} = 2/13 \approx 15\%$, while the marginal probability density p_i for $i = 5$ is given by $p_{i=5} = 3/13 \approx 23\%$. Using the probabilities described in Fig. 3, the mutual
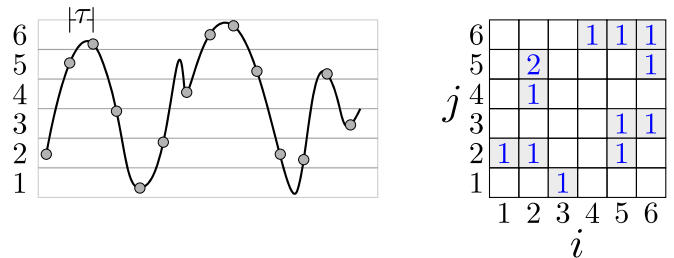

FIG. 3. The computation of the mutual information function. Each box on the right represents the count of the transitions from the ith strip on the left graph to the jth strip as the gray points on the time series are traversed from left to right.
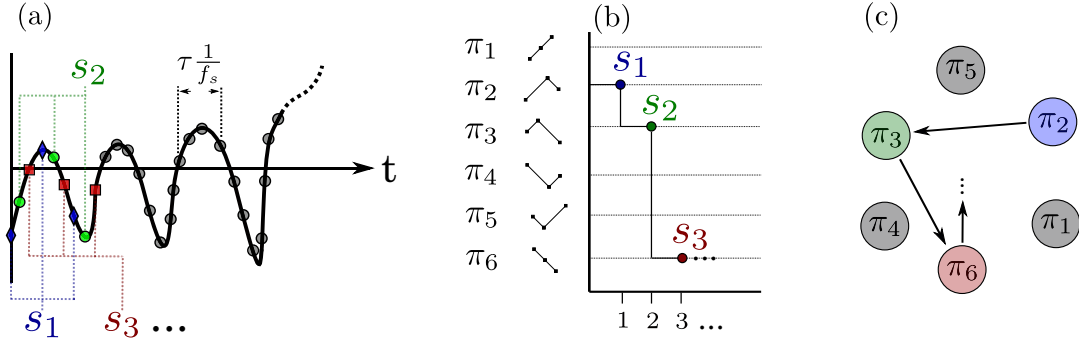
FIG. 4. Process for developing ordinal network from times series with permutation dimension $n = 3$ and delay $\tau = 3$: (a) permutations from sliding set $s_i = [x_i, x_{i+\tau}, x_{i+2\tau}, \ldots, x_{i+(n-1)\tau}]$, (b) array of ordered permutations $\mathbf{S} = [\pi_2, \pi_3, \pi_6, \ldots]$, and (c) directed path in ordinal partition network from $\mathbf{S}$.

information function can be obtained according to

$$I_\epsilon(\tau) = \sum_{i,j} p_{ij}(\tau) \ln p_{ij}(\tau) - 2 \sum_i p_i \ln p_i,$$

where $\ln(\cdot)$ is the natural logarithm function. By plotting $I_\epsilon(\tau)$ for a range of delays τ, an embedding delay can be chosen by observing the first minimum in $I_\epsilon(\tau)$. This minimum indicates the first value of τ at which minimum information is shared between $\beta(\mathbf{x}(t))$ and $\beta(\mathbf{x}(t + \tau))$. We note that the implementation that we describe here is the original implementation described in Ref. [7]; however, an adaptive implementation can be found in Ref. [83] while an entropy-based approach can be found in Ref. [84].

The other component in Takens's embedding is the embedding dimension d, which must be large enough to unfold the attractor. If this dimension in not sufficient, then some points can falsely appear to be neighbors at a smaller dimension due to the projection of the attractor onto a lower dimension. One of the classical time series analysis tools for finding a proper embedding dimension is the method of false nearest neighbors [7]. In this method, it is assumed that an appropriate embedding dimension is given by d_0. Any embedding dimension $d < d_0$ is therefore a projection from d_0 into a lower-dimensional space. Consequently, some of the coordinates are lost in this projection and points that are not neighbors in d_0 appear to be neighbors in d. The idea is therefore to embed the time series in spaces with increasingly larger dimension while keeping track of false neighbors in successive embeddings, i.e., points that appear to be neighbors due to insufficient embedding dimension. If the ratio of the false neighbors falls below a certain threshold at some embedding dimension d, then we set $d_0 \approx d$.

D. Ordinal partition graph

The ordinal partition graph [19,20] is another method for constructing a graph from a time series. Using a sufficiently sampled time series, an ordered list of permutations is collected by first finding a set of d-dimensional embedded vectors v_i using a similar algorithm to Takens's embedding. More specifically, the set of d-dimensional vectors $v_i = [x_i, x_{i+\tau}, x_{i+2\tau}, \ldots, x_{i+(d-1)\tau}]$ uses an embedding delay τ and motif dimension d. An example of this embedding is shown

in Fig. 4(a) with $\tau = 3$ and $d = 3$. These parameters were chosen as they simplify the demonstration and visualization of the method. However, to automate the method, we suggest that both τ and d are selected by using permutation entropy (PE) as defined by Bandt and Pompe [21]. The specific method for selecting τ and d is explained in Sec. III. By applying v_i over the entire length of the time series, a sequence of vectors $\mathbf{V}$ is generated.

For each vector $v_i = (x_1, \ldots, x_d)$, the associated permutation π is the permutation of the set $\{1, \ldots, d\}$ that satisfies $x_{\pi(1)} \leqslant x_{\pi(2)} \leqslant \cdots \leqslant x_{\pi(d)}$. Each vector in $\mathbf{V}$ is translated into its associated permutation symbol π_j to generate a sequence of permutations $\mathbf{P}$, where $j \in \mathbb{Z} \cap [1, n!]$. An example of this process is shown for the first three vectors in Fig. 4(b). Finally, using the array of permutations $\mathbf{P}$, a directional network is formed by transitioning from one permutation, represented by the graph in Fig. 4(c), to another in the sequential order. If we want to build an unweighted version of this graph, we include the edge $\pi\pi'$ if there is a transition from one permutation to the next. If we want this graph to be weighted, we set $\omega(\pi\pi')$ to be the number of times this transition occurs.

E. Simplicial complexes

A simplicial complex can be thought of as a generalization of the concept of a graph to higher dimensions. Given a vertex set V, a simplex $\sigma \subseteq V$ is simply a collection of vertices. The dimension of a simplex σ is $\dim(\sigma) = |\sigma| - 1$. The simplex σ is a face of τ, denoted $\sigma \preceq \tau$ if $\sigma \subseteq \tau$. A simplicial complex K is a collection of simplices $\sigma \subseteq V$ such that if $\sigma \in K$ and $\tau \preceq \sigma$, then $\tau \in K$. Equivalently stated, K is a collection of simplices which is closed under the face relation. The dimension of a simplicial complex is the largest dimension of its simplices, $\dim(K) = \max_{\sigma \in K} \dim(\sigma)$. The d-skeleton of a simplicial complex is all simplices of K with dimension at most d, $K^{(d)} = \{\sigma \in K \mid \dim(\sigma) \leqslant d\}$.

Given a graph $G = (V, E)$, we can construct the clique complex

$$K(G) = \{\sigma \subseteq V \mid uv \in E \text{ for all } u \neq v \in \sigma\}.$$

This is sometimes called the flag complex. The clique complex of the complete graph on n vertices is called the complete simplicial complex on n vertices.
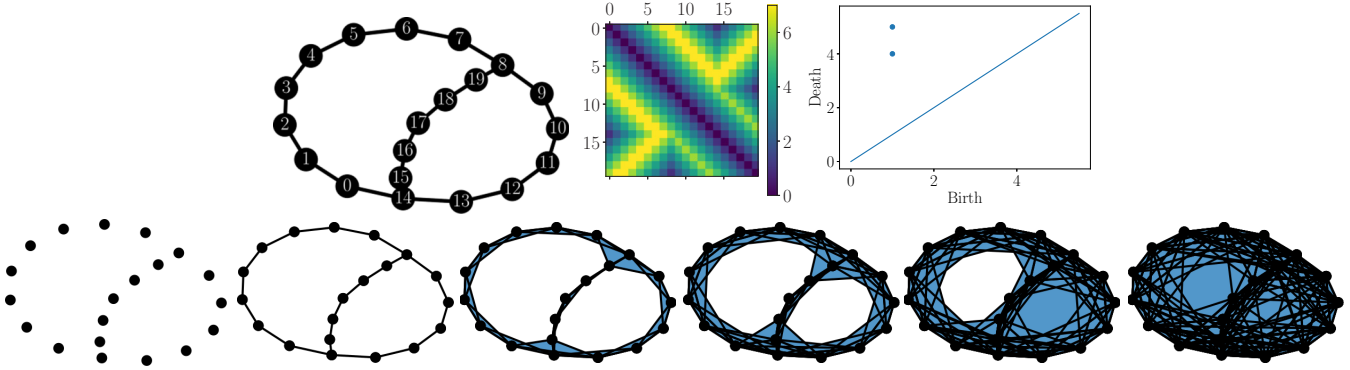
FIG. 5. An example of the method used for turning a graph (top left) with pairwise distance information (top middle) into a filtration (bottom row, shown with thresholded values 0 through 5), and then computing the resulting persistence diagram (top right).

A filtration is a collection of nested simplicial complexes

$$K_1 \subseteq K_2 \subseteq \cdots \subseteq K_N.$$

See the bottom row of Fig. 5 for an example of a filtration. A weighted graph gives rise to a filtration that we make use of extensively. Given a weighted graph $G = (V, E, \omega)$ and $a \in \mathbb{R}$, we set

$$K_a = \{\sigma \in K(G) \mid \omega(uv) \leqslant a \text{ for all } u \neq v \in \sigma\}.$$

Since $K_a \subseteq K_b$ for $a \leqslant b$, this can be viewed as a filtration

$$K_{a_1} \subseteq K_{a_2} \subseteq \cdots \subseteq K_{a_N}$$

for any collection $a_1 \leqslant a_2 \leqslant \cdots \leqslant a_N$.

In particular, for this paper, we build a filtration from an unweighted graph G by the following procedure. First, we construct the pairwise distance matrix for the vertices of G using shortest paths. This can be viewed as a weighting on the complete graph with the same vertex set as G. Thus, it induces a filtration on the complete simplicial complex K where the 1-skeleton of K_a includes edges between any pair of vertices u and v for which $d(u, v) \leqslant a$. See Fig. 5 for an example.

F. Homology

Traditional homology [30,31] counts the number of structures of a particular dimension in a given topological space, which in our context is a simplicial complex. In this context, the structures measured can be connected components (zero-dimensional structure), loops (one-dimensional structure), voids (two-dimensional structure), and higher-dimensional analogs as needed.

For the purposes of this paper, we only ever need zero- and one-dimensional persistent homology so we provide the background necessary in these contexts. Further, as a note for the expert, we always assume homology with $\mathbb{Z}_2$ coefficients which removes the need to be careful about orientation.

We start by describing homology. Assume we are given a simplicial complex K. Say the d-dimensional simplices in K are denoted $\sigma_1, \ldots, \sigma_\ell$. A d-dimensional chain is a formal sum of the d-dimensional simplices $\alpha = \sum_{i=1}^{\ell} a_i \sigma_i$. We assume the coefficients $a_i \in \mathbb{Z}_2 = \{0, 1\}$ and addition is performed mod 2. For two chains $\alpha = \sum_{i=1}^{\ell} a_i \sigma_i$ and $\beta = \sum_{i=1}^{\ell} b_i \sigma_i$, $\alpha + \beta = \sum_{i=1}^{\ell} (a_i + b_i) \sigma_i$. The collection of all d-dimensional chains forms a vector space denoted $C_d(K)$. The

boundary of a given d-simplex is

$$\partial_d(\sigma) = \sum_{\tau \prec \sigma, \dim(\tau) = d-1} \tau.$$

That is, it is the formal sum of the simplices of exactly one lower dimension. If $\dim(\sigma) = 0$, that is, if σ is a vertex, then we set $\partial_d(\sigma) = 0$. The boundary operator $\partial_d : C_d(K) \to C_{d-1}(K)$ is given by

$$\partial_d(\alpha) = \partial_d\left(\sum_{i=1}^{\ell} a_i \sigma_i\right) = \sum a_i \partial_d(\sigma_i).$$

A d-chain $\alpha \in C_d(K)$ is a cycle if $\partial_d(\alpha) = 0$; it is a boundary if there is a $(d + 1)$-chain β such that $\partial_{d+1}(\beta) = \alpha$. The group of d-dimensional cycles is denoted $Z_d(K)$; the boundaries are denoted $B_d(K)$.

In particular, any 0-chain is a 0-cycle since $\partial_0(\alpha) = 0$ for any α. A 1-chain is a 1-cycle if and only if the 1-simplices (i.e., edges) with a coefficient of 1 form a closed loop. It is a fundamental exercise in homology to see that $\partial_d \partial_{d+1} = 0$ and therefore that $B_d(K) \subseteq Z_d(K)$. The d-dimensional homology group is $H_d(K) = Z_d(K)/B_d(K)$. An element of $H_d(K)$ is called a homology class and is denoted $[\alpha]$ for $\alpha \in Z_d(K)$ where $[\alpha] = \{\alpha + \partial(\beta) \mid \beta \in C_{d+1}(K)\}$. We say that the class is represented by α, but note that any element of $[\alpha]$ can be used as a representative so this choice is by no means unique.

In the particular case of zero-dimensional homology, there is a unique class in $H_0(K)$ for each connected component of K. For one-dimensional homology, we have one homology class for each "hole" in the complex.

G. Persistent homology

Persistent homology is a tool from topological data analysis which can be used to quantify the shape of data. The main idea behind persistent homology is to watch how the homology changes over the course of a given filtration.

Fix a dimension d. First, note that if we have an inclusion of one simplicial complex to another $i : K_1 \subseteq K_2$, we have an obvious inclusion map on the d-chains $i : C_d(K_1) \to C_d(K_2)$ by simply viewing any chain in the small complex as one in the larger. Less obviously, this extends to a map on homology $i_* : H_d(K_1) \to H_d(K_2)$ by sending $[\alpha] \in H_d(K_1)$ to the class

in $H_d(K_2)$ with the same representative. That this is well defined is a nontrivial exercise in the definitions [32].

Given a filtration

$$K_1 \subseteq K_2 \subseteq \cdots \subseteq K_N$$

we have a sequence of maps on the homology

$$H_d(K_1) \to H_d(K_2) \to \cdots \to H_d(K_N).$$

A class $[\alpha] \in H_d(K_i)$ is said to be born at i if it is not in the image of the map $H_d(K_{i-1}) \to H_d(K_i)$. The same class dies at j if $[\alpha] \neq 0$ in $H_d(K_{j-1})$ but $[\alpha] = 0$ in $H_d(K_j)$.

Given all this information, we construct a persistence diagram as follows. For each class that is born at i and dies at j, we add a point in $\mathbb{R}^2$ at (i, j). For this reason, we often write a persistence diagram by its collection of off-diagonal points, $D = \{(b_1, d_1), \ldots, (b_k, d_k)\}$. See the top right of Fig. 5 for an example. Note that the farther a point is from the diagonal, the longer that class persisted in the filtration, which signifies large scale structure. The *lifetime* or *persistence* of a point $x = (b, d)$ in a persistence diagram D is given by $\mathrm{pers}(x) = |b - d|$.

Note that it is possible to have multiple points in a single diagram of the same form (b, d) if there are multiple classes that are born at b and die at d. In this case, we sometimes employ histograms or annotations to emphasize that a single point seen in a diagram is actually multiple points overlaid.

Further, any filtration has a persistence diagram for each dimension d. In this paper, when we wish to emphasize dimension, we write the diagram as D_d; here we only use the one-dimensional diagram.

H. A first example

Here, we begin to construct the pipeline we use via an example shown in Fig. 5. We start with a graph with $|V| = n = 20$ as shown in the top left. We can construct a distance between each pair of vertices using the path length. Then, a filtration on the full simplicial complex K with $n = 20$ vertices is constructed using the clique complex method described at the end of Sec. II E. Finally, the one-dimensional persistence diagram is given at the top right. The existence of two points in the persistence diagram means that two circular structures existed over the course of the filtration. The first is the large left loop that can be seen in K_1 and persists until it gets filled in at K_4. This is represented by the point $(1,4)$ in the diagram. The other smaller loop is the right loop that appears in K_1, but is filled in before K_5. This is represented by the point $(1,5)$ in the diagram.

I. Point summaries of persistence diagrams

A common issue with persistence diagrams is that they are notoriously difficult to work with as a summary of data. While they are quantitative in nature, determining differences in structure such as "has a point far from the diagonal" is often a qualitative procedure. Metrics for persistence diagrams exist, namely, the bottleneck and p-Wasserstein [85] distances; however, these objects are not particularly easy to work with in a statistical or machine learning context. Thus, we pass

n	Cycle	r_B	r_D	L_n
3		1	1	0
4		1	2	1
5		1	2	1
6		1	2	1
⋮				
10		1	4	3
⋮				

FIG. 6. Table of examples showing the lifetime L_n of the single class (r_B, r_D) in the persistence diagram for the pipeline applied to a cycle with n nodes.

to working with the simplest of featurizations, namely, point summaries of a given diagram, which we also call *scores*.

1. Maximum persistence

The first very simple but extremely useful point summary is maximum persistence. Given a persistence diagram D, the maximum persistence is simply

$$\mathrm{maxpers}(D) = \max_{x \in D} \mathrm{pers}(x).$$

In the example of Fig. 5 with $D = \{(1, 4), (1, 5)\}$, we have $\mathrm{maxpers}(D) = 4$. While this is obviously a very lossy point summary for a persistence diagram, it is quite useful in that, particularly for applications where the existence of a large circle is of interest, it often does what we need. See, e.g., Refs. [45,86].

2. Periodicity score

Next, we set out to build a point summary which we can use to measure the similarity of our weighted graph to a cycle graph which is independent of the number of nodes. If G' is an unweighted cycle graph with n vertices, then following the procedure of Sec. II H using the shortest path metric, we have that there is exactly one cycle which is born at 1, and fills in at $\lceil \frac{n}{3} \rceil$. See the examples of Fig. 6. This means the persistence diagram D' has exactly one point $(1, \lceil \frac{n}{3} \rceil)$, and so we denote the maximum persistence of this diagram as

$$L_n = \mathrm{maxpers}(D') = \left\lceil \frac{n}{3} \right\rceil - 1.$$

Then, assume we are given another unweighted graph G with $|V| = n$ and persistence diagram D. We define the network periodicity score

$$P(D) = 1 - \frac{\mathrm{maxpers}(D)}{L_n}. \qquad (1)$$

This score is an extension of the periodicity score in Ref. [51] to unweighted networks, and it has the property that $P(D) \in [0, 1]$, with $P(D) = 0$ if and only if the input graph G is a cycle graph. In the example of Fig. 5, we have $L_{20} = 6$, so $P(D) = 1 - 4/6 = 1/3$.

3. The ratio of the number of homology classes to the graph order

The next point summary we define is

$$M(D) = \frac{|D|}{|V|}, \qquad (2)$$

which is the reciprocal of the ratio between the number of vertices in the network $|V|$, i.e., the order of the graph, and the number of classes in the persistence diagram $|D|$. In the example of Fig. 5, this is $M(D) = 2/20 = 0.1$.

We can think of this number as an approximation of the reciprocal of the number of vertices in each class; however, this is only an approximation because some classes in 1D persistence diagrams may share vertices in the network. Note that for a network with n nodes, the zero-dimensional persistence diagram will always have $n-1$ points, and so this metric is not particularly useful. In this paper, we only use this summary for one-dimensional persistence diagrams.

The logic behind this heuristic is that for a periodic signal we would expect to see a small number of 1D homology classes in comparison to a chaotic time series. Therefore, for two networks of similar order but with different dynamic behavior, i.e., one is chaotic and one is periodic, the ratio $M(D)$ for the periodic time series will be smaller than its chaotic counterpart.

4. Normalized persistent entropy

Persistent entropy is a method for calculating the entropy from the lifetimes of the points in a persistence diagram, inspired by Shannon entropy. This summary function, first given by Chintakunta et al. [77], is defined as

$$E(D) = -\sum_{x \in D} \frac{\mathrm{pers}(x)}{\mathscr{L}(D)} \log_2 \left(\frac{\mathrm{pers}(x)}{\mathscr{L}(D)} \right), \qquad (3)$$

where $\mathscr{L}(D) = \sum_{x \in D} \mathrm{pers}(x)$ is the sum of lifetimes of points in the diagram. In the example of Fig. 5 and Sec. II H, $D = \{(1, 4), (1, 5)\}$, so $\mathscr{L}(D) = 3 + 4 = 7$. Thus for this example, $E(D) = 0.985$.

However, we cannot easily compare this value across different diagrams with different numbers of points. To deal with this issue, we provide the following normalization heuristic. Specifically, we normalize E as

$$E'(D) = \frac{E(D)}{\log_2(\mathscr{L}(D))}. \qquad (4)$$

This normalization allows for an accurate measurement of the entropy even when there are few significant lifetimes. Returning to the example of Fig. 5, $E'(D) = 0.351$.

III. METHOD

In this section, we discuss the specifics of the method studied for turning a time series into a persistence diagram following Fig. 2.

We have two initial choices for how to turn a time series into a network. In the case of the Takens embedding, we determine the embedding dimension using false nearest neighbors [7], and determine the lag using the mutual information function [6]. We then construct the k-NN graph for these points. Following Khor and Small [17], we use $k = 4$.

The second method for constructing a network is the ordinal partition network. As mentioned in Sec. II D, this also requires a decision of dimension and lag, which we determine following Reidl et al. [87]. Specifically, we initially fix $d = 3$, and plot the permutation entropy $H(d) = -\sum p(\pi_i) \log_2 p(\pi_i)$, where $p(\pi_i)$ is the probability of a permutation π_i, for a range of τ values. In the resulting τ versus H curve we choose the value of τ at the location of the first prominent peak as the lag parameter. The dimension $d = 3$ is used because it was shown in Ref. [87] that the first peak of $H(d)$ occurs at approximately the same value of τ independent of the dimension d for $d \in [3, 4, \ldots, 8]$ [87]. The logic behind this approach is that when the time series points are strongly correlated due to the insufficient unfolding of the trajectories, only a few regions of the state space are visited resulting in low values for H. As τ is increased, H increases and it reaches a maximum when the trajectory unfolding leads to the appearance of a large subset of the possible $d!$ motifs. We only include vertices in the graph for permutations which have been visited, which keeps us from needing to work with the full set of $d!$ which quickly becomes computationally intractable.

Using the identified delay τ at the first maximum of $H(d = 3)$, we then define the permutation entropy per symbol

$$h'(d) = \frac{1}{d-1} H(d), \qquad (5)$$

where we make d a free parameter that we are seeking to determine. The dimension for the ordinal partition network is obtained by plotting $h'(d)$ for $d \in [3, 4, \ldots, 8]$ and choosing the value of d that maximizes $h'(d)$.

Once the graph is constructed, we compute shortest paths using `all_pairs_shortest_path_length` from the PYTHON NetworkX package. Finally, we compute the persistence diagram using the PYTHON wrapper Scikit-TDA [88] for the software package RIPSER [89].

Rössler system example

We demonstrate the method on the Rössler system and the ordinal partition network representation. The Rössler system is defined as

$$\frac{dx}{dt} = -y - x, \quad \frac{dy}{dt} = x + ay, \quad \frac{dz}{dt} = b + z(x - c). \quad (6)$$

Equations (6) were solved at a rate of 20 Hz for 1000 s with parameters $a = 0.41$, $b = 2.0$, and $c = 4.0$, which results in a three-period, periodic response. Only the last 200 s [see Fig. 7(a)] are used to avoid the transients.

We form a permutation sequence from the time series using a time delay $\tau = 40$ and dimension $d = 6$, which were found using Multi-scale Permutation Entropy (MPE) as described in Sec. III. The resulting permutation sequence is shown in Fig. 7(b). Next, we form the unweighted ordinal partition network shown in Fig. 7(c). Note that this graph is drawn using the electrical-spring layout function provided by NetworkX since the permutations do not have a natural embedding into Euclidean space. Using the network, we build the distance matrix in Fig. 7(d). Finally, by applying persistent homology to the distance matrix, we obtain the persistence diagram in Fig. 7(e). However, Fig. 7(e) does not
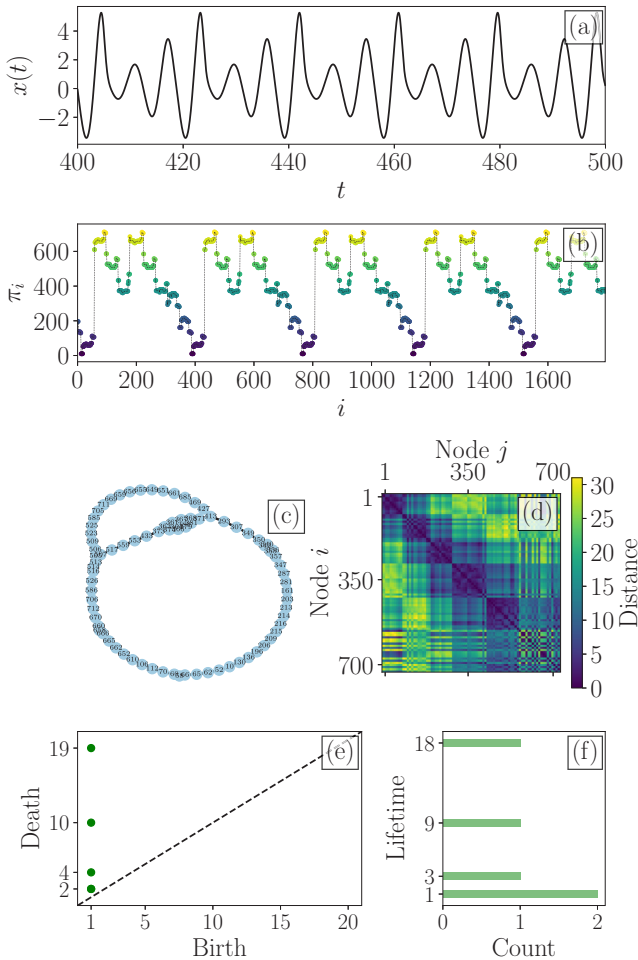
FIG. 7. Periodic Rössler system example: (a) periodic (three-period) time series, (b) resulting permutation sequence from embedded time series, (c) ordinal partition network drawn with a spring layout, (d) pairwise distance matrix using the shortest path metric, and (e) the resulting persistence diagram with (f) a histogram showing multiplicities of points in the diagram at left.

show the possibility of point multiplicity in the persistence diagram. To demonstrate this occurrence we utilize a histogram of the number of classes at each lifetime as shown in Fig. 7(f). This shows there are actually two points in the persistence diagram with lifetime 1. The point summaries described in Sec. II I are calculated as $M(D_1) = 0.06$, $P(D_1) = 0.33$, and $E'(D_1) = 0.32$.

IV. RESULTS

This section compares the persistence-based point summaries and the standard network scores, and illustrates the ability of these scores to detect dynamic state changes. Specifically, we compare the point summaries $M(D_1)$, $P(D_1)$, and $E'(D_1)$ to some commonly used network quantitative characteristics such as the mean out degree $\langle k \rangle$, the out degree variance σ^2, and the number of vertices, N. These comparisons are shown in Sec. IV A for a family of trajectories from the Rössler system, while Sec. IV B tabulates the different scores for a variety of dynamical systems. In Sec. IV C we contrast

the noise robustness of our approach to the standard network scores for ordinal partition networks.

A. Dynamic state change detection on the Rössler system

Letting the parameter a in Eqs. (6) vary in the range $0.37 < a < 0.43$ in steps of $\Delta a = 0.001$ and setting $\beta = 2$ and $\gamma = 4$, we obtain 1201 time series of length 1000 s for the state x. We only retain the last 400 s of the simulation to allow the trajectory to settle on an attractor. For the construction of the corresponding k-NN networks, we sample the time series at 2 Hz in order to capture a sufficient number of oscillations while avoiding overly large point clouds for computing persistence. For the Takens embedding we use the mutual information function approach and the nearest-neighbor method, respectively, to choose the parameters $\tau = 4$ and $d = 7$.

For constructing the ordinal partition networks we use the higher sampling frequency of 20 Hz, and we use MPE to select $\tau = 40$ and $d = 6$. We found that a higher sampling rate for ordinal partition networks and the resulting longer time series are not an issue due to the maximum number of vertices not being dependent on the length of the time series, but rather on the motif dimension d and time series complexity. Furthermore, a higher sampling rate tends to improve the detection of periodic and chaotic time series for ordinal partition networks.

The resulting point summaries were found for both ordinal partition networks (left column plots of Fig. 8) and k-NN of the Takens embedding networks (right column plots of Fig. 8). The top two graphs in Fig. 8 show the bifurcation diagram depicting the local extrema of x and the Lyapunov exponent [90], respectively. The periodic regions (shown as the regions between vertical, dashed green lines with a solid green line below) were identified by investigating the bifurcation diagram and the Lyapunov exponent plots.

For the ordinal networks, the left column plots of Fig. 8 show a significant drop in all six scores for the large periodic window corresponding to approximately $0.409 \leqslant a \leqslant 0.412$. There are also less pronounced drops in these scores for the other shorter periodic windows. These drops are especially evident for $\langle k \rangle$, $E'(D_1)$, and $P(D_1)$, where the scores significantly decrease in comparison to their surrounding values. However, some scores such as $\langle k \rangle$ are not normalized, e.g., so that $0 \leqslant \langle k \rangle \leqslant 1$. Given one time series, and not a parametrized set of series, this makes it difficult or even impossible to distinguish between chaotic and periodic regions. On the other hand, the normalized scores that we introduce in this paper, $E'(D_1)$ and $P(D_1)$, suggest periodic regions when $E'(D_1) < 0.5$ and $P(D_1) < 0.75$. It should be noted that the difference between chaotic and periodic regions, as shown in Sec. IV C, starts degrading as noise levels are increased.

For the k-NN Takens embedding networks, the right column plots of Fig. 8 show a significant drop in $P(D_1)$, $M(D_1)$, and $E'(D_1)$ during periodic windows. However, for the traditional graph scores $\langle k \rangle$ and σ^2 this drop does not clearly correspond to the beginning and end of the periodic window. Further, for the smaller periodic windows interspersed with the chaotic regions we found that $\langle k \rangle$, σ^2, and $M'(D_1)$ are too noisy to identity the dynamic state changes in these areas. In contrast, our scores $P(D_1)$ and $E'(D_1)$ retain the ability to
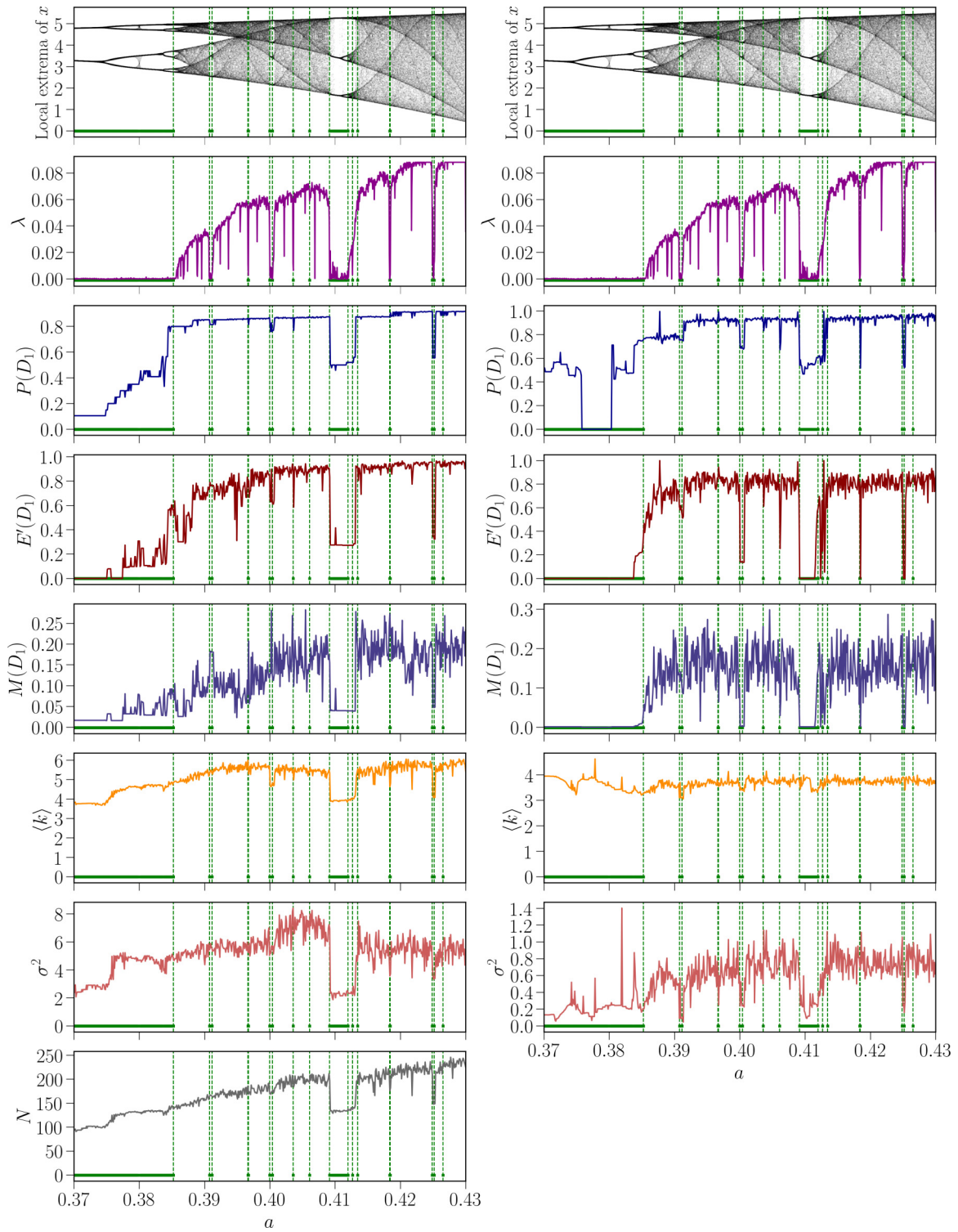
FIG. 8. Rössler system bifurcation for $0.37 < a < 0.43$ with steps of 0.001 solved using parameters provided in Eqs. (6). Left column plots include point summaries calculated from ordinal partition networks with parameters $\tau = 40$ and $d = 6$; right column plots show the same results for the k-NN networks generated from the Takens embedding with parameters $\tau = 4$ and $d = 7$. The figure compares point summaries $P(D_1)$, $M(D_1)$, and $E'(D_1)$ with the Lyapunov exponent λ [90] and some common network parameters including the number of vertices, N, mean out degree $\langle k \rangle$, and out degree variance σ^2.

TABLE I. A comparison between persistence diagram point summaries $M(D_1)$, $P(D_1)$, and $E'(D_1)$ for detecting differences in the networks generated from periodic (Per.) and chaotic (Ch.) time series using both k-NN graphs and ordinal partition graphs. NA, not available.

| | | k-NN graph from Takens's embedding | | | | | | Ordinal partition graph | | | | | |
| | | $E'(D_1)$ | | $M(D_1)$ | | $P(D_1)$ | | $E'(D_1)$ | | $M(D_1)$ | | $P(D_1)$ | |
System or Data	Ref.	Per.	Ch.	Per.	Ch.	Per.	Ch.	Per.	Ch.	Per.	Ch.	Per.	Ch.
Chua circuit	Sec. A 1	0.00	0.80	0.001	0.19	0.54	0.89	0.21	0.72	0.051	0.19	0.42	0.88
Lorenz	Sec. A 2	0.04	0.84	0.005	0.16	0.64	0.93	0.18	0.95	0.026	0.36	0.28	0.96
Rössler	Eqs. (6)	0.00	0.85	0.001	0.18	0.50	0.94	0.00	0.89	0.036	0.28	0.33	0.85
Coupled Lorenz-Rössler	Sec. A 3	0.00	0.82	0.003	0.16	0.46	0.94	0.00	0.87	0.033	0.35	0.56	0.92
Bidirectional Rössler	Sec. A 4	0.00	0.76	0.004	0.13	0.55	0.87	0.25	0.91	0.064	0.29	0.40	0.92
Mackey-Glass	Sec. A 5	0.00	0.67	0.001	0.07	0.56	0.93	0.30	0.96	0.077	0.37	0.25	0.93
Logistic map	Sec. A 8	NA	NA	NA	NA	NA	NA	0.00	0.93	0.125	0.70	0.00	0.91
Henon map	Sec. A 9	NA	NA	NA	NA	NA	NA	0.00	0.88	0.111	0.48	0.00	0.96
ECG	Sec. A 7	0.95	0.86	0.282	0.14	0.97	0.97	0.82	0.89	0.268	0.45	0.92	0.97
EEG	Sec. A 6	0.96	0.94	0.627	0.33	0.99	0.98	0.89	0.84	0.513	0.31	0.97	0.93

distinguish between dynamics regimes, and for k-NN networks of Takens embedding we suggest tagging the time series as periodic when $E'(D_1) < 0.5$ and $P(D_1) < 0.7$.

B. Tabulated results

This section uses a variety of dynamical systems to validate the observations we made for the Rössler system in Sec. IV A related to the point summaries $E'(D_1)$, $M(D_1)$, and $P(D_1)$ that we introduced in Sec. III. The results for each system when using ordinal partition networks and the k-NN network from Takens's embedding are provided side by side in Table I. The model and time series information for all of these systems are provided in the Appendix. The table can be categorized into three types of dynamical systems: (1) systems of differential equations (Chua circuit, Lorenz, Rössler, coupled Lorenz-Rössler, bidirectional Rössler, and Mackey-Glass equations), (2) discrete-time dynamical systems (logistic map and Hénon map), and (3) electrocardiogram (ECG) and electroencephalogram (EEG) signals. The paragraphs below discuss the results for each one of these systems.

1. Systems of differential equations

As shown in Table I, our point summaries from both networks yield distinguishable differences between periodic and chaotic time series. The k-NN graph results in Table I show that periodic time series have $E'(D_1) < 0.5$, $M(D_1) < 0.15$, and $P(D_1) < 0.7$. Similarly, the ordinal partition graph scores in Table I show that periodic time series have $E'(D_1) < 0.5$, $M(D_1) < 0.07$, and $P(D_1) < 0.75$.

2. Discrete dynamical systems

The results for the discrete dynamical equations in Table I show distinguishable differences between periodic maps in comparison to chaotic maps when using ordinal partition networks. Takens's embedding was not applied to the discrete dynamical systems, and only the ordinal partition network results are reported here because working with these networks is more natural for maps.

3. EEG and ECG results

The point summary results from real world data sets (ECG and EEG) shown in Table I have inherent noise, which causes the differences between the compared states to be less significant as shown in Fig. 9. The k-NN graph results in Table I do not show a significant difference between the two groups for either ECG or EEG data. This is most likely due to the sensitivity of Takens's embedding to noise and perturbations. However, we did find a difference between epileptic and healthy patients through the networks formed by ordinal partitions for ECG [91] and EEG [92] data. IV C discusses the effect of additive noise on the point summaries in more detail. As a note, there have been other methods for characterizing EEG data using TDA and persistent entropy [82], but our method is different from prior works because we apply persistent homology to the generated networks.

C. Effects of additive noise

In this section we investigate the noise robustness of the point summaries in comparison to some common network parameters—mean out degree $\langle k \rangle$, out degree variance σ^2, and the number of vertices, N. The ordinal partition networks are based on time series from the Rössler system defined in Eqs. (6) with $b = 2.0$, $c = 4.0$, and either $a = 0.41$ or $a = 0.43$ for a periodic or chaotic response, respectively.

To make comparisons on the noise robustness we add Gaussian noise to the signal and calculate the point summaries and network parameters at various signal-to-noise ratios (SNR) for both periodic and chaotic Rössler systems. The chosen SNR values were all the integers from 1 to 50, and at each SNR value we obtain 25 realizations of noisy signals.

To determine the 68% confidence interval at each SNR, we repeat the calculation of the point summaries and network parameters for all noise realizations at each SNR level, and we set our confidence interval to $\bar{x}(\mathrm{SNR}) \pm s(\mathrm{SNR})$ where $\bar{x}(\mathrm{SNR})$ and $s(\mathrm{SNR})$ are the sample average and sample standard deviation, respectively, at a specific SNR value. Figure 9 shows the mean values and confidence intervals for each SNR. To assess the ability of point summaries to assign a distinguishing score to a periodic versus a chaotic system in the
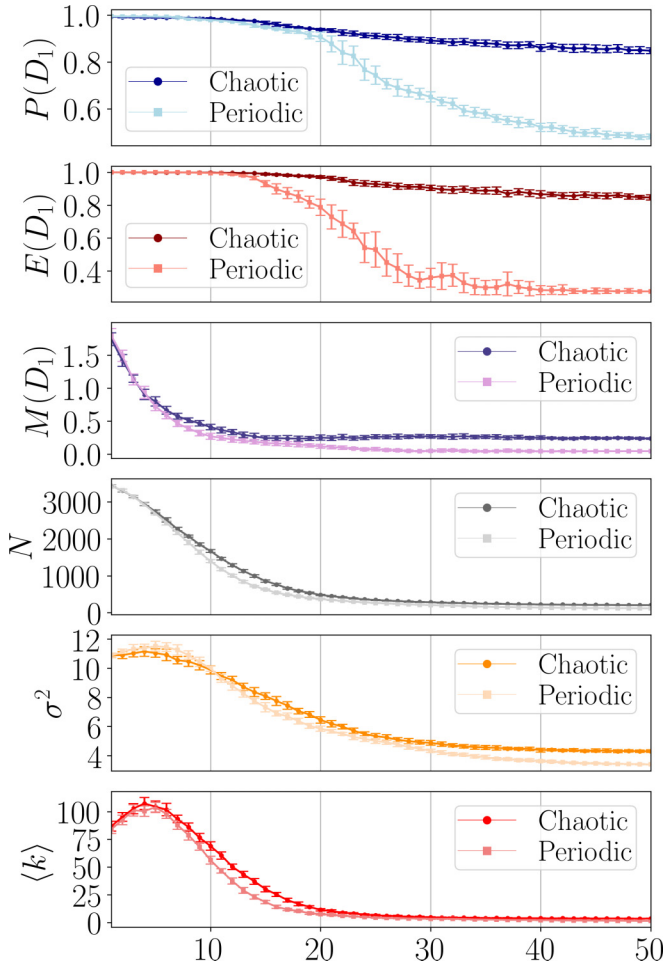
FIG. 9. Average point summaries and network parameters for varying signal-to-noise ratios (SNRs) from Gaussian noise added to time series generated from periodic and chaotic Rössler systems. For each SNR, 25 separate samples are taken to provide mean values and standard deviations, which are shown as the error bars.

presence of noise, we check for an overlap in the confidence intervals for the periodic and chaotic results at each SNR. If for a particular point summary there is an overlap between the scores for periodic and chaotic time series, then that point summary is not effective in distinguishing the dynamics at that specific SNR. Table II summarizes the noise robustness by providing the lowest SNR at which each point summary

TABLE II. Noise robustness comparison for persistence diagram point summaries and network parameters using ordinal partition network.

Point summary or network parameter	Lowest distinguishing SNR
$E'(D_1)$	14
$M(D_1)$	19
$P(D_1)$	20
$\langle k \rangle$	29
σ^2	29
N	8

and network parameter no longer has an overlap between the periodic and chaotic confidence intervals. This result shows a lower capable SNR for the persistence-based point summaries than the mean out degree $\langle k \rangle$ and variance σ^2. Another trend that should be noted is the reduction in difference between periodic and chaotic time series for high levels of noise. This should be taken into account when applying the point summaries to real world data with intrinsic noise.

V. CONCLUSIONS

In this paper we develop an alternative framework for time series analysis using TDA. We investigate two methods for embedding a time series into an unweighted graph: (1) utilizing standard Takens theorem techniques, then building a k-NN graph; and (2) using ordinal partition networks to turn (visited) parts of the state space into symbols, and obtaining a graph by tracking sequential transitions between these subspaces. We then describe how to obtain the 1D persistence diagram corresponding to the graph by defining a filtration on the full simplicial complex using the pairwise distances between the graph vertices. The obtained persistence diagram then allows the application of tools from TDA to gain insights into the system's underlying dynamics. Specifically, a graph embedding of a periodic time series has long connected network loops, while a chaotic time series has many short loops. These characteristics allow persistent homology to accurately distinguish periodic and chaotic time series by measuring the shape of the networks.

In addition to describing this approach for time series analysis, another contribution of this work is the introduction of new point summaries for extracting information about the dynamic state (periodic or chaotic) from time series measurements. Specifically, we extend the periodicity score $P(D_1)$, which was defined on $\mathbb{R}^n$ in Ref. [51], to abstract graph spaces. We also define a heuristic $M(D_1)$ which represents an approximation of the ratio of the number of homology classes to the graph order. The last point summary we define is a normalized version of the persistence entropy $E'(D_1)$ [77].

We found that these point summaries outperform standard graph scores (see Fig. 8). Specifically, our point summaries are more capable of distinguishing shifts in the dynamic behavior than their traditional graph scores counterpart. Further, our point summaries, especially the two normalized scores $P(D_1)$ and $E'(D_1)$, enable making inferences about the dynamic behavior from isolated time series, as opposed to tracking changes in the scores of parametrized time series some of which belong to a known dynamic regime. For example, applying our point summaries to ordinal partition networks from a variety of dynamical systems in Table I, we observed that that periodic time series typically have $E'(D_1) < 0.5$, $M(D_1) < 0.15$, and $P(D_1) < 0.7$. Similarly, using the networks obtained from k-NN embedding shows that periodic time series have $E'(D_1) < 0.5$, $M(D_1) < 0.07$, and $P(D_1) < 0.75$ (see Table I). However, for both discrete dynamical systems as well as ECG and EEG data, only the persistent homology of the ordinal partition network was able to distinguish between the two data sets. Additionally, we showed in Fig. 9 that the point summaries of the ordinal partition networks are noise robust down to an SNR of approximately 15 with additive Gaussian noise. In future work,

to develop more precise ranges for periodic point summary scores and to improve the dynamic state detection, it would be beneficial to investigate the statistical significance of these point summaries as well as the correlation between the point summaries for the k-NN and ordinal partition networks.

APPENDIX: DYNAMICAL SYSTEM EXAMPLES

1. Chua circuit

The Chua circuit used is defined as

$$\frac{dx}{dt} = \alpha(y - x - f(x)),$$
$$\frac{dy}{dt} = \gamma(x - y + z), \qquad (A1)$$
$$\frac{dz}{dt} = -\beta y,$$

where $f(x)$ is defined as $f(x) = m_1 x + \frac{1}{2}(m_0 + m_1)(|x+1| - |x-1|)$, with $m_0 = -8/7$ and $m_1 = -5/7$. Additionally, the Chua circuit had a sampling rate of 50 Hz with parameters $\alpha = 15.6$, $\gamma = 1.0$, and $\beta = 33.80$ for a periodic response or $\beta = 33.55$ for a chaotic response. This system was solved for 500 s and the last 100 s were used. The generated time series were down-sampled to 7 Hz for k-NN networks.

2. Lorenz system

The Lorenz system used is defined as

$$\frac{dx}{dt} = \sigma(y - x), \quad \frac{dy}{dt} = x(\rho - z) - y, \quad \frac{dz}{dt} = xy - \beta z. \quad (A2)$$

The Lorenz system had a sampling rate of 100 Hz with parameters $\sigma = 10.0$, $\beta = 8.0/3.0$, and $\rho = 180.1$ for a periodic response or $\rho = 181.0$ for a chaotic response. This system was solved for 100 s and the last 24 s were used. The generated time series were down-sampled to 35 Hz for k-NN networks.

3. Coupled Lorenz-Rössler system

The coupled Lorenz-Rössler system used is defined as

$$\frac{dx_1}{dt} = -y_1 - z_1 + k_1(x_2 - x_1),$$
$$\frac{dy_1}{dt} = x_1 + a_2 y_1 + k_2(y_2 - y_1),$$
$$\frac{dz_1}{dt} = b_2 + z_1(x_1 - c_2) + k_3(z_2 - z_1),$$
$$\frac{dx_2}{dt} = \sigma(y_2 - x_2),$$
$$\frac{dy_2}{dt} = \lambda x_2 - y_2 - x_2 z_2,$$
$$\frac{dz_2}{dt} = x_2 y_2 - b_1 z_2, \qquad (A3)$$

with $\sigma = 10$, $b_1 = 8/3$, $b_2 = 0.2$, $c_2 = 10$, $k_1 = 1$, $k_2 = 10$, $k_3 = 0$, $\lambda = 28$, and $a_2 = 0.25$ for a periodic response or $a_2 = 0.51$ for a chaotic response. This was solved for 400 s with a sampling rate of 50 Hz. Only the last 200 s of the x solution were used in the analysis. The generated time series were down-sampled to 2 Hz for k-NN networks.

4. Bidirectional coupled Rössler system

The bidirectional Rössler system is defined as

$$\frac{dx_1}{dt} = -w_1 y_1 - z_1 + k(x_2 - x_1),$$
$$\frac{dy_1}{dt} = w_1 x_1 + 0.165 y_1,$$
$$\frac{dz_1}{dt} = 0.2 + z_1(x_1 - 10),$$
$$\frac{dx_2}{dt} = -w_2 y_2 - z2 + k(x_1 - x_2),$$
$$\frac{dy_2}{dt} = w_2 x_2 + 0.165 y_2,$$
$$\frac{dz_2}{dt} = 0.2 + z_2(x_2 - 10), \qquad (A4)$$

with $w_1 = 0.99$, $w_2 = 0.95$, and $k = 0.0544$ for a periodic response or $k = 0.0558$ for a chaotic response. This was solved over 4000 s with a sampling rate of 10 Hz. Only the last 400 s of the x solution were used in the analysis. The generated time series were down-sampled to 1 Hz for k-NN networks. The generated time series were down-sampled to 2 Hz for k-NN networks.

5. Mackey-Glass delayed differential equation

The Mackey-Glass delayed differential equation is defined as

$$x(t) = -\gamma x(t) + \beta \frac{x(t - \tau)}{1 + x(t - \tau)^n} \qquad (A5)$$

with $\tau = 2$, $\beta = 2$, $\gamma = 1$, and $n = 7.00$ for a periodic response or $n = 9.65$ for a chaotic response. This was solved for 400 s with a sampling rate of 100 Hz. Only the last 300 s of the x solution were used in the analysis. The generated time series were down-sampled to 2.5 Hz for k-NN networks and 25 Hz for ordinal partition networks.

6. EEG data

The EEG signal was taken from Andrzejak *et al.* [92]. More specifically, the first 2000 data points from the EEG data of a healthy patient from set A, file Z-093, was used as the periodic series, and the first 2000 data points from the EEG data of a patient during active seizures from set E, file S-056, was used as the chaotic series. The generated time series were down-sampled to 80 Hz for k-NN networks.

7. ECG data

The electrocardiogram (ECG) data were taken from SCIPY's misc.electrocardiogram data set. These ECG data were originally provided by the MIT-BIH Arrhythmia Database [91]. We used data points 3000 to 4500 during normal sinus rhythm as the periodic time series and data

points 8500 to 10 000 during ventricular contractions as the chaotic time series.

8. Logistic map

The logistic map was generated as

$$x_{n+1} = rx_n(1 - x_n), \tag{A6}$$

with $x_0 = 0.5$ and $r = 3.50$ for periodic results or $r = 3.95$ for chaotic results. Equation (A6) was solved for the first 500 data points.

9. Hénon map

The Hénon map was solved as

$$x_{n+1} = 1 - ax_n^2 + y_n,$$
$$y_{n+1} = bx_n, \tag{A7}$$

where $b = 0.3$, $x_0 = 0.1$, $y_0 = 0.3$, and $a = 1.05$ for a periodic response and $a = 1.4$ for a chaotic response. This system was solved for the first 500 data points of the x solution.

[1] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis* (Cambridge University Press, Cambridge, UK, 2004).

[2] E. Bradley and H. Kantz, Chaos **25**, 097610 (2015).

[3] F. Takens, in *Dynamical Systems and Turbulence, Warwick 1980*, edited by D. Rand and L.-S. Young, Lecture Notes in Mathematics Vol. 898 (Springer, Berlin, 1981), pp. 366–381.

[4] T. Sauer, J. Yorke, and M. Casdagli, J. Stat. Phys. **65**, 579 (1991).

[5] M. Casdagli, S. Eubank, J. Farmer, and J. Gibson, Physica D **51**, 52 (1991).

[6] A. M. Fraser and H. L. Swinney, Phys. Rev. A **33**, 1134 (1986).

[7] M. B. Kennel, R. Brown, and H. D. I. Abarbanel, Phys. Rev. A **45**, 3403 (1992).

[8] J.-P. Eckmann, S. O. Kamphorst, and D. Ruelle, Europhys. Lett. **4**, 973 (1987).

[9] N. Marwan, M. Carmenromano, M. Thiel, and J. Kurths, Phys. Rep. **438**, 237 (2007).

[10] In this paper, we use the words network and graph interchangeably.

[11] Z. Gao and N. Jin, Phys. Rev. E **79**, 066303 (2009).

[12] Z. Gao and N. Jin, Chaos **19**, 033137 (2009).

[13] N. Marwan, J. F. Donges, Y. Zou, R. V. Donner, and J. Kurths, Phys. Lett. A **373**, 4246 (2009).

[14] R. V. Donner, Y. Zou, J. F. Donges, N. Marwan, and J. Kurths, New J. Phys. **12**, 033025 (2010).

[15] R. V. Donner, M. Small, J. F. Donges, N. Marwan, Y. Zou, R. Xiang, and J. Kurths, Int. J. Bifurcation Chaos **21**, 1019 (2011).

[16] Y. Shimada, T. Kimura, and T. Ikeguchi, in *Artificial Neural Networks: ICANN 2008* (Springer, Berlin, 2008), pp. 61–70.

[17] A. Khor and M. Small, Chaos **26**, 043101 (2016).

[18] X. Xu, J. Zhang, and M. Small, Proc. Natl. Acad. Sci. USA **105**, 19601 (2008).

[19] M. Small, in *Proceedings of the 2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)* (IEEE, New York, 2013).

[20] M. McCullough, M. Small, T. Stemler, and H. H.-C. Iu, Chaos **25**, 053101 (2015).

[21] C. Bandt and B. Pompe, Phys. Rev. Lett. **88**, 174102 (2002).

[22] While the ordinal partition network can be defined as a directed, weighted graph, we work with the undirected, unweighted analog.

[23] J. M. Amigó, S. Zambrano, and M. A. F. Sanjuán, Europhys. Lett. **79**, 50001 (2007).

[24] J. Amigó, *Permutation Complexity in Dynamical Systems* (Springer, Berlin, 2010).

[25] G. Carlsson, Bull. Am. Math. Soc. **46**, 255 (2009).

[26] R. Ghrist, *Elementary Applied Topology* (CreateSpace, Scotts Valley, CA, 2014).

[27] R. Ghrist, Bull. Am. Math. Soc. **45**, 61 (2008).

[28] E. Munch, J. Learn. Anal. **4**, 47 (2017).

[29] J. A. Perea, arXiv:1809.03624.

[30] A. Hatcher, *Algebraic Topology* (Cambridge University Press, Cambridge, UK, 2002).

[31] J. R. Munkres, *Elements of Algebraic Topology* (Addison-Wesley, Reading, MA, 1993).

[32] H. Edelsbrunner, D. Letscher, and A. Zomorodian, Discrete Comput. Geom. **28**, 511 (2002).

[33] A. Zomorodian and G. Carlsson, Discrete Comput. Geom. **33**, 249 (2005).

[34] M. Robinson, *Topological Signal Processing* (Springer, Berlin, 2014).

[35] J. A. Perea and J. Harer, Found. Comput. Math. **15**, 799 (2015).

[36] J. J. Berwald, M. Gidea, and M. Vejdemo-Johansson, Discontin. Nonlin. Complex. **3**, 413 (2014).

[37] J. A. Perea, in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, New York, 2016).

[38] B. Xu, C. J. Tralie, A. Antia, M. Lin, and J. A. Perea, arXiv:1809.07131.

[39] N. Sanderson, E. Shugerman, S. Molnar, J. D. Meiss, and E. Bradley, in *Computational Topology Techniques for Characterizing Time-Series Data*, edited by N. Adams, A. Tucker, and D. Weston, Advances in Intelligent Data Analysis XVI, IDA 2017, Lecture Notes in Computer Science, Vol. 10584 (Springer, Cham, 2017), pp. 284–296.

[40] M. Robinson, in *Proceedings of the 2015 International Conference on Sampling Theory and Applications (SampTA)* (IEEE, Washington, DC, 2015), pp. 588–592.

[41] J. R. Tempelman and F. A. Khasawneh, arXiv:1902.05918.

[42] J. Garland, E. Bradley, and J. D. Meiss, Physica D **334**, 49 (2016).

[43] S. Maletić, Y. Zhao, and M. Rajković, Chaos **26**, 053105 (2016).

[44] C. M. Pereira and R. F. de Mello, Expert Syst. Appl. **42**, 6026 (2015).

[45] F. A. Khasawneh and E. Munch, Mech. Syst. Signal Process. **70-71**, 527 (2016).

[46] F. A. Khasawneh and E. Munch, in *Proceedings of the ASME 2014 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Volume 8: 26th Conference on Mechanical Vibration and Noise,*

Buffalo, NY, USA, 2014 (ASME, New York, 2014), Paper no. DETC2014/VIB-35655, p. V008T11A034.

[47] F. A. Khasawneh and E. Munch, in *Proceedings of the ASME 2014 International Mechanical Engineering Congress & Exposition, IMECE2014, Nov. 14–20, 2014, Montreal, Quebec, Canada, Volume 4B: Dynamics, Vibration, and Control* (ASME, New York, 2014), Paper no. IMECE2014-40221, p. V04BT04A038.

[48] F. A. Khasawneh and E. Munch, in *Time Delay Systems: Theory, Numerics, Applications, and Experiments*, edited by T. Insperger, T. Ersal, and G. Orosz (Springer International, Cham, 2017), pp. 93–106.

[49] F. A. Khasawneh and E. Munch, Proc. R. Soc. A **474**, 20180027 (2018).

[50] J. Berwald and M. Gidea, Math. Biosci. Eng. **11**, 723 (2014).

[51] J. A. Perea, A. Deckard, S. B. Haase, and J. Harer, BMC Bioinform. **16**, 257 (2015).

[52] M. Gidea and Y. Katz, Physica A **491**, 820 (2018).

[53] M. Gidea, in *Proceedings of the 3rd International Winter School and Conference on Network Science: NetSci-X 2017*, edited by E. Shmueli, B. Barzel, and R. Puzis (Springer International, Cham, 2017), pp. 47–59.

[54] M. Gidea, D. Goldsmith, Y. A. Katz, P. Roldan, and Y. Shmalo, *Topological Recognition of Critical Transitions in Time Series of Cryptocurrencies* (2018), https://ssrn.com/abstract=3202721.

[55] S. Emrani, T. Gentimis, and H. Krim, IEEE Signal Process. Lett. **21**, 459 (2014).

[56] M. Robinson, S. Fennell, B. DiZio, and J. Dumiak, J. Acoust. Soc. Am. **143**, 1630 (2018).

[57] C. Tralie, in *Proceedings of the 32nd International Symposium on Computational Geometry (SoCG 2016)*, edited by S. Fekete and A. Lubiw, Leibniz International Proceedings in Informatics (Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2016), Vol. 51, pp. 71:1–71:5.

[58] C. J. Tralie and J. A. Perea, SIAM J. Imaging Sci. **11**, 1049 (2018).

[59] C. J. Tralie and M. Berger, in *Proceedings of the 2018 25th IEEE International Conference on Image Processing (ICIP)* (IEEE, New York, 2018).

[60] P. Bendich, E. Gasparovic, J. Harer, and C. J. Tralie, in *Association for Women in Mathematics Series* (Springer International, Berlin, 2018), pp. 93–114.

[61] C. J. Tralie and B. McFee, arXiv:1902.01023.

[62] Y. Mileyko, S. Mukherjee, and J. Harer, Inverse Probl. **27**, 124007 (2011).

[63] K. Turner, Y. Mileyko, S. Mukherjee, and J. Harer, Discrete Comput. Geom. **52**, 44 (2014).

[64] E. Munch, K. Turner, P. Bendich, S. Mukherjee, J. Mattingly, and J. Harer, Electron. J. Statist. **9**, 1173 (2015).

[65] P. Bubenik, J. Mach. Learn. Res. **16**, 77 (2015).

[66] H. Adams, T. Emerson, M. Kirby, R. Neville, C. Peterson, P. Shipman, S. Chepushtanova, E. Hanson, F. Motta, and L. Ziegelmeier, J. Mach. Learn. Res. **18**, 218 (2017).

[67] A. Adcock, E. Carlsson, and G. Carlsson, Homol. Homotopy Appl. **18**, 381 (2016).

[68] G. Carlsson and S. K. Verovšek, J. Pure Appl. Algebra **220**, 3610 (2016).

[69] S. Kališnik, Found. Comput. Math. **19**, 101 (2019).

[70] B. Di Fabio and M. Ferri, in *Comparing Persistence Diagrams Through Complex Vectors*, edited by V. Murino and E. Puppo, Image Analysis and Processing — ICIAP 2015, ICIAP 2015, Lecture Notes in Computer Science, Vol. 9279 (Springer, Cham, 2015), pp. 294–305.

[71] E. Berry, Y.-C. Chen, J. Cisewski-Kehe, and B. T. Fasy, arXiv:1804.01618.

[72] I. Chevyrev, V. Nanda, and H. Oberhauser, IEEE Trans. Pattern Anal. Mach. Intell. 1 (2018).

[73] Y.-C. Chen, D. Wang, A. Rinaldo, and L. Wasserman, arXiv:1510.02502.

[74] F. Chazal, B. T. Fasy, F. Lecci, A. Rinaldo, and L. Wasserman, in *Proceedings of the Thirtieth Annual Symposium on Computational Geometry, SOCG'14* (ACM Press, New York, 2014), pp. 474–483.

[75] T. Padellini and P. Brutti, arXiv:1709.07097.

[76] J. A. Perea, E. Munch, and F. A. Khasawneh, arXiv:1902.07190.

[77] H. Chintakunta, T. Gentimis, R. Gonzalez-Diaz, M.-J. Jimenez, and H. Krim, Pattern Recognit. **48**, 391 (2015).

[78] M. Rucco, R. Gonzalez-Diaz, M.-J. Jimenez, N. Atienza, C. Cristalli, E. Concettoni, A. Ferrante, and E. Merelli, Signal Process. **134**, 130 (2017).

[79] N. Atienza, R. Gonzalez-Diaz, and M. Soriano-Trigueros, arXiv:1803.08304.

[80] E. Merelli, M. Rucco, P. Sloot, and L. Tesei, Entropy **17**, 6872 (2015).

[81] M. Rucco, F. Castiglione, E. Merelli, and M. Pettini, in *Proceedings of ECCS 2014* (Springer International, Berlin, 2016), pp. 117–128.

[82] M. Piangerelli, M. Rucco, L. Tesei, and E. Merelli, BMC Research Notes **11**, 392 (2018).

[83] G. Darbellay and I. Vajda, IEEE Trans. Inf. Theory **45**, 1315 (1999).

[84] A. Kraskov, H. Stögbauer, and P. Grassberger, Phys. Rev. E **69**, 066138 (2004).

[85] This metric is closely related to but not the same as the eponymous metric from probability theory.

[86] S. Tymochko, E. Munch, J. Dunion, K. Corbosiero, and R. Torn, arXiv:1902.06202.

[87] M. Riedl, A. Müller, and N. Wessel, Eur. Phys. J.: Spec. Top. **222**, 249 (2013).

[88] N. Saul and C. Tralie, Scikit-TDA: Topological data analysis for PYTHON (unpublished).

[89] U. Bauer, RIPSER, https://github.com/Ripser/ripser.

[90] G. Benettin, L. Galgani, A. Giorgilli, and J.-M. Strelcyn, Meccanica **15**, 21 (1980).

[91] G. B. Moody and R. G. Mark, MIT-BIH arrhythmia database (unpublished).

[92] R. G. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David, and C. E. Elger, Phys. Rev. E **64**, 061907 (2001).