University of Oxford

UNIVERSITY OF OXFORD DEPARTMENT OF STATISTICS

Topological Data Analysis of Temporal Networks

by

Dimitri Lozeve

St Catherine's College

A dissertation submitted in partial fulfilment of the degree of Master of Science in Applied Statistics

Department of Statistics, 24–29 St Giles,
Oxford, OX1 3LB

September 2018

Declaration of authorship

This my own work (except where otherwise indicated).

Date	Signature

Abstract

Abstract here

Acknowledgements

Thank you!

Contents

List of Figures

List of Tables

1 Introduction

2 *Topological Data Analysis and Persistent Homology*

2.1 Homology

Our goal is to understand the topological structure of a metric space. For this, we can use *homology*, which consists in associating for a metric space X and a dimension i a vector space $H_i(X)$. The dimension of $H_i(X)$ will give us the number of i-dimensional components in X: the dimension of $H_0(X)$ is the number of path-connected components in X, the dimension of $H_1(X)$ is the number of holes in X, and the dimension of $H_2(X)$ is the number of voids.

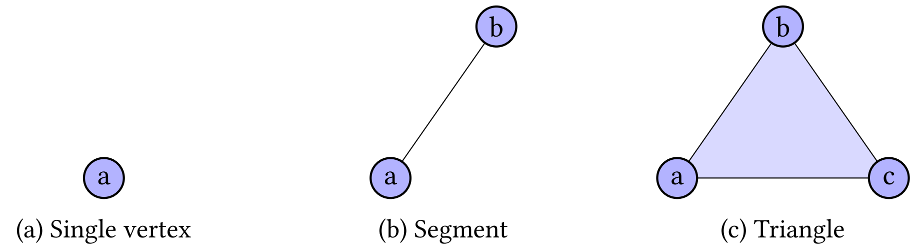
Crucially, these vector spaces are robust to continuous deformation of the underlying metric space (they are *homotopy invariant*). However, computing the homology of an arbitrary metric space can be extremely difficult. It is necessary to approximate it in a structure that would be both combinatorial and topological in nature.

2.2 Simplicial Complexes

In order to understand the topological structure of a metric space, we need a way to decompose it in smaller pieces which, when assembled, conserve the overall organisation of the space. For this, we use a structure called a *simplicial complex*, which is a kind of higher-dimensional generalization of graphs.

The building blocks of this representation will be *simplices*, which are simply the convex hull of an arbitrary set of points. Examples of simplices include single points, segments, triangles, and tetrahedrons (in dimensions 0, 1,, 2, and 3 respectively).

Definition 2.1 (Simplex). The *k-dimensional simplex* $\sigma = [x_0, \ldots, x_k]$ is the convex hull of the set $\{x_0, \ldots, x_k\} \in \mathbb{R}^d$, where $x_0, \ldots, x_k$ are affinely independent. $x_0, \ldots, x_k$ are called the *vertices* of σ, and the simplices defined by the subsets of $\{x_0, \ldots, x_k\}$ are called the *faces* of σ.



(a) Single vertex (b) Segment (c) Triangle

Figure 21: Examples of simplices

We then need a way to combine these basic building blocks meaningfully so that the resulting object can adequately reflect the topological structure of the metric space.

Definition 2.2 (Simplicial complex)**.** A *simplicial complex* is a collection K of simplices such that:

- any face of a simplex of K is a simplex of K
- the intersection of two simplices of K is either the empty set or a common face or both.

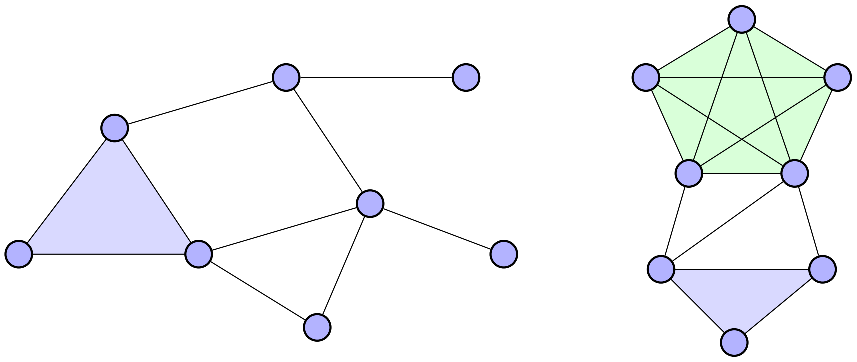


Figure 22: Example of a simplicial complex, with two connected components, two 3-simplices, and one 5-simplex.

The notion of simplicial complex is closely related to that of a hypergraph. The important distinction lies in the fact that a subset of a hyperedge is not necessarily a hyperedge itself.

Using these definitions, we can define homology on simplicial complexes.

2.3 Filtrations

If we consider that a simplicial complex is a kind of "discretization" of a metric space, we realise that there must be an issue of *scale*. For our analysis to be invariant under small perturbations in the data, we need a way to find the optimal scale parameter to capture the adequate topological structure, without taking into account some small perturbations, nor ignoring some important smaller features.

The ideal solution to these problems is to consider all scales at once: this is the objective of *filtered simplical complexes*.

Definition 2.3 (Filtration)**.** A *filtered simplicial complex*, or simply a *filtration*, K is a sequence $(K_i)_{i\in I}$ of simplicial complexes such that:

- for any $i, j \in I$, if $i < j$ then $K_i \subseteq K_j$,
- $\bigcup_{i\in I} K_i = K$.

2.4 Persistent Homology

We can now compute the homology for each step in a filtration. This leads to the notion of *persistent homology* [1, 7], which gives us all the information necessary to establish the topological structure of the metric space at multiple scales.

Definition 2.4 (Persistent homology)**.** The *p-th persistent homology* of a simplicial complex $K = (K_i)_{i\in I}$ is the pair $(\{H_p(K_i)\}_{i\in I}, \{f_{i,j}\}_{i,j\in I, i\leq j})$, where for all $i \leq j$, $f_{i,j} : H_p(K_i) \mapsto H_p(K_j)$ is induced by the inclusion map $K_i \mapsto K_j$.

The functions $f_{i,j}$ allow us to link generators in each successive homology space in the filtration. Since each generator correspond to a topological feature (connected component, hole, void, etc, depending on the dimension p), we can determine whether it survives in the next step of the filtration. We can now determine when each feature is born and when it dies (if it dies at all). This representation will be dependent on the choice of basis for each homology space $H_p(K_i)$. However, by the Fundamental Theorem of Persistent Homology, we can choose base vectors in each homology space such that the collection of half-open intervals is well-defined and unique. This construction is called a *barcode*.

2.5 Topological summaries: barcodes and persistence diagrams

In order to interpret the results of the persistent homology computation, we need to compare the output for a particular data set to a suitable null model. For this, we need some kind of a similarity measure between barcodes and a way to evaluate the statistical significance of the results.

One possible approach for this is to define a space in which we can project barcodes and study their geometric properties. *Persistence diagrams* are an example of such a space.

Definition 2.5 (Persistence diagrams)**.** A *persistence diagram* is the union of a finite multiset of points in $\bar{\mathbb{R}}^2$ zith the diagonal $\Delta = \{(x,x) \mid x \in \mathbb{R}^2\}$, where every point of Δ has infinite multiplicity.

The diagonal Δ is added to facilitate comparisons between diagrams, as points near the diagonal correspond to short-lived topological feature, thus likely to be caused by small perturbations in the data.

We can now define several distances on the space of persistence diagrams.

Definition 2.6 (Wasserstein distance)**.** The *p-th Wasserstein distance* between two diagrams X and Y is

$$W_p[d](X,Y) = \inf_{\phi : X \mapsto Y} \left[\sum_{x \in X} d\left(x, \phi(x)\right)^p \right]$$

for $p \in [1, \infty)$, and

$$W_\infty[d](X,Y) = \inf_{\phi : X \mapsto Y} \sup_{x \in X} d\left(x, \phi(x)\right)$$

for $p = \infty$, where d is a distance on $\mathbb{R}^2$ and ϕ ranges over all bijections from X to Y.

Definition 2.7 (Bottleneck distance)**.** The *bottleneck distance* is defined as the infinite Wasserstein distance with d the uniform norm: $d_B = W_\infty[L_\infty]$.

Since the bottleneck distance is by far the most commonly used, we will focus on it in the following. It is symmetric, non-negative, and satisfies the triangle inequality. However, it is not a true distance, as it is fairly straightforward to come up with two distinct diagrams at bottleneck distance zero, even on multisets not touching the diagonal Δ.

2.6 Stability

2.7 Algorithms and implementations

3 *Temporal Networks*

3.1 Definition and basic properties

In this section, we will introduce the notion of temporal networks or graphs. This is a complex notion, with many concurrent definitions and interpretations. First, we restate the standard definition of a non-temporal, static graph.

Definition 3.1 (Graph)**.** A *graph* is a couple $G = (V, E)$, where V is a finite set of *nodes* (or *vertices*), and $E \subseteq V \times V$ is a set of *edges*. A *weighted graph* is defined by $G = (V, E, w)$, where $w : E \mapsto \mathbb{R}_+$ is fcalled the *weight function*.

We also define some basic concepts that will be needed later on to build simplicial complexes on graphs.

Definition 3.2 (Clique)**.** A *clique* is a set of nodes where each pair is connected. That is, a clique C of a graph $G = (V, E)$ is a subset of V such that $\forall i, j \in C, i \neq j \implies (i, j) \in E$. A clique is said to be *maximal* if it cannot be augmented by any node.

Temporal networks are defined in the more general framework of *multilayer networks* [3]. However, this definition is much too general for our simple applications, and we restrict ourselves to edge-centric time-varying graphs [2]. In this model, the set of nodes is fixed and doesn't change over time, whereas edges can appear or disappear at different timestamps.

Definition 3.3 (Temporal network)**.** A *temporal network* (or graph) is a tuple $G = (V, E, \mathcal{T}, \rho)$, where:

- V is a finite set of nodes,
- $E \subseteq V \times V$ is a set of edges,
- $\mathbb{T}$ is the *temporal domain* (often taken as $\mathbb{N}$ or $\mathbb{R}_+$), and $\mathcal{T} \subseteq \mathbb{T}$ is the *lifetime* of the network,
- $\rho : E \times \mathcal{T} \mapsto \{0, 1\}$ is the *presence function*, which determines whether an edge is present in the network at each timestamp.

The *available dates* of an edge are the set $\mathcal{I}(e) = \{t \in \mathcal{T} : \rho(e, t) = 1\}$.

Temporal networks can also have weighted edges. In this case, it is possible to have constant weights (edges can only appear or disappear over time, and always have the same weight), or time-varying weights. In the latter case, we can set the domain of the presence function to be $\mathbb{R}_+$ instead of $\{0, 1\}$, where by convention a zero weight corresponds to an absent edge.

Definition 3.4 (Additive temporal network)**.** A temporal network is said to be *additive* if for all $e \in E$ and $t \in \mathcal{T}$, if $\rho(e, t) = 1$, then $\forall t' > t, \rho(e, t') = 1$. Edges can only be added to the network, never removed.

3.2 Examples of applications

3.3 Network partitioning

Temporal networks are a very active research subject, leading to multiple interesting problems. The additional time dimension adds a significant layer of complexity that cannot be adequately treated by the common methods on static graphs.

Moreover, data collection can lead to large amount of noise in datasets. Combined with large dataset sized due to the huge number of data points for each node in the network, temporal graphs cannot be studied effectively in their raw form. Recent advances have been made to fit network models to rich but noisy data [4], generally using some variation on the expectation-maximization (EM) algorithm.

One solution that has been proposed to study such temporal data has been to *partition* the time scale of the network into a sequence of smaller, static graphs, representing all the interactions during a short interval of time. The approach consists in subdividing the lifetime of the network in *sliding windows* of a given length. We can then "flatten" the temporal network on each time interval, keeping all the edges that appear at least once (or adding their weights in the case of weighted networks).

This partitioning is sensitive to two parameters: the length of each time interval, and their overlap. Of those, the former is the most important: it will define the *resolution* of the study. If it is too small, too much noise will be taken into account; if it is too large, we will lose important information. There is a need to find a compromise, which will depend on the application and on the task performed on the network. In the case of a classification task to determine periodicity, it will be useful to adapt the resolution to the expected period: if we expect week-long periodicity, a resolution of one day seems reasonable.

Once the network is partitioned, we can apply any statistical learning task on the sequence of static graphs. In this study, we will focus on classification of time steps. This can be used to detect periodicity, outliers, or even maximise temporal communities.

3.4 Persistent homology for networks

We now consider the problem of applying persistent homology to network data. An undirected network is already a simplicial complex of dimension 1. However, this will not be sufficient to capture enough topological information: we need to introduce higher-dimensional simplices. The first possible method is to project the network on a metric space [5], thus transforming the network data into a point cloud data. For this, we need to compute the distance between each pair of nodes in the network (via shortest path distance for instance). This also requires the network to be connected.

Another usual method for weighted networks is called the *weight rank clique filtration* (WRCF) [6], which filters the network based on weights. The procedure works as follows:

1. Set the set of all nodes, without any edge, as filtration step 0.
2. Rank all edge weights in decreasing order $\{w_1, \dots, w_n\}$.
3. At filtration step t, keep only the edges whose weights are less than w_t, thus creating an unweighted graph.
4. Define the maximal cliques of the resulting graph to be simplices.

At each step of the filtration, we construct a simplicial complex based on cliques: this is called a *clique complex*. It is necessarily valid since a subset of a clique is necessarily a clique itself, and the same is true for the intersection of two cliques.

This leads to a first possibility for applying persistent homology to temporal networks. It is possible to segment the lifetime of the network into sliding windows, creating a static graph on each window by retaining only the edges available during the time interval. We can then apply WRCF on each static graph in the sequence, obtaining a filtered complex for each window, to which we can then apply persistent homology.

This method can quickly become very computationally expensive, as finding all maximal cliques (using the Bron-Kerbosch algorithm for example) is a complicated problem in itself. In practice, we often restrict the search to cliques of dimension lower than a certain bound d_M. With this restriction, the new simplicial complex is homologically equivalent to the original: they have the same homology groups up to H_{d_M-1}.

This method is sensitive to the choice of sliding windows on the time scale. The width and the overlap of the windows can completely change the networks created and their topological features. Too small a window, and the network becomes too small to have any significant topological properties, too large, and we lose important information in the evolution of the network over time.

3.5 Zigzag persistence

4 *Persistent Homology for Machine Learning applications*

4.1 Vectorization methods

4.2 Kernel-based methods

4.3 Comparison

Bibliography

[1] Gunnar Carlsson. “Topology and data”. In: *Bulletin of the American Mathematical Society* 46.2 (Jan. 29, 2009), pp. 255–308. ISSN: 0273-0979. DOI: 10.1090/S0273-0979-09-01249-X. URL: http://www.ams.org/journal-getitem?pii=S0273-0979-09-01249-X (visited on 11/03/2017).

[2] Arnaud Casteigts et al. “Time-varying graphs and dynamic networks”. In: *International Journal of Parallel, Emergent and Distributed Systems* 27.5 (Oct. 1, 2012), pp. 387–408. ISSN: 1744-5760. DOI: 10.1080/17445760.2012.668546. URL: https://doi.org/10.1080/17445760.2012.668546 (visited on 02/21/2018).

[3] Mikko Kivelä et al. “Multilayer Networks”. In: *Journal of Complex Networks* 2.3 (Sept. 1, 2014), pp. 203–271. ISSN: 2051-1310, 2051-1329. DOI: 10.1093/comnet/cnu016. arXiv: 1309.7233. URL: http://arxiv.org/abs/1309.7233 (visited on 02/13/2018).

[4] M. E. J. Newman. “Network structure from rich but noisy data”. In: *Nature Physics* 14.6 (June 2018), pp. 542–545. ISSN: 1745-2481. DOI: 10.1038/s41567-018-0076-1. URL: https://www.nature.com/articles/s41567-018-0076-1 (visited on 07/10/2018).

[5] Nina Otter et al. “A roadmap for the computation of persistent homology”. In: *EPJ Data Science* 6.1 (Dec. 1, 2017), p. 17. ISSN: 2193-1127. DOI: 10.1140/epjds/s13688-017-0109-5. URL: https://link.springer.com/article/10.1140/epjds/s13688-017-0109-5 (visited on 04/12/2018).

[6] Giovanni Petri et al. “Topological Strata of Weighted Complex Networks”. In: *PLoS ONE* 8 (June 1, 2013), e66506. DOI: 10.1371/journal.pone.0066506. URL: http://adsabs.harvard.edu/abs/2013PLoSO...866506P (visited on 04/20/2018).

[7] Afra Zomorodian and Gunnar Carlsson. “Computing Persistent Homology”. In: *Discrete & Computational Geometry* 33.2 (Feb. 1, 2005), pp. 249–274. ISSN: 0179-5376, 1432-0444. DOI: 10.1007/s00454-004-1146-y. URL: https://link.springer.com/article/10.1007/s00454-004-1146-y (visited on 04/16/2018).