ABSTRACT

GAMBLE, JENNIFER PAMELA. Complex and Dynamic Network Analysis: A Topological Perspective. (Under the direction of Dr. Hamid Krim.)

This dissertation develops and applies topological methods for the analysis of complex and dynamic networks. In contrast to the term 'topology' used in traditional network analysis, here we use the term in the sense of algebraic topology. The field of topological data analysis (TDA) has blossomed in recent years, allowing researchers to analyze point cloud data sets using methods that take into account the 'shape' of the data. This includes geometrical features, as well as topological features such as connected components, loops, and voids. These methods have been applied with great success to data sets which are naturally embedded in a metric space (such as Euclidean space), because distances between points can be used to form a parameterized sequence of spaces, and studying the changing topology of this sequence gives information about both the topology and geometry of the data under analysis.

In our setting, the input data are simply graphs, consisting of vertices connected by (undirected, unweighted) edges, with no underlying metric other than the graph distance between vertices. We demonstrate how one can still consider the 'shape' of such objects in a topologically-informed way, using a simplicial complex representation, and that such a viewpoint has great advantages.

We first apply this methodology to analyzing coverage properties in dynamic sensor networks. The dynamic sensor network under consideration is studied through a series of snapshots, and is represented by a sequence of simplicial complexes, built from the communication graph of the network at each time point. A method from TDA called zigzag persistent homology takes this sequence of simplicial complexes as input, and returns a 'barcode' containing the birth and death times of topological features in this sequence. We derive useful statistics from this output for analyzing time-varying coverage properties.

In addition, we develop a method which returns specific representative cycles for these homological features, at each point along the birth-death intervals. These representative cycles are then used to track coverage holes in the network, and obtain size estimates for individual holes at each time point. A weighted barcode, incorporating the size information, is then used as a visual and quantitative descriptor of the dynamic network coverage.

Finally, we take a topologically-motivated approach to social network analysis, through the local property of node dominance (which was developed in relation to a strong homotopy collapse of a simplicial complex). By iteratively applying node dominance collapses, we are able to obtain a core-periphery decomposition of a network, where the nodes in the core are essential for network flow, community structure, and the global structure of the network. Additionally, the peripheral components are seen to have applications for community detection, and we propose an algorithm which uses them to obtain "candidate sets" which are meant to approximate communities or unions of communities. This community detection method is seen to have better performance than a state-of-the-art algorithm for overlapping community detection on two large, real-world networks with ground-truth community information.

Complex and Dynamic Network Analysis: A Topological Perspective

by
Jennifer Pamela Gamble

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Electrical Engineering

Raleigh, North Carolina

2015

APPROVED BY:

Huaiyu Dai	Patricia Hersh
Brian Hughes	Hamid Krim Chair of Advisory Committee

DEDICATION

To Chris. You are my home.

BIOGRAPHY

Jennifer Pamela Gamble was born in 1984 in Victoria, British Columbia to parents Donna Ellsay and Ken Gamble. She grew up in Calgary, Alberta, and graduated from Queen Elizabeth Senior High School in 2002. Next came Edmonton, with a BSc in Mathematics (2006) and an MSc in Statistics (2008) from the University of Alberta. Her Masters' was under the supervision of Dr. Giseon Heo, who Jennifer continued to work with from 2008-2010 as a research assistant. In 2009, she married Christopher Avanthey, in Edmonton, Alberta, and in 2010 they moved to Raleigh, North Carolina where Jennifer began her PhD at North Carolina State University under Dr. Hamid Krim. In 2013, Jennifer and Chris had a daughter, Lucy, who is a complete joy. After graduating from her PhD in 2015, Jennifer and the family are moving to Menlo Park, California, where Jennifer will begin work as a data scientist at the company Ayasdi.

ACKNOWLEDGEMENTS

I first have to acknowledge my advisor, Dr. Hamid Krim, without whom this would not have been possible. I especially valued the format of his weekly group meetings, which required us to express our ideas clearly and regularly, and greatly increased our breadth of knowledge by keeping on top of the research of our labmates. I know of very few university professors who devote this many hours per week to meeting with their students. Additionally, his bi-weekly seminar series brought countless world-class researchers to the department, and having the opportunity to meet with them not only yielded many helpful insights, but required us to become comfortable presenting our work to professional audiences. I would also like to thank the rest of my advisory committee: Dr. Huaiyu Dai, Dr. Brian Hughes, and Dr. Patricia Hersh, as well as Dr. Yannis Vinotis for substituting for Dr. Hughes during my final defense. I greatly value your time and advice, and your comments and suggestions along the way have always been helpful and insightful.

The person who has had the greatest impact on my research life is my onetime labmate, frequent co-author, and dear friend, Harish Chintakunta. He has the true inquiring spirit all academics should aspire towards, and in many ways he is the one who taught me, both directly and indirectly, how to do research. I often question whether I would have even completed my PhD, had it not been for Harish.

The daily life of a graduate student is in the lab, and as such, I have been very happy to share it with Scott Clause, Tian Wang, Xiao Bian, Adam Wilkerson, Saba Emrani, Deokwoo Lee, Sheng Yi, Shahin Mahdizadehaghdam, Hui Guan, Sally Ghanem, and Erik Skau. Over the years we developed many tight bonds, and I really value both their camaraderie and amazing intellectual resources. I was also pleased to work with and learn from our postdocs, Athanasios Gentemis, and Han Wang, who are both great people as well.

I have had the benefit of interacting with a number of academic role models, and I have appreciated their professional and personal advice, including Vin de Silva, Radmila Sazdanovic, Robert Ghrist, Matthew Kahle, Sayan Mukherjee, Hatice Ozturk, and Cranos Williams. Seeing professors who are so genuinely engrossed in their work made academic life feel really possible, and I hope to approach my work with the same thoughtfulness and passion that they display.

My parents Ken and Donna, are the personification of unconditional love. The fact that my two sisters and I are all profoundly happy, doing completely different things, on opposite sides of the world, speaks to our parents love and guidance in encouraging us to follow our own paths. Katie and Meggie, I love you too! Thank goodness for Skype. I'd also like to thank Tia Halliday (my bff), and my wonderful in-laws Barb and Gabby for all their love and support.

My sweet Lucy, who made me a mother. You have added so much joy to my life, with a power I couldn't have anticipated.

Finally, and most importantly, is Chris Avanthey. The love of my life. I can't believe how lucky I am to have you. The words patient, understanding, supportive and encouraging, do you no justice. Always a calming presence when I'm under stress, and the best cook I know. I dedicate this work to you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER

1

INTRODUCTION

The fundamental viewpoint motivating this work is that incorporating topological features when analyzing complex data can yield surprising and informative results about the data's structure, which are not obtainable from other methods. In the case of network analysis, the use of higher-order information, such as n-tuple relationships between nodes, is encoded using simplicial complexes. This allows for topological information to be considered, which opens access to a wealth of mathematical tools.

Here, we apply this philosophy to two problems in complex network analysis. The first is the study of coverage properties in time-varying sensor networks, and the second is community detection and core-periphery decomposition in social networks. These seemingly disparate problems both have topologically-informed solutions. In the first case, we are able to obtain a quantification of the time-varying coverage of a network in a completely coordinate-free manner, using only the adjacency matrix for the communication graph at each time point. In the second case, the local property of node dominance yields a distributed algorithm for computing a core-periphery decomposition of a social network, where the core is shown to be essential for the network in terms of network flow and global structure, and additionally the peripheral components give information about the community structure that yields an effective algorithm for community detection.

“Topological data analysis” (TDA), is a somewhat broad term which can be meant to describe any data analysis methods which use a topological space (or sequence of spaces) built from the data, and use features of this space (topological, geometric, or other) to describe the data set. This is often interpreted as studying the ‘shape’ of the data, because topological features include things like the number of connected components, loops, or voids in a space, and because the topological space is usually constructed using a function on the metric space the data lie in (incorporating geometric information). As a field of study, TDA was not born until nearly the turn of the 21st century. Prior to that, algebraic topology was viewed as a pure form of mathematics, or ’math for math’s sake’, without express purpose for real-world applications. Since it is a field of mathematics which uses algebraic objects to study topological spaces, algebraic topology is very broad and can become quite abstract. Later on, when we discuss the special case of simplicial homology theory in Section 2.2.1, we see that it will become quite concrete (reducing to linear algebra computations), and extremely useful for our applications.

The advent of the persistent homology algorithm [16] [45] and its mathematical formalization [62] are often considered the turning point, which allowed topological features to be considered applicable to the study of data sets (although some earlier methods in computational geometry [17] and size theory [19] had a topological flavor as well). Many of the relevant notions had been available for years: a Čech or Vietoris-Rips complex could be used to turn a data set into a simplicial complex; and simplicial homology theory could be used to compute the topological features (or *homology*) of a given simplicial complex. The problem was that building the simplicial complex required a choice of parameter value, and that the resulting topological features are extremely non-robust to perturbations in the underlying data set. The key to persistent homology, was that it took a multi-resolution approach by considering a *sequence* of simplicial complexes, over a range of parameter values. Studying the changing homology of the sequence gave information about the topology *and* geometry of the data set, and the summary for this changing sequence of spaces (crucially) also had good stability properties [12] [11]. A second algorithm which has become synonymous with TDA is the mapper algorithm [49], which also builds a simplicial complex to represent a dataset, but in this case the complex obtained is dependent on the choice of a function, so one may study a single data set through different "lenses" by using different choices of functions. For some excellent survey articles of persistent homology and topological data analysis, see Ghrist [23] and Carlsson [6]. Currently, the theory and applications of TDA are predominantly oriented towards the persistent homology and mapper algorithms and their generalizations. It is important to note that these methods are motivated by applications in data analysis, where the data lie in a metric space (often $\mathbb{R}^n$), and it is natural to use parameter and function values relative to this metric.

In this dissertation, the data structures we consider are fundamentally different from the point cloud data in typical TDA applications. We consider network (or graph) structures, which consist only of vertices, and edges connecting them. They are combinatorial in nature, and for our applications the edges are unweighted, undirected, and connect two distinct vertices, so the only notion of distance between two vertices is the graph distance between them. We propose and explore applications of methods for network analysis which are topologically-motivated, and we see that, as in the case with traditional TDA methods on point cloud data, the incorporation of topological features and ideas into the existing network analysis toolbox yields greater insights into the structure of our data, with fewer assumptions for the input (in the sensor network case) and more computational efficiency (in the social network setting).

First, in Chapter 2 we consider the problem of localizing and tracking coverage holes in dynamic sensor networks, with the additional constraint that no node locations or inter-node distances (edge lengths) are known. Only binary information about which nodes are within communication range at each time point is available, and through a combination of methods from computational topology, along with a novel algorithm, we are able to obtain a quantitative descriptor of the dynamic network coverage, which includes the number of holes at each time point, as well as estimated hole size and duration.

The algorithm is described in detail in Chapter 3. It takes the sequence of simplicial complexes, and chooses specific representative cycles for the homology classes at each time point, which are geometrically-relevant.

Finally, in Chapter 4 we turn our attention to study large-scale social networks. Here, we use topology-preserving collapses [4] [54] of the network to identify nodes which belong to a 'core', and develop theoretical results involving the properties of this core and the remaining periphery. The nodes in the core are seen to be very important to the global structure of the network, as well as network flow. Moreover, the peripheral components are related to the community structure of the network, and we propose an algorithm for their use in community detection, which is seen to perform well against a state-of-the-art method for overlapping community detection in large networks. Results are placed in the context of existing theories of social network structure, and support the view that overlapping communities yield a core-periphery network structure.

In Chapter 5, we conclude by discussing the implications of our work, and propose some directions for future research.

CHAPTER

2

COORDINATE-FREE QUANTIFICATION OF COVERAGE IN DYNAMIC SENSOR NETWORKS

The paper this chapter is based on was published in *Signal Processing*. See [22] or arXiv:1411.7337.

2.1 Introduction

Wireless sensor networks gained attention and popularity when technological advances allowed for the development of small, low-cost wireless sensors. These simple devices could be distributed over a region, with each sensor (or 'node') gathering data about its local environment for purposes of monitoring, detecting or reporting. In recent years, the study of wireless sensor networks has significantly increased, with research into methodologies for the different layers of the sensor network protocol stack (physical, data link, network, transport and application layers), each developing into their own sub-field. Areas of application include military, industrial, and environmental monitoring and tracking. See [3] and [60] for surveys of the field.

A particular problem in sensor networks which quickly gained research interest is the so-called 'coverage problem' [28]. Given a set of (typically homogeneous) sensors, each with the ability to sense some region of immediate proximity to it, one wishes to make statements about the sensing ability of the entire network, taken as a whole. An initial question is whether every point in a region of interest is covered by at least one sensor. As sensor networks developed, it was no longer realistic to assume a static network, and node mobility became a factor in network analysis and design. It became clear that mobility of nodes could be considered for initial deployment [44] [27], as well as for improving coverage over time [36]; thus, the development of methods to study dynamic, or time-varying sensor networks has become increasingly important.

A number of methods for determining area coverage were developed, and for efficiently deploying nodes to provide complete or optimal coverage, see [50] for a survey. Such methods require geometric information about the locations of the sensors, or their distances from each other, in addition to information about the geometry of the coverage area for each sensor. Methods from computational and stochastic geometry have been used to study the coverage properties of dynamic sensor networks when complete geometric information is available [43]. The coverage is described using statistics such as the proportion of uncovered area at each time point, or the proportion uncovered over a time interval (where a point is considered covered if it is covered at any time during the interval). These descriptors have been used to analyze and compare various mobility models for dynamic networks, to determine advantages and disadvantages of each, as well as optimal strategies for intruder detection [37].

The availability of geometric information, such as global coordinates for the nodes, or distances between them, is often an overassumption. Instead, 'coordinate-free' methods compute network properties using only local, binary information about which nodes are within communication range of each other. De Silva and Ghrist [47] were the first to propose a rigorous method for determining coverage which did not require location or distance information, by invoking tools from simplicial homology theory (see Section 2.2 for details). Such homological methods are able to give guarantees that a network is covered at a single time point, or over a time interval, using only coordinate-free data.

Other researchers have used coordinate-free data to study network coverage by detecting approximate boundaries of coverage holes in static networks. Some methods (such as in [30] or [35]) define interior nodes using specifically structured sub-graphs ('flowers' or '3MeSH rings', respectively), while another method defines boundary nodes by using breaks in iso-contours formed by hop distance from a base node [20]. One method estimates the boundary by using a multi-step procedure built using the cuts in a shortest path tree which 'forks' around coverage holes [51]. All of these methods can obtain good experimental results, but are relatively dependent on the network having a high density, so the holes are large compared to the distances between neighboring sensors [29].

In this chapter, we consider the study of coverage properties of sensor networks which are both coordinate-free and time-varying. Information from the network is available as a series of discrete-time snapshots, where each node returns a list of other nodes which are within its local area. In so doing, we compute the number of coverage holes at each time point, as well as information about estimated hole sizes, and how the holes persist over time. This information is summarized in a 'barcode' describing the birth and death times of homological features in the network over time, and we describe the relationship between these features and the coverage properties. The barcode is obtained by employing a method from the mathematical field of computational topology, called zigzag persistent homology ([8], [7]). We also propose an additional algorithm which returns specific cycles in the network characterizing the coverage holes over time, which aid in estimating the size of the holes.

The method we describe here is the only one currently available which can quantify the coverage dynamics in a coordinate-free network. We will also see that it correlates well with other coverage measures which utilize full geometric information. Further, the barcode includes information about how coverage holes form, merge, split and close in the time-varying network, which is not available using existing methods (whether geometric information is included or not). In the past, homological methods have been able to give guarantees that a network is covered at a single time point, or over a time interval, while geometric methods have been used to obtain summary statistics which describe the time-varying nature of the network coverage. Here, we use homological, coordinate-free methods to obtain a descriptor of the dynamic network coverage.

As our primary contributions, we propose how the 'barcode' output from zigzag persistence can be used as a quantitative descriptor of time-varying coverage in a network, and moreover describe an algorithm we developed for choosing a specific geometrically-relevant cycle for each coverage hole in the network at each time point. The utility of the barcode is illustrated by using it to quantify and compare coverage dynamics for different models of sensor mobility. Our novel representative cycles are used in conjunction with a hop distance-based method to obtain size estimates for the holes, and this information is incorporated back into the barcodes, giving a visual and quantitative summary of the dynamic network coverage. Further examples demonstrate the effectiveness of this descriptor in tracking small coverage holes appearing in dense networks, in identifying expanding failure regions, and in monitoring the maintenance of a protective barrier of mobile sensors around a guarded region.

The organization of this chapter is as follows: In Section 2.2 we will first describe the basics of simplicial homology, and how it has been effectively used to give global coverage guarantees for both static and dynamic coordinate-free networks. In Section 2.3 we will outline our primary computational tool, zigzag persistent homology, and describe the additional types of coverage results it allows. Section 2.4 details the hop distance-based filtration, and its use in estimating hole sizes for a given simplicial complex. Section 2.5 provides an outline of our method for obtaining specific representative cycles (which will be described in full detail in Chapter 3), and how these cycles can be used with the hop distance filtration to enhance the barcode with estimated size information for each bar at each time point. This is followed by examples illustrating the utility of the method, and by concluding remarks.

2.2 Preliminaries

The adopted sensor network coverage model assumes homogeneous, isotropic sensors with sensing radius r, so that each sensor is at the center of its associated coverage region, which is a disk of radius r. This 'Boolean disk coverage model' is the most widely used sensor coverage model in the literature [50]. Throughout this chapter, we will assume that the network consists of n sensors, indexed 1 through n. If sensor i is located at $\mathbf{x}_i \in \mathbb{R}^2$, then denote the disk of radius r centered at $\mathbf{x}_i$ as $B(\mathbf{x}_i, r)$. Then the coverage region $\mathscr{R}$, for the entire network, is the union of all such disks:

$$\mathscr{R} = \bigcup_{i=1}^{n} B(\mathbf{x}_i, r). \tag{2.1}$$

To study the coverage holes appearing in $\mathscr{R}$ two concepts are useful: the concept of *homology*, and that of representing a sensor network with a *simplicial complex*. Homology is a mathematical method which, intuitively, is used to define and categorize holes in spaces, (which are exactly the features of interest here, and are called *topological features*). Thus, coverage analysis reduces to analysis of the topology of the space $\mathscr{R}$. The tools for this analysis are well established in algebraic topology, and aim at quantifying the topology of a space by assigning algebraic invariants called homology groups. Representing a space as a simplicial complex (which can be achieved using local information only), provides a discrete combinatorial representation enabling computations of the homology groups. Thus in a sensor network setting, guarantees can be made about coverage for the entire network [47], using reasonably coarse local information. The local information required at each sensor is simply a list of the other nodes within a known communication range.

2.2.1 Simplicial homology

The theory of homology has a long and rich history, with results available in much greater generality than necessary for our purposes here (see [25] for a good introduction to algebraic topology, including homology theory). The situation we will be considering is when the spaces under analysis are simplicial complexes, yielding matrix calculations for computing homology. First, we define a simplicial complex, and its homology.

Definition: A k*-simplex* is a set of $k+1$ vertices, or singleton elements. Any subset of the $k+1$ vertices forming a simplex is called a *face* of the simplex, where each face is, itself, also a simplex.

A *simplicial complex,* K, is a set of simplices such that any simplex in K also has all of its faces in K.

A simplicial complex can be thought of as a higher-dimensional analogue to a graph. Although simplicial complexes can be represented purely abstractly as a collection of sets of vertices (as above), they are also often defined or visualized as being embedded in a Euclidean space. These geometric simplicial complexes must additionally satisfy the requirement that the intersection of any two simplices is a face of each of them. The geometric setting is useful because vertices are represented as specific coordinate points, and simplices as their convex hull. In that case, a 0-simplex is a vertex (also called a node), a 1-simplex is an edge between two vertices, a 2-simplex is a triangle, and higher dimensional simplices are defined analogously. For computational purposes the abstract combinatorial representation is used, because its discrete nature lends itself well to compact storage and calculations. In particular, this representation allows for straightforward computation of homology. Figure 2.1 shows 0-, 1-, 2-, and 3-dimensional simplices (left), and an example of a small simplicial complex (right), with vertices labeled and orientations indicated on the edges.

Definition: *(Homology)* Given a simplicial complex K we build the *chain spaces* C_0, C_1, C_2, ..., where C_k is the vector space formed by using the k-simplices as basis elements. We then encode information about the specific structure of the simplicial complex in the *boundary maps* $\partial_1, \partial_2, \ldots$, where

$$\partial_k : C_k \to C_{k-1}$$

describes explicitly how the k-simplices are connected to the $(k-1)$-simplices. For k-simplex $\sigma = [v_0, v_1, \ldots, v_k]$, the boundary map ∂_k maps σ onto the alternating sum of its faces:

$$\partial_k \sigma = \sum_{i=0}^{k} (-1)^i [v_0, \ldots, \hat{v}_i, \ldots, v_k]$$

where $\hat{v}_i$ indicates the vertex v_i removal. Note that the above definition of the boundary operator depends on the initial ordering of the simplex, which is referred to as *orientation*. The simplices are assigned arbitrary orientations. Then the k^{th} *homology group* is defined to be

$$H_k(K) = \ker(\partial_k)/\mathrm{im}(\partial_{k+1})$$

and the k^{th} *Betti number* (denoted β_k) of the simplicial complex K is the rank of $H_k(K)$.

To understand this definition, let us look at what $\ker(\partial_k)$ and $\mathrm{im}(\partial_{k+1})$ mean individually. In general, ∂_k maps a k-simplex σ onto its boundary (which is made up of $(k-1)$-simplices), so if $\sigma = [v_i, v_j]$ is an edge, then $\partial_1 \sigma = v_j - v_i$ is the difference of σ's vertices. Similarly, if $\sigma = [v_i, v_j, v_k]$ is a triangle (a 2-simplex), then $\partial_2 \sigma = [v_j, v_k] - [v_i, v_k] + [v_i, v_j]$ is the alternating sum of its edges. An element c in the chain space C_k is just a linear combination of k-simplices $\sigma_1, \ldots, \sigma_{n_k}$,

$$c = \sum_{i=1}^{n_k} a_i \sigma_i$$

and can be written as a vector $c = [a_1, \dots, a_{n_k}]$ of length $n_k = (\#\text{ of } k\text{-simplices in } K)$. The coefficients a_i come from a field $\mathbb{F}$ (such as the real numbers), but we choose to perform our computations over the field $\mathbb{Z}_2 = \{0, 1\}$ (in this case, the interpretation is that simplices with nonzero coefficients are the ones present in the chain c). The boundary operator ∂_k is written as a $n_{k-1} \times n_k$ matrix, so the computation of the boundary for any chain reduces to the matrix multiplication $\partial_k c$. Any chain with boundary zero (i.e. any c such that $\partial_k c = 0$) is called a *cycle*, and so $\ker \partial_k$ is the set of all k-cycles. In particular, the boundary of a simplex will form a cycle, which implies that all boundaries are themselves cycles (i.e. $\text{im}\partial_{k+1} \subseteq \ker \partial_k$). This also implies the general property that $\partial_k \partial_{k+1} = 0$. We can now reinterpret the definition of homology as "cycles which are not boundaries".

Definition: Two cycles c_1 and c_2 are *homologous* (written $c_1 \sim c_2$) if their difference can be written as a linear combination of boundaries. The set of all cycles that are homologous to a given cycle (say c) is called a *homology class* (denoted $[c]$). All cycles in the same homology class will surround exactly the same hole (or set of holes). When a specific cycle is chosen to represent an entire homology class, it is called a *representative cycle*. The span of the homology classes defined by k-cycles form the k^{th} homology space.

It is in this sense that the rank of the k^{th} homology group (the Betti number β_k) counts the number of k-dimensional 'holes' in the simplicial complex. Intuitively, β_0 counts the number of connected components, β_1 counts the number of 'holes' as we normally think of them (empty regions that one can form a loop around), β_2 counts the number of enclosed voids, and higher-dimensional homology is defined analogously.

In Figure 2.1, the cycle formed by edges $[v_2, v_3]$, $[v_3, v_4]$, and $[v_2, v_4]$ is the boundary of the triangle $[v_2, v_3, v_4]$, and thus is equivalent to zero (trivial) with respect to homology. The cycle formed by edges $[v_1, v_2]$, $[v_2, v_4]$, $[v_4, v_5]$, and $[v_1, v_5]$, which we denote by c, cannot be written as the boundary of triangles, and is thus non-trivial with respect to homology. Note also that the non-trivial cycle c, is homologous to the cycle formed by edges $[v_1, v_2]$, $[v_2, v_3]$, $[v_3, v_4]$, $[v_4, v_5]$, and $[v_1, v_5]$.

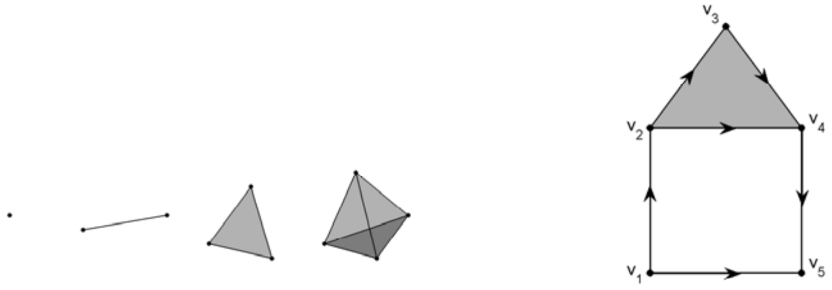

Figure 2.1 Simplices and a small simplicial complex

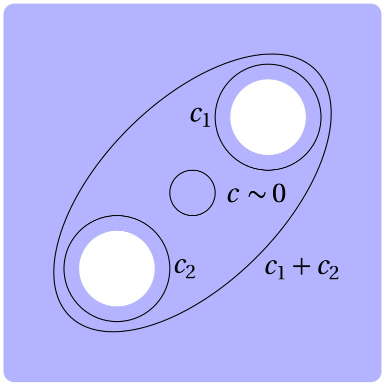

Figure 2.2 Topological space with first homology of dimension two.

A final concept to highlight is that of a *homology basis*. As seen in the above definitions, given a simplicial complex K, the k^{th} homology group $H_k(K)$ is a vector space of dimension β_k, and therefore any linearly independent set of β_k homology classes form a basis for $H_k(K)$. As an example, consider Figure 2.2, which illustrates a space with two holes (so $\beta_1 = 2$). The cycle c does not surround any holes, and is homologous to zero (i.e. it is trivial). The homology class $[c_1]$ contains all cycles which surround only the righthand hole, and is represented by cycle c_1. Similarly, c_2 represents the homology class of cycles surrounding the lefthand hole. Note that the cycle $c_1 + c_2$ is homologous to the sum of the cycles c_1 and c_2. Thus, this space has three distinct, non-trivial homology classes: $[c_1]$, $[c_2]$, and $[c_1 + c_2]$, any two of which form a basis for the first homology (eg. $\{[c_1],[c_2]\}$ form a basis, as does $\{[c_1],[c_1 + c_2]\}$). Given a compact region of the plane, such as the one shown, there exists a *canonical basis* for its first homology, namely the basis with one homology class surrounding each of the holes ($[c_1]$ and $[c_2]$ in our example). This result is a specific case of the more general principle of Alexander Duality (see, for example Ch. 5 of [39]). The concept of a canonical homology basis will become relevant for us again in Section 2.5, where we describe our method for choosing a set of representative cycles in an attempt to approximate the canonical basis for the coverage area of a sensor network.

2.2.2 Simplicial complex representation of a sensor network

For the purposes of analyzing the coverage region of a sensor network, we are interested in computing the homology of $\mathscr{R}$ (the coverage region for the network - defined in Equation (2.1)). Specifically, we are interested in $\beta_1 = \text{rank}(H_1(\mathscr{R}))$, the rank of the first homology group, to determine how many holes are present in the network. Given a set of sensors, proceed to build a simplicial complex by using the sensors as vertices, and adding higher dimensional simplices (edges, triangles, etc) between them on the basis of the distances between the sensor vertices. Two common ways to build a simplicial complex from a set of points entail the use of the Čech, and the Vietoris-Rips complexes. For the following definitions, we assume vertex v_i corresponds to sensor i, which has location $\mathbf{x}_i \in \mathbb{R}^2$, and the disk of radius r centered at $\mathbf{x}_i$ is denoted $B(\mathbf{x}_i, r)$.

Definition: A *Čech complex* contains the k-simplex formed by vertices $\{v_0, v_1, \ldots, v_k\}$ whenever

$$\bigcap_{i=0}^{k} B(\mathbf{x}_i, r) \neq \emptyset.$$

Definition: A *Vietoris-Rips complex* (also referred to as a *Rips complex*) includes the k-simplex formed by vertices $\{v_0, v_1, \ldots, v_k\}$ whenever

$$B(\mathbf{x}_i, r) \cap B(\mathbf{x}_j, r) \neq \emptyset \text{ for all } 0 \leq i < j \leq k.$$

In other words, the Čech complex contains the higher-dimensional simplex formed by a group of sensors whenever all the coverage disks of those sensors have a nonempty intersection, and the Rips complex contains the higher-dimensional simplex whenever the coverage disks of a group of sensors all intersect pairwise. The coverage region formed by the union of coverage disks for a sensor network is shown in Figure 2.3 (left), with the associated Rips complex (right). Note that computation of the $(k+1)$-wise intersection of disks in the Čech definition requires precise geometric information about the relative locations $\mathbf{x}_i$ of the sensors. For the Rips complex, on the other hand, once edges are formed between all sensors of distance less than $2r$, the information about which higher-dimensional simplices to include directly follows. This is equivalent to requiring only the binary information contained in the adjacency matrix for the communication graph (where sensors can communicate whenever they are within distance $2r$ from each other). Both the Čech and Rips complexes depend on the choice of parameter r, and for a given value of r, the two complexes will differ precisely when a set of sensors are all pairwise within $2r$, but do not all intersect at any point. A 2D example of when the Čech and Rips complexes will differ is shown in Figure 2.4. Since the three coverage disks intersect pairwise, but have no triplet-wise intersection (leaving a small area uncovered), the associated triangle will be in the Rips complex, but not in the Čech (thus, the

Čech complex reflects the true homology of the coverage region).

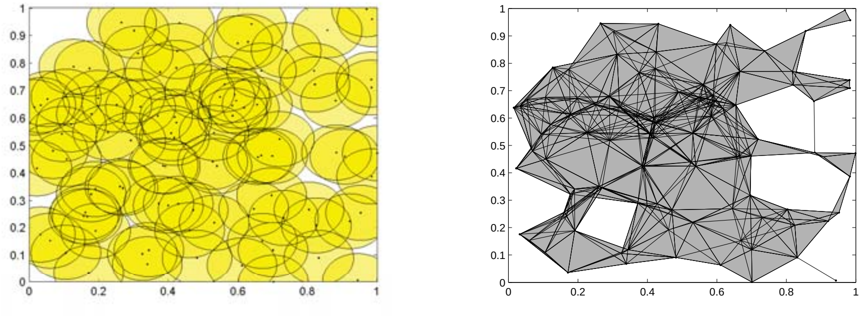

Figure 2.3 Coverage region and Rips complex for a sensor network.

The configuration displayed in Figure 2.4 also illustrates one of the properties of the Čech complex: it has the exact same homology (number of holes) as the coverage region $\mathscr{R}$, while the Rips complex can 'miss' such small coverage holes. The worst-case detection of missed area is when the three nodes form an equilateral triangle with edge lengths $2r$, is witnessed network-wide when the sensors lie on a hexagonal lattice. In this case the holes account for $\sim 7\%$ of the total area, and are not detected by the Rips complex. In practice, when the nodes are distributed uniformly and randomly, we found the holes missed by the Rips complex amount to $\ll 1\%$ of the total area (simulations over a range of network sizes and node densities showed instances where the area of the 'missed' holes was up to 0.15% of the total area, but more typically they accounted for less than 0.03% of the total area).

The results by De Silva and Ghrist [47] use this simplicial complex representation of a sensor network, and describe a precise relationship between the sensing radius and the communication radius of each node which allows coverage guarantees to be made. The sensing radius defines the coverage region, and the communication radius is used to build the Rips complex used for computing the homology, so their results allow very coarse binary information about pairwise com-

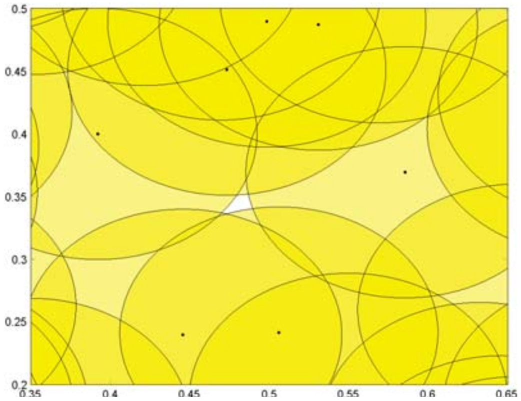

Figure 2.4 Illustration of 'missed' coverage hole.

munication to infer whether global coverage is achieved. They additionally consider a problem in dynamic networks: does an *evasion path* exist and allow an intruder to remain undetected over a time interval?. Their results give conditions which will guarantee that no such evasion path exists.

For our purposes, we will understand that although the holes detected by the first homology of the Rips complex do differ from the holes in the coverage region (in exactly the way described above), the holes which are missed are extremely small relative to the size of the network. We will therefore use the homology computed using the Rips complex as a sufficient approximation. This is a particularly safe assumption in the time-varying case, because a very small hole which remains very small over time is justifiably ignored. Throughout, when we discuss 'network coverage', we are referring to the coverage as characterized by the Rips complex.

An additional note on the use of the Rips complex in characterizing a network: the only assumption that is really required is that whenever three sensors can communicate pairwise, then the entire triangle that they define is considered covered. The assumption that the coverage region is the union of identical coverage disks centered at each node, is thus somewhat stricter than necessary.

We now consider a time-varying network, which again has only pairwise communication information at each time point. We next present a method which, in addition to detecting global coverage, will track homological features over time, and provide information about the number and duration of coverage holes.

2.3 Coverage properties of dynamic networks

2.3.1 Zigzag persistent homology

Zigzag persistent homology is a recently developed computational method to track homological features (such as those described in Section 2.2) through a sequence of spaces. In our problem setting, where sensor networks are represented by simplicial complexes, and the first homology detects coverage holes, we employ this method to tell us about coverage holes in a time-varying sensor network. While we give a brief summary here, we defer to [8] and [7] for complete mathematical and algorithmic details (respectively) of zigzag persistence.

We use zigzag persistent homology to study a sequence of simplicial complexes

$$K_1 \longleftrightarrow K_2 \longleftrightarrow \ldots \longleftrightarrow K_n.$$

Call this sequence $\mathscr{K}$, and assume each map '$\longleftrightarrow$' is an inclusion: either 'forward' as $K_i \longrightarrow K_{i+1}$ or 'backward' as $K_i \longleftarrow K_{i+1}$. This sequence is studied by computing the associated homology spaces to obtain the *zigzag persistence module*

$$H_p(\mathscr{K}) = H_p(K_1) \longleftrightarrow H_p(K_2) \longleftrightarrow \ldots \longleftrightarrow H_p(K_n) \tag{2.2}$$

One of the main theorems in the theory of zigzag persistent homology, is that such a module can be uniquely decomposed. Each $H_p(K_i)$ is a vector space, and the module in Equation (2.2) can be decomposed into a set of 'interval modules', each consisting of one-dimensional vector spaces, for some range $[b, d]$, where $1 \leq b \leq d \leq n$, and zeros outside of this range (see [8] for details). The intervals in this decomposition are interpreted as the lifetimes of individual homological features in the sequence, which are summarized by their birth and death times (b and d). In the sensor network setting, the decomposition of the zigzag persistence module for the first homology gives a list of birth and death times of the one-dimensional homological features in the sequence. These homological features describe the time-varying coverage of the network, in a way described precisely in Section 2.3.2. The multi-set of birth and death times

$$\text{Pers}(\mathscr{K}) = \{[b_j, d_j],\}$$

is the zigzag persistence of our sequence of spaces, and is represented pictorially in two common ways. The first is a *barcode* where the x-axis represents time t, the y-axis represents individual homological features, and each feature is depicted as a horizontal line from its birth time (b_i) to death time (d_i). The second visual representation is a *persistence diagram*, which plots the points (b_i, d_i) on two-dimensional coordinate axes. Thus, all points lie above the diagonal (death occurs after birth), and points further from the diagonal indicate longer lifetimes. Figure 2.5 shows the barcode (left) and persistence diagram (right) corresponding to $\text{Pers}(\mathscr{K}) = \{[2,9],[4,7],[6,8],[9,10]\}$, as an example.

This output of a discrete set of birth and death times for homological features can be used to quantify the time-varying coverage for a given dynamic sensor network, as described in the following section.

2.3.2 Barcodes as descriptors of coverage

Here we describe how the framework of zigzag persistent homology may be adapted to describe information about time-varying coverage in a dynamic sensor network. Section 2.2.2 described how a sensor network is represented as a simplicial complex derived from the communication graph, and

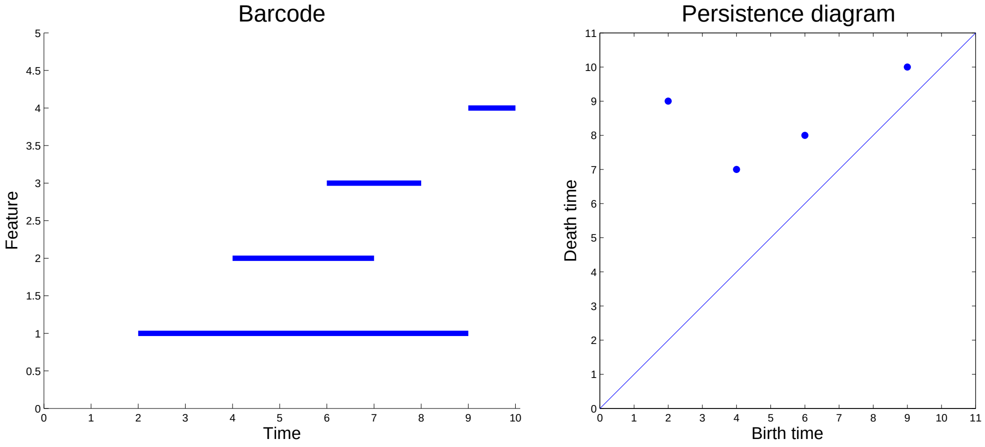

Figure 2.5 Barcode and persistence diagram.

the first homology of this complex is used to determine coverage of the network. We now consider a time-varying sensor network, whose communication graph (and thus its associated simplicial complex) is available at a sequence of discrete time points. It is assumed that each sensor has a unique node identification number in $\{1,\ldots,n\}$, and so a correspondence can be made between the simplicial complex at one time point and the next.

Given simplicial complexes at two consecutive time points t_i and t_{i+1}, we do not have a direct inclusion map $K_{t_i} \to K_{t_{i+1}}$ or $K_{t_i} \leftarrow K_{t_{i+1}}$, because there may be a number of simplices that are present in K_{t_i} but not in $K_{t_{i+1}}$, and vice versa. To employ the machinery of zigzag persistent homology, we require inclusion maps (either forward or backward) between consecutive spaces. To that end, we map through the union space $K_{t_i} \cup K_{t_{i+1}}$, with each of the simplicial complexes K_{t_i} and $K_{t_{i+1}}$ mapping by inclusion into $K_{t_i} \cup K_{t_{i+1}}$, as shown in Equation (2.3). Note that the union $K_{t_i} \cup K_{t_{i+1}}$ is obtained using the abstract simplicial complexes K_{t_i} and $K_{t_{i+1}}$ by identifying vertices that correspond to the same sensor. For a set of T snapshots at time points $t_1, t_2, \ldots, t_T$, we thus obtain the sequence of simplicial complexes:

$$\begin{tikzcd}
 & (K_{t_1} \cup K_{t_2}) & & (K_{t_2} \cup K_{t_3}) & & (K_{t_{T-1}} \cup K_{t_T}) & \\
K_{t_1} \arrow[ur] & & K_{t_2} \arrow[ul] \arrow[ur] & & \cdots \arrow[ul] \arrow[ur] & & K_{t_T} \arrow[ul]
\end{tikzcd} \tag{2.3}$$

and the associated zigzag persistence module:

$$\begin{tikzcd}
 & H_1(K_{t_1} \cup K_{t_2}) & & H_1(K_{t_2} \cup K_{t_3}) & & H_1(K_{t_{T-1}} \cup K_{t_T}) & \\
H_1(K_{t_1}) \arrow[ur] & & H_1(K_{t_2}) \arrow[ul] \arrow[ur] & & \cdots \arrow[ul] \arrow[ur] & & H_1(K_{t_T}) \arrow[ul]
\end{tikzcd}$$

See Figure 2.6 for Rips complexes of four time points in a dynamic network (top row), with the union complexes used for mapping through (bottom row). From the zigzag persistence module above, the Pers($\mathscr{K}$) = $\{[b_j, d_j]\}$ containing the lifetimes of the homological features can be computed. At this stage it is worth noting the distinction between homology classes and coverage holes, as well as the lack of a straightforward definition of what a 'time-varying coverage hole' is.

One characterization of a time-varying coverage hole is known as an 'evasion path', which means that there exists a spatiotemporal path which remains uncovered. This can be thought of as a path that an intruder could travel in order to avoid detection. Homological methods have been used [48] to give necessary conditions for such an evasion path to exist, using the same coordinate-free setting assumed here. More recently, it has been further shown [1] that the opposite implication does not hold. Specifically, the presence of interval $[b, d]$ in the zigzag persistence output, does not imply that there exists an evasion path over that same interval. Because of this, there is not a one-to-one relationship between birth-death intervals and evasion paths. Further, when a hole opens, travels around in space, and eventually closes, it is clear what is meant by 'time-varying coverage hole', but in some cases a single hole may split into two, or two holes may merge into one. Because of the ambiguity introduced about 'which hole' is obliterated or preserved during these processes, it is unclear what constitutes a single coverage hole over time. The tracking of lifetimes of homolog-

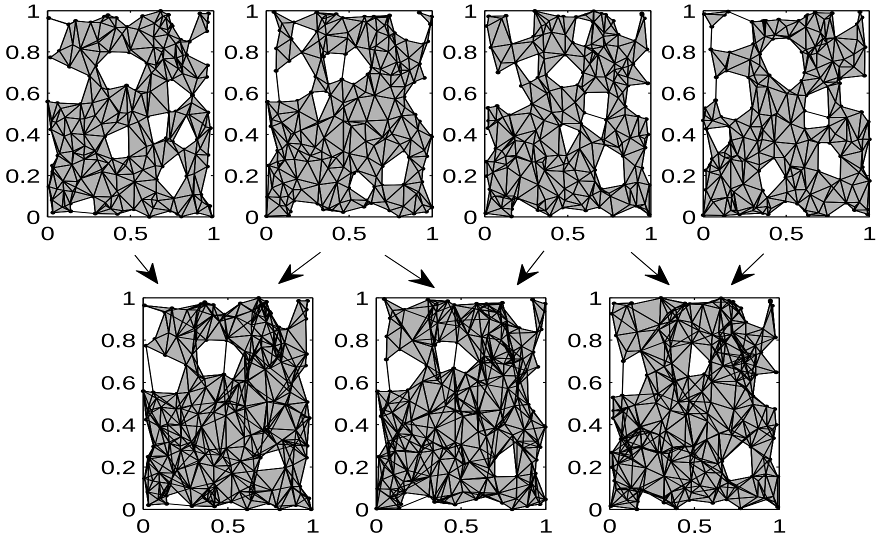

Figure 2.6 Sequence of Rips complexes and their union complexes.

ical features in zigzag persistence can, however, be done unambiguously, and although these are not interpreted as individual time-varying coverage holes, they are related to the coverage region in the following ways:

1. If a coverage hole appears at time b and remains isolated (does not split or merge with any other holes) until it disappears at time d, then the exact interval $[b,d]$ will be present in Pers($\mathscr{K}$). This means that in the case where a time-varying coverage hole is well defined, its lifetime is exactly represented in the barcode.

2. If an evasion path exists over interval $[b,d]$, then there exists an interval in Pers($\mathscr{K}$) containing $[b,d]$. This means that no evasion paths will be missed.

3. If Pers($\mathscr{K}$) $= \{[b_j,d_j] \mid j \in 1,\ldots,m\}$ are the intervals output from zigzag persistence, then define $\Lambda_i = \{j \in 1,\ldots,m \ \mid b_j \leq i \leq d_j\}$ to index the set of intervals which are 'alive' at time i.

Then

$$|\Lambda_i| = \beta_1(K_i)$$

(the number of intervals alive at any time point is equal to the number of holes in the simplicial complex at that time).

In light of this, we propose the use of the barcode/persistence diagram from zigzag persistent homology as a descriptor of the coverage of a network over time. In general, more bars and longer bars correspond to worse coverage. Since the computation only requires the Rips complexes (i.e. adjacency matrices of the communication graph) at each time point, this measure can be computed without requiring coordinates or distances between the sensors. In particular, summary statistics such as maximum and mean lifetimes of homological features can be computed, in addition to analysis of the barcode/persistence diagram as a whole. Metrics (such as the Wasserstein or bottleneck distances on persistence diagrams - see [12], [11]) have also been developed to compute pairwise distances between two persistence diagrams, which allows for quantification of differences between the coverage patterns of multiple time-varying networks.

At present, the only methods [37] available for analyzing coverage in dynamic sensor networks are to measure the coverage directly (using geometric information), and compute the proportion of uncovered area at each time point, or the proportion uncovered over a time interval (including a point as covered if it has been covered at any time during the interval).

In the following sections we describe how Pers($\mathscr{K}$) can be used effectively to quantify coverage in mobile sensor networks, and we illustrate its use in comparing mobility models. We further propose a method which is used in conjunction with the current zigzag persistence algorithm, and obtains, for each bar, specific representative cycles which are adaptively tracked over time, and can be used to obtain coarse size information about the holes present in the network.

2.3.3 Comparing mobility models

We present here some results on how the output from zigzag persistence can be used to characterize the coverage obtained by different mobility models for dynamic sensor networks. The analysis of coverage properties of mobility models previously used geometric descriptors to derive analytical results about the network, such as the limiting distribution of the nodes, the expected time-until-coverage for uncovered points, or expected proportion of uncovered area [37]. Ours is the first method which can additionally describe the dynamics of the coverage, in terms of the formation, duration, and behavior of coverage holes over time.

The two models we discuss are based on Brownian motion, and straight-line motion. For each of these, it is assumed that the nodes move independently from one another.

2.3.3.1 Mobility patterns: Discrete Brownian and Straight Line

Discrete Brownian: One model used to approximate the random movement of nodes in a large scale sensor network assumes the moves of each node to be independently and identically distributed (i.i.d.) according to a Brownian motion (eg. [43]). This is modeled in discrete-time by allowing each sensor to move according to a 2-dimensional Gaussian distribution at each time step (with variance proportional to the time increment).

Straight Line: A second commonly-used i.i.d. mobility model has each node choose an initial random direction and velocity, and then proceed (indefinitely) along this course ([36], [37]). In this setting, at $t = 0$ each node randomly chooses a direction $\theta \in [0, 2\pi)$ according to some distribution described by $f_\Theta(\theta)$, and randomly chooses a speed $v \in [v_{min}, v_{max}]$ according to a distribution described by $f_V(v)$. Typically $f_\Theta(\theta)$ and $f_V(v)$ are uniform distributions over their respective intervals, but other distributions are also possible. To compare the experimental results for the two cases, we choose the initial vector describing the velocity and direction for the Straight Line model from the same 2-dimensional Gaussian distribution used for each time step in the Discrete Brownian.

2.3.3.2 Simulations

Simulations were performed in a bounded region $[0,1]^2$, and for both mobility patterns the initial positions of the nodes were drawn from a uniform distribution over the region. When the movement of a sensor causes it to reach the boundary of the region, it bounces off with elastic (billiard-like) collisions, which will cause a change in the direction but not the speed.

Using $n = 100$ nodes, over an interval of $T = 50$ time points, 50 replications were generated for each mobility pattern. The simulations were paired, in that the initial coordinates of the sensors were the same for the two patterns, and were generated independently for each replication. All pairings for computing differences between the patterns, and computing the Wilcoxon signed rank were done by pairing the two replications (one from each mobility pattern) with the same initial configuration of sensors. The 2-dimensional Gaussian distribution used to initialize the movement in Straight Line pattern, and at each time point for the Discrete Brownian, had a mean zero and standard deviation equal to $0.1r$ (where $r = 0.977$ is the radius of the coverage disk for each sensor. This was chosen so that the communication graph would have an average degree of 15). A trace of one sensor following each of the mobility patterns for $T = 20$ (top row) and 1000 (bottom row) time points is shown in Figure 2.7, with the Discrete Brownian mobility pattern on the left, and the Straight Line mobility pattern on the right.

For each replication, the sequence of T simplicial complexes $K_1, \ldots, K_T$ (representing the sensor network at time points $1, \ldots, T$) are used along with the union complexes $K_i \cup K_{i+1}$ for $i = 1, \ldots, T-1$ to build the sequence $\mathscr{K}$, as in Equation (2.3). This is used as an input to compute the zigzag persistence birth-death intervals $\text{Pers}(\mathscr{K}) = \{[b_j, d_j]\}$ and associated representative cycles. A statistical analysis was performed to test for differences in the coverage properties of the two patterns, using both traditional coverage measures and descriptors obtained from our homological methods. The variables used for analysis are described in Section 2.3.3.3.

Due to the spatial distribution of the nodes being stationary in time (uniformly distributed on

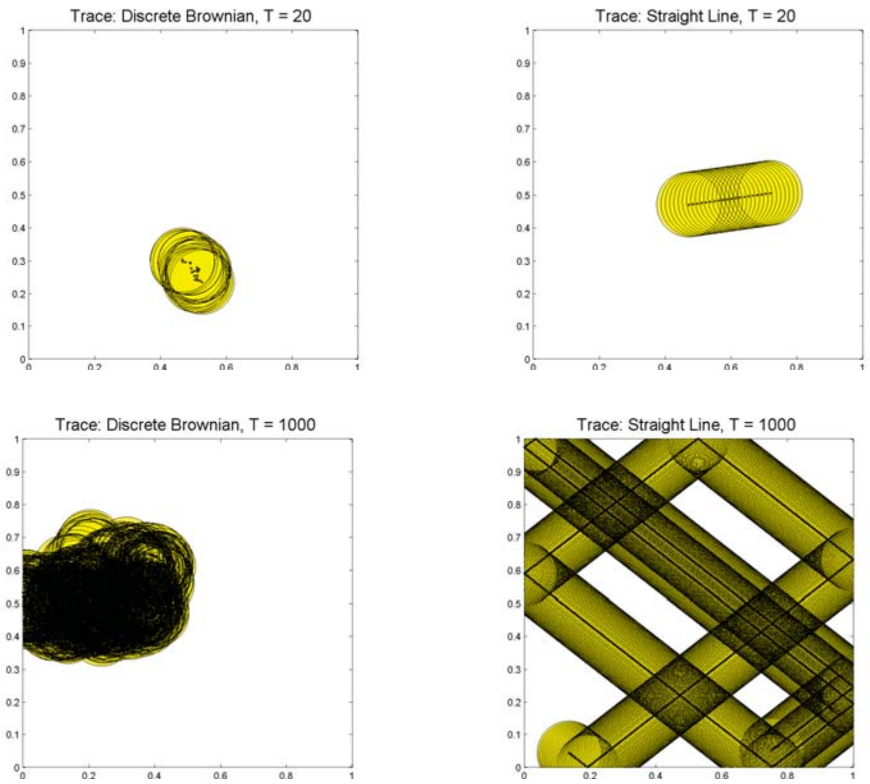

Figure 2.7 Discrete Brownian and Straight Line mobility patterns.

Table 2.1 List of the summary statistics extracted from the barcodes.

Variable	**Description**
barcode	An m-by-2 matrix containing the set of birth-death pairs for a given simulation run (the number m will vary, run to run). This is the main descriptor of the tracked homological features using zig-zag persistence.
LTcounts	A T-length vector with the counts of how many bars have length (lifetime) t, for $t = 1, \ldots, T$ in a given simulation run. i.e.) *LTcounts*(1) is the number of bars that persist for only a single timepoint, *LTcounts*(2) is the number of bars with a lifetime of 2,..., *LTcounts*(T) is the number of bars that persist over the entire simulation run.
# of bars	(scalar) The number m of birth-death intervals $\{[b_j, d_j] \mid j = 1, \ldots, m\}$ in a barcode for a given simulation run.
sum of bars	(scalar) The sum $\sum_{j=1}^{m}(d_j - b_j)$ of all bar lengths (lifetimes) in a given simulation run.
interval coverage	A T-length vector giving the proportion of the simulation region covered by time t, for $t = 1, \ldots, T$ in a given simulation run. A point in the simulation region is considered covered by time t, if it is covered at any point in the interval $[0, t]$.

$[0,1]^2$), all point-wise coverage statistics, such as the average proportion of uncovered area or average number of coverage holes at any time point, should be the same for the two mobility patterns [31]. What we expect might differ between the two groups, is the way in which coverage holes form, merge, split, and close, which can be detected in differences in the distribution of lifetimes of homology classes (the number and lengths of the intervals in Pers($\mathscr{K}$). i.e. bars in the barcode).

2.3.3.3 Coverage statistics

The output of zigzag persistent homology on the sequence of simplicial complexes gives a set of birth-death intervals Pers($\mathscr{K}$) $= \{[b_j, d_j] \mid j = 1, \ldots, m\}$ for each replication (each represented as a barcode). Statistical analysis is performed using 50 barcodes for the Discrete Browian simulation runs, and 50 barcodes for the Straight Line simulations. The summary statistics extracted from the barcodes are described below, with results of the statistical analysis on theses variables detailed in Section 2.3.3.4.

Example barcodes from one simulation run and for each mobility pattern are shown in Figure 2.8 (Discrete Brownian - left, Straight Line - right). The colors of the bars will be later used when identifying bars with specific representative cycles. Note that a quick look at a single pair of barcodes does not unveil a clear indication of whether there is a difference between the time-varying first homology of the two patterns, thus justifying a more careful statistical analysis. Since the mobility patterns are time-stationary, the variables involving the lifetimes of the homological features are of greatest interest, in contrast to those which depend on the specific birth or death time. The statistically insignificant differences in the barcodes led us to look at lifetimes of the homological features.

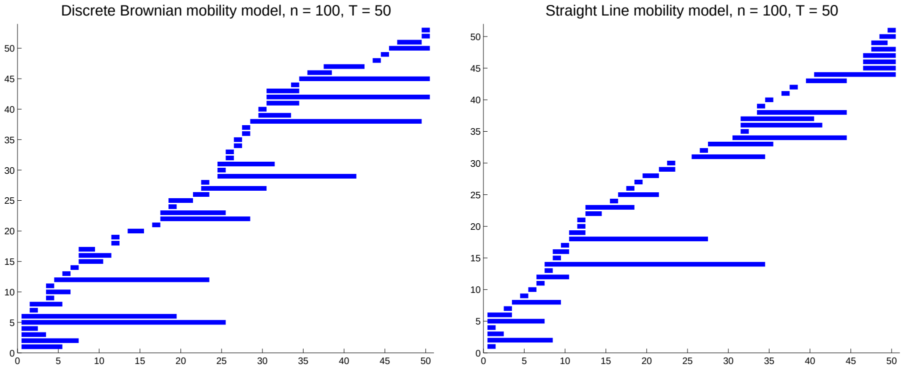

Figure 2.8 Barcodes for realizations of Discrete Brownian and Straight Line.

2.3.3.4 Results

The results given in this section are each followed by a set of brackets containing the mean and standard deviation (in parentheses) for each group, followed by the p-value obtained from the nonparametric Wilcoxon signed rank test. This test was used as a result of the invalid normality assumption necessary for a paired samples t-test.

Comparing the Discrete Brownian motion and Straight Line mobility patterns, there is no statistically significant difference between the two groups for *# of bars* [DB=42.42(10.52), SL=42.6(6.42), $p = 0.70$], or *sum of bars* [DB=201.96(48.19), SL=197.58(25.50), $p = 0.71$]. There is however a statistically significant difference in the variance of the two patterns for both *# of bars* ($p < 0.001$) and *sum of bars* ($p < 0.0001$), with the Discrete Brownian pattern having larger variability than the Straight Line. Figure 2.9 shows histograms for the distribution of *# of bars* (left) and *sum of bars* (right) for the Discrete Brownian (top row) and Straight Line (bottom row) mobility patterns.

The counts of lifetimes are distributed differently for the two groups. The Discrete Brownian mobility pattern has a significantly higher number of very short lifetimes (for $t = 1$, $p < 0.0001$), and very long lifetimes (for $t = 50$, $p < 0.001$). A few long lifetimes ($t = 19$ and 22) are also more frequent in the Discrete Brownian pattern, with moderate significance ($p < 0.05$). The Straight Line mobility pattern has a significantly higher number of short-medium length lifetimes ($t = 4, \ldots, 13$, all have $0.001 < p < 0.05$). Even after a Bonferroni correction for multiple hypothesis testing, the differences for $t = 1$ and 50 are still statistically significant (at the level $p < 0.001$). Histograms of *LTcounts* for the two mobility patterns are shown in Figure 2.10, with the lifetimes that show statistically significant differences highlighted. The Discrete Brownian (top left) and Straight Line (bottom left) mobility patterns, as well as the paired difference in *LTcounts* (right) are shown, with the lifetimes whose frequency has a statistically significant difference between the groups highlighted. The lifetimes that occur more frequently in the Discrete Brownian pattern ($t = 1, 19, 22$ and ,50) are highlighted in red in the top plot, and those that occur more frequently in the Straight Line pattern ($t = 4, \ldots, 13$) are highlighted in green in the bottom plot.

The variable *interval coverage* is a more traditional coverage measure, and we see that over time, the Straight Line mobility pattern will sweep out coverage of a greater proportion of the total area than the Discrete Brownian model. The proportion of area covered over time interval $[0, t]$ for each of the two mobility patterns, are shown in Figure 2.11, with all simulation runs overlaid as dotted lines. Mean interval coverage is shown as a thick solid line for each mobility pattern (Discrete

Brownian in red, Straight Line in green). The difference between the two mobility patterns in *interval coverage* is statistically significant for time points $t = 7, \ldots, T$ $(p < 0.001)$. This is in agreement with previous work [36], as well as the fact that a sensor traveling a path of fixed total length will cover the greatest area if it travels in a straight line. We additionally note that the time-point-wise coverage, measured by proportion of covered area, as expected shows no statistically significant difference between the patterns, see Figure 2.12. Again, all simulation runs are overlaid as dotted lines, with mean coverage shown as a thick, solid line for each mobility pattern (Discrete Brownian in red, Straight Line in green)

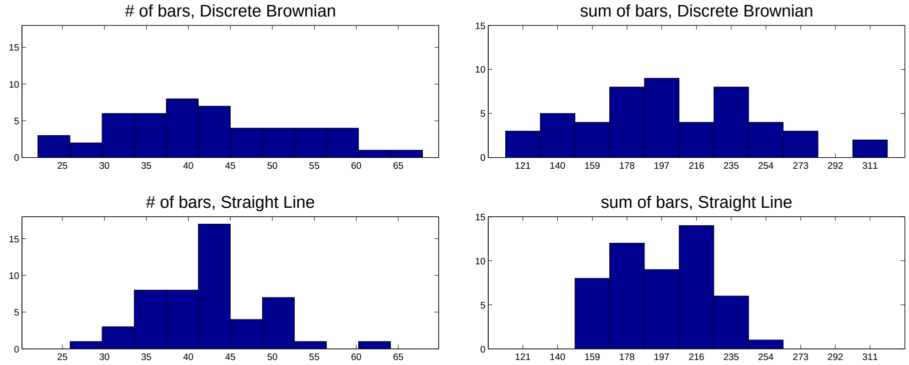

Figure 2.9 Histograms comparing number of bars in two mobility patterns.

2.3.3.5 Discussion

While it may appear by the above results that the two mobility patterns both have the same stationary distribution, and the same average energy expenditure at each time point, there is a difference in the time-varying coverage pattern displayed by the two models. For the Discrete Brownain mobility pattern, the erratic movement of the sensors results in many quickly appearing and disappearing small holes (usually present for only a single time-point). Additionally, this mobility

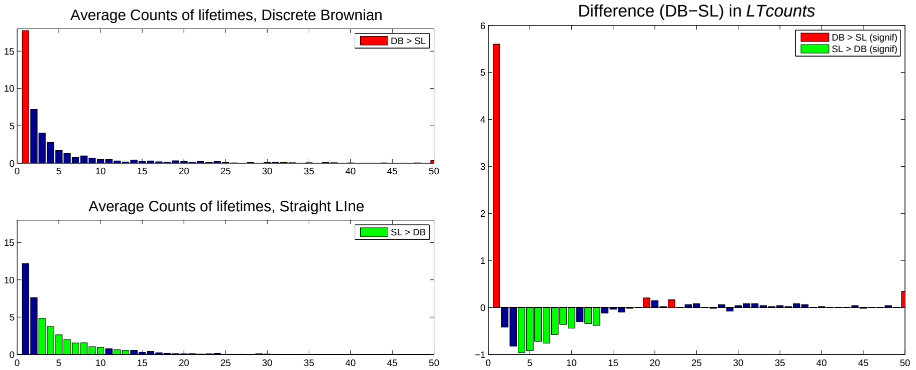

Figure 2.10 Comparison of counts of lifetime lengths for the two patterns.

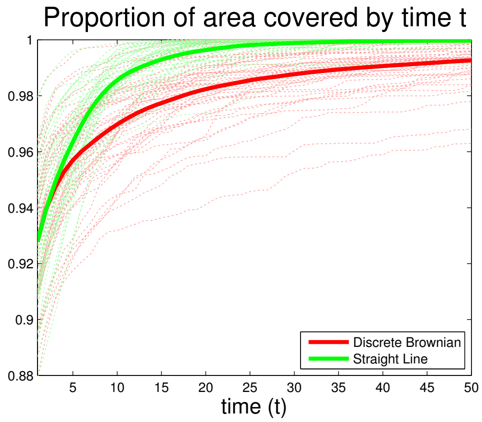

Figure 2.11 Mean interval coverage for the two patterns.

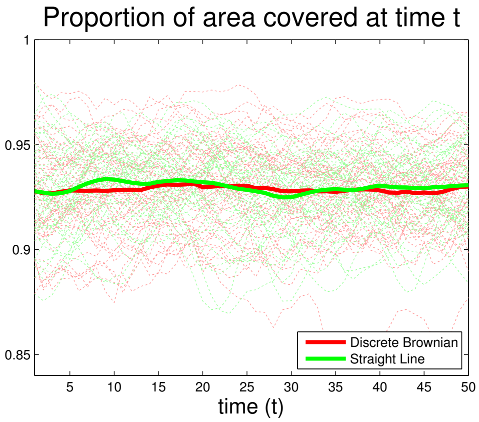

Figure 2.12 Proportion of area covered over time for the two patterns.

pattern displays significantly more long-lasting coverage holes, which typically correspond to large holes that are present in the initial configuration (i.e. the mobility pattern does not fill in existing holes quickly). For the Straight Line mobility pattern, since the sensors are each following a smooth trajectory, the coverage holes seem to appear, grow, shrink and disappear smoothly, instead of appearing and disappearing rapidly, or remaining uncovered for longer periods. In light of this, the Straight Line mobility model would be preferable in situations such as surveillance, or intruder detection, where it is important to quickly cover holes present in the initial deployment, and long-lasting coverage holes would prove costly. The Discrete Brownian model might be more desirable in circumstances where a thorough inspection takes precedence over time, such as in geographical surveying or environmental monitoring.

2.4 Coordinate-free estimation of hole size

The barcode obtained from zigzag persistence gives us a quantitative descriptor for the time-varying coverage of a network. Just as knowing the Betti number (number of holes) for a given simplicial complex tells us nothing about the hole sizes, the presence of a long bar in the barcode may or may not correspond to a large hole geometrically. Given that our network is described as a sequence of adjacency matrices (describing the simplicial complex at each snapshot, but without coordinate information), the best available estimate is the hop-length of the shortest cycle surrounding a hole. This can be obtained without having to compute the shortest cycle explicitly, by performing a hop-distance filtration on the simplicial complex (at each time point). For a simplicial complex K, the hop distance filtration is a nested sequence of simplicial complexes $K^1 \subseteq K^2 \subseteq \ldots$ defined as follows:

Definition: The *hop distance filtration* on a simplicial complex K, performed up to a maximum hop distance of m, is a nested sequence of simplicial complexes $K^1 \subseteq K^2 \subseteq \ldots K^m$, defined inductively:

1. K^1 is the original complex K
2. K^h contains all of the simplices of K^{h-1}, and adds edges between any nodes that were h hops apart in K, as well as all possible higher-dimensional simplices (i.e. if three edges forming a triangle are present in K^h, the associated 2-simplex will be added to K^h as well).

A hop distance filtration for a simplicial complex consisting of a single loop is shown in Figure 2.13. The original complex, K, consisting of a single loop is shown on the left, followed by the complexes K^2 (center) where there is still a non-trivial homology class, and K^3 (right) where the hole is completely filled in. Since the loop has a hop-length of 7 hops, it becomes ‘filled in’ by a triangle at a depth of 3 in the hop distance filtration (when edges are added between nodes that are three hops apart). Table 2.2 gives the relationship between the hop-length of the shortest cycle surrounding a hole, and its persistence in the hop distance filtration. For a given simplicial complex, each of its holes will have a corresponding ‘depth’ to which they persist in the hop distance filtration. The depths themselves can be taken as measures of the sizes of the holes, or alternatively the depths squared may be used (since the depth is a length-based measurement, its square will be proportional to area). This one measure may be used as an overall relative measure of persistence.

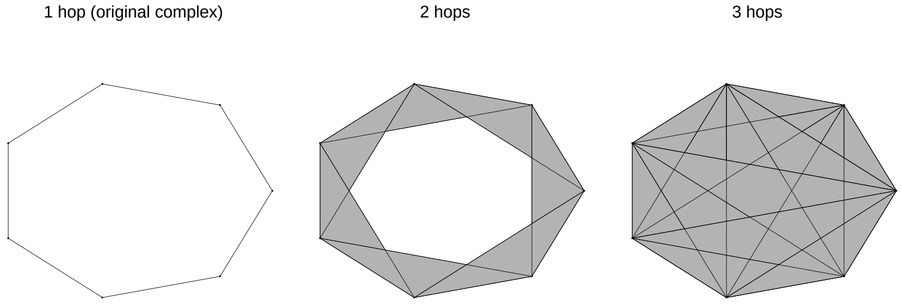

Figure 2.13 Illustration of hop distance filtration.

Hop-length of shortest cycle surrounding hole	Persistence of hole in hop distance filtration
4, 5, 6	1
7, 8, 9	2
10, 11, 12	3
$\vdots$	$\vdots$
$3k+1$, $3k+2$, $3k+3$	k

Table 2.2 Hole size (in hop-length) and corresponding depth in hop-distance filtration.

To illustrate the benefit of incorporating hop-distance size estimates, we compare various possible homological descriptors of timepoint-wise coverage with the true geometric coverage information. This was carried out for each time point in all of the simulation runs for the Discrete Brownian model described in the previous section (for $50 \times 50 = 2500$ points), the results of which are shown in scatterplots in Figure 2.14. The plots shown are: left - first betti number ($r = 0.176$); middle - sum of hole sizes (measured using depth in hop distance filtration, $r = 0.505$); right - sum of squared hole sizes (measured using depth in hop distance filtration, $r = 0.747$). For a given sensor network we measure the geometric coverage by the proportion of total area contained inside the coverage holes (ignoring uncovered area along the boundary of the simulation region, which is undetectable by the simplicial complex), and refer to this measure as *coverage hole area*. The homological coverage descriptors based on coordinate-free data only are:

1. The number of holes in the complex (first Betti number)
2. The sum of the hole sizes (measured by depth in the hop distance filtration)
3. The sum of the squared hole sizes (i.e. sum of squared depths)

As mentioned above, the number of holes in a simplicial complex (the first Betti number) does not describe the hole sizes at all. By combining information about the number of holes along with their estimated sizes, we are able to obtain a coordinate-free descriptor which correlates well with the true geometric information about the size of the coverage holes. This is rather surprising, since the coordinate-free information is very coarse relative to the geometric.

A note on the computational complexity of performing the hop distance filtration is in order, since the simplicial complexes K^h grow large quickly as h increases. The filtration does not generate any new first homology, and all holes are present in the original complex, so only the death times need to be computed. This may be accomplished by computing the first Betti number for each of the subsequent complexes, yielding the number of holes 'killed' at each depth. A topology-preserving simplicial collapse [54] is performed on each of the complexes before computing the

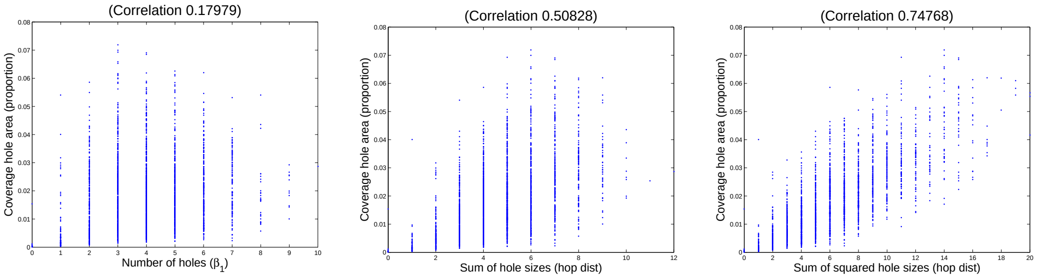

Figure 2.14 Coverage hole area vs homological features.

first homology, to increase efficiency of the computations. An alternative method to improve efficiency would be to compute persistent homology of the filtration using the Morse theoretic collapse algorithm presented in [40].

In addition to using the hop-distance filtration as a measure of the sizes of the holes present in the network at each time point, we would like to link the hole sizes present at time i, with the bars (obtained from zigzag persistence) at time i, for each time point i. The hop depth information can be combined with the zigzag persistence, to enhance a barcode with estimated size information for each bar at each time point. To that end, we need to make a choice for the homology class corresponding to each bar, as well as a specific representative cycle for that homology class. Observing when the inclusion of the cycle becomes trivial in the hop-distance filtration will tell us the size of the largest hole that cycle encircles. Unlike the set of birth-death intervals, the choice of homology class for each bar is not unique, so we would like our choice to be geometrically-motivated, and as close as possible to the 'canonical basis' described at the end of Section 2.2.1, thus having each homology class surrounding exactly one hole. The method we propose to achieve this is described in the following section.

To obtain the set of depths for a given complex, it is not necessary to use persistent homology to compute explicitly the depth of each hole in the hop distance filtration. Since all the holes are present at $h = 1$ (the original complex), simply computing the first Betti number for each of the subsequent complexes, until the first homology is trivial, will yield the number of holes that are 'killed' at each depth. Since the sizes of the complexes grows large quickly as h increases, this can be obtained efficiently by performing a topology-preserving simplicial collapse method [54] before computing the homology.

2.5 Tracking representative cycles

Given the set of intervals $\{[b_j, d_j]\}$ obtained from zigzag persistence, we want to have a choice of representative cycle for each interval, at each time point. The homology classes for this set of cycles should form a basis for the homology, and the choice of representative cycles over time should map into each other in a meaningful way. We propose a method, to be computed alongside the zigzag algorithm, which returns such representative cycles. The method is briefly described here, with a detailed mathematical and algorithmic description reserved for Chapter 3.

Intuitively this method aims to compute a 'canonical basis' (described at the end of Section 2.2.1), where there is one representative cycle surrounding each hole. Given the Rips complex for a static sensor network, without an embedding or geometric information, such a canonical basis is impossible to obtain. In the time-varying setting however, a small amount of 'canonical' information is available: when a coverage hole is first formed by the removal of a 2-simplex (triangle), the boundary of that triangle is known to surround exactly the hole of interest. The idea behind our method is then to use that boundary as the representative cycle for the homology class at its birth time, and propagate that information forward through the sequence of complexes as best as possible. The representative cycles we choose need to also be compatible with the interval decomposition in the zigzag algorithm, (the technical detail of this compatibility is described in [21]).

When applying this method alongside the zigzag algorithm, each bar in $\text{Pers}(\mathscr{K}) = \{[b_j, d_j]\}$ is associated with a specific representative cycle at each time point. This associates each bar with a specific hole (or set of holes) that it surrounds, even though this information is not directly available to us. We can obtain size estimates for the hole(s) by including the representative cycle in the hop-distance filtration of the complex (at each time point), as described in Section 2.4. If the representative cycles did indeed form a canonical basis, then the size information about each hole over time would be attached one-to-one with a corresponding bar. Although guarantees of a true canonical basis are impossible, when implemented in practice the method gives representative cycles that are geometrically quite meaningful. Short-lived holes are typically surrounded by a tight cycle at their boundary, and holes that begin with the removal of a triangle and then grow in size are also well-tracked.

2.5.1 Examples

Here, we present a number of examples where the representative cycles and associated size estimates give useful and interesting results, unavailable through other methods. Recall that all of the results and computations discussed in this section are obtained using only the communication graph of the network at each time point, with no information about coordinates or distances between neighboring sensors.

2.5.1.1 Tracking holes in a dense network

Figure 2.15 illustrates a network which is initially fully covered, and has a number of small coverage holes appearing over time, one of which is persistent. The barcode displaying lifetimes of homological features can be seen in the top left, with the bars color-coded to correspond to their associated representative cycles in the other figures. It can be seen that each representative cycle remains relatively tight around one coverage hole, and the set of cycles does correspond to a canonical basis at each time point. Overall, when a network is sufficiently dense that its coverage holes appear and disappear in an isolated fashion (as opposed to splitting and merging with other holes), this method performs very well.

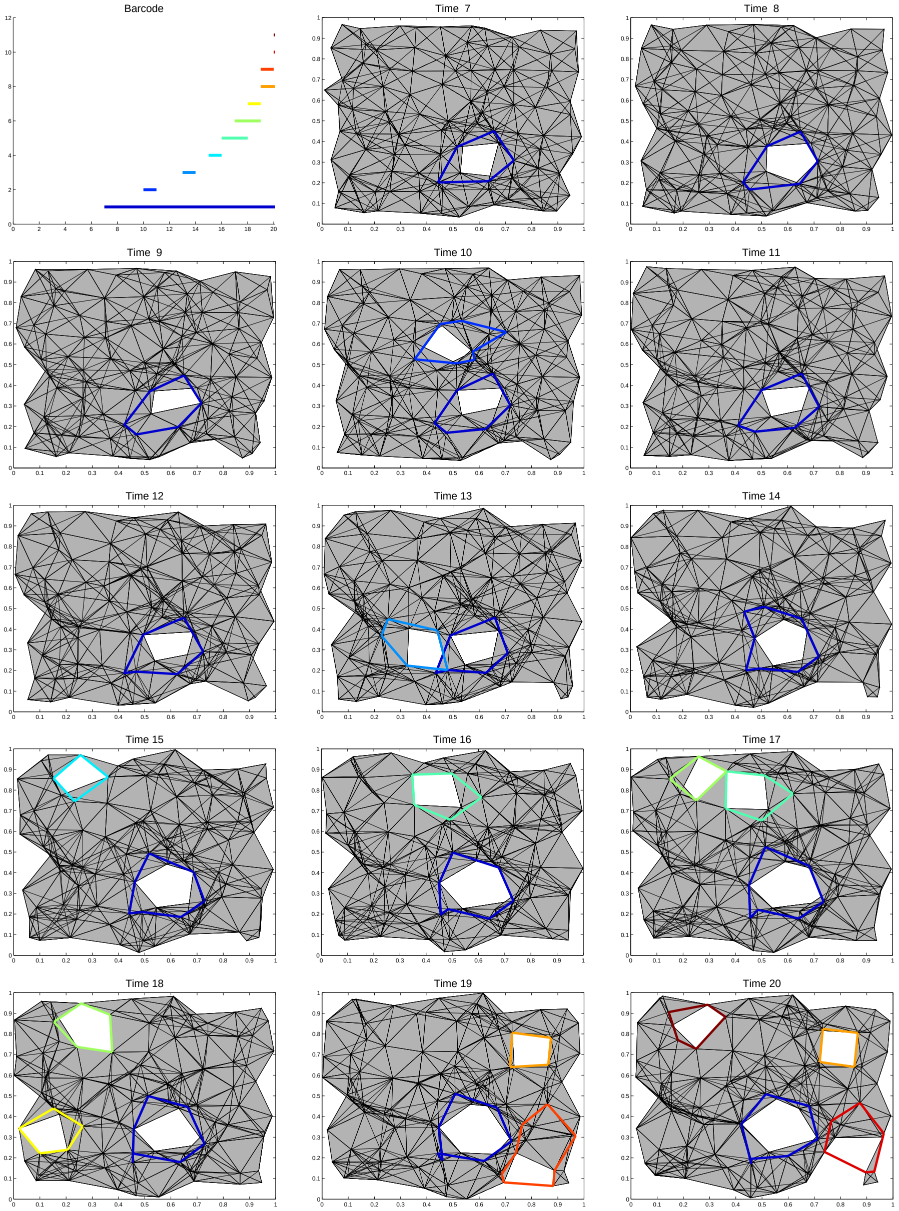

Figure 2.15 Tracking representative cycles in a dense network.

2.5.1.2 Detecting and evaluating severity of expanding failure region

In dense networks, the coverage holes are typically small and short-lived, so the representative cycles themselves provide fairly accurate tracking. Cases where the representative cycle itself does not 'tightly' surround a hole, its inclusion in the hop-distance filtration will still accurately reflect the size of the hole. This is especially useful for holes that are persistent over time, to better understand whether the hole is of increasing severity (perhaps due to a malicious attack or systematic failure). To compute dynamic size estimates for the hole(s) associated with each bar, the persistence in the hop-distance filtration for each representative cycle is attached to its corresponding bar at each time point. This is visualized in the barcode by thickening the bar by an amount proportional to the depth its representative cycle persists in the hop-distance filtration at that time. Figure 2.16 shows snapshots of a time-varying network with an expanding failure region, and the associated thickened barcode is shown in Figure 2.17 (with hop-distance computed up to a maximum depth of 3). It can be seen that the hole which is growing in time is easily observed in the barcode as a bar which thickens over time.

2.5.1.3 Maintaining perimeter around a guarded region

Representative cycles can also be used to determine whether an existing cycle remains unbroken over time. This can be of particular use when there is an area which needs to remain isolated, while guards roam about the region surrounding it. Without requiring precise locations of the guards, we can determine whether there remains an unbroken cycle surrounding the protected area, by tracking the persistence of the cycle that is initially present. Figure 2.18 shows a set of sensors/guards which initially surround a protected area tightly (top row), and then begin randomly moving about the environment. After some time, the guards still form a cycle (drawn in red, bottom left) which has been continuously enclosing the protected area, but eventually when the guards wander too far apart we detect the breaking of the cycle (bottom right).

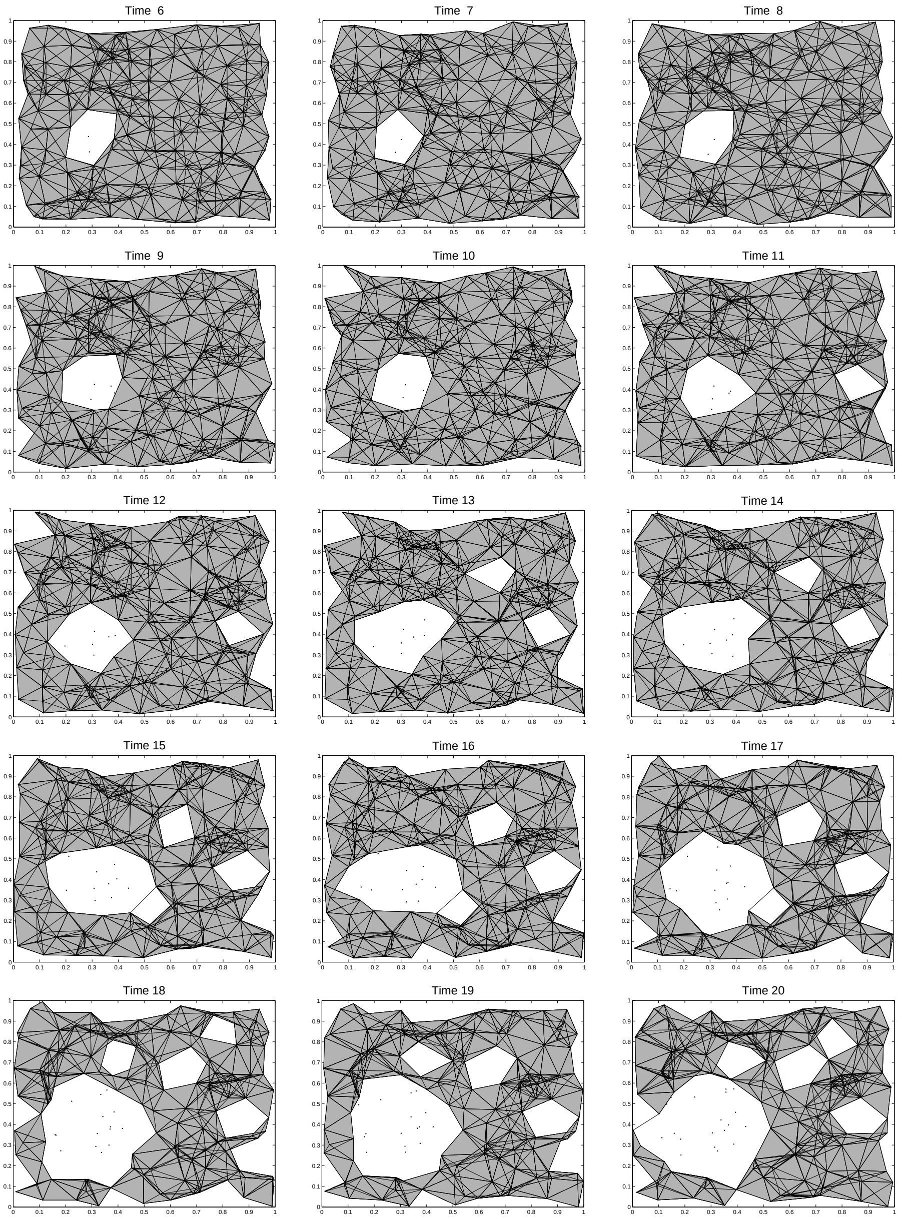

Figure 2.16 Network with an expanding failure region.

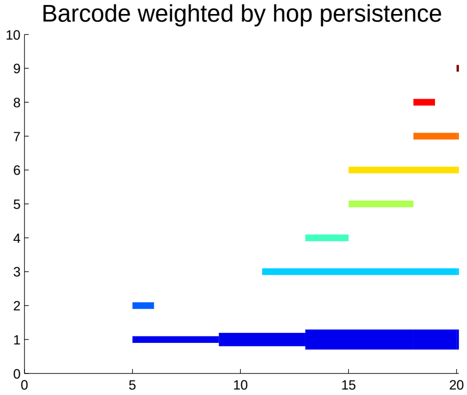

Figure 2.17 Weighted barcode for expanding failure region.

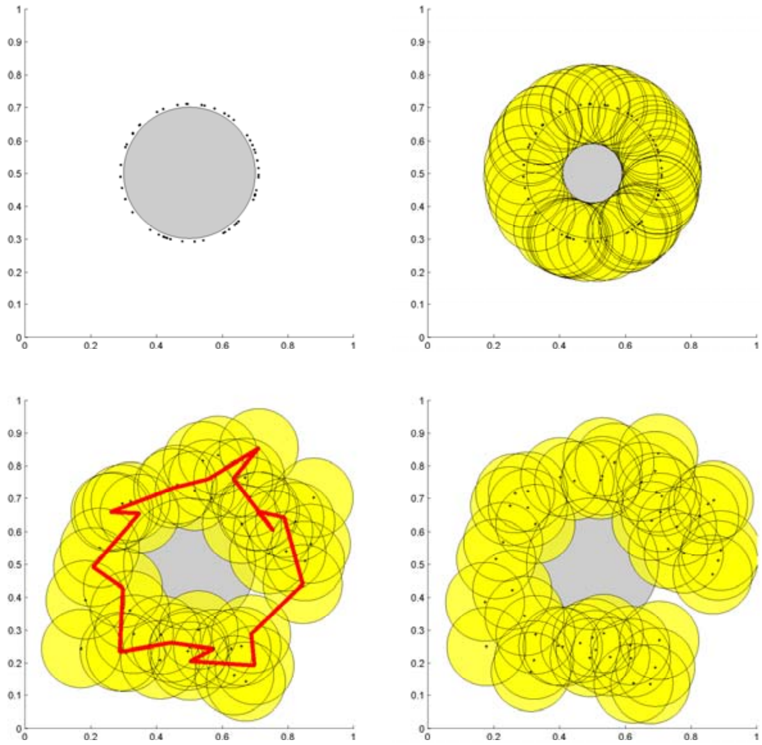

Figure 2.18 Tracking perimeter formed by mobile guards.

2.6 Conclusions and Future Work

We have presented strategies of exploiting computational topology to describe time-varying coverage in a dynamic sensor network, while using only local information about which nodes neighbor each other at each time step.

Zigzag persistent homology takes the sequence of simplicial complexes (representing the dynamic network), and outputs a barcode of birth and death times of homological features in the sequence. We described the relationship between these birth-death intervals and the time-varying coverage holes in the network, and demonstrated how the barcode output is a useful quantitative descriptor to detect coverage differences when comparing sensor network mobility patterns.

We developed a method to obtain a specific set of geometrically-meaningful representative cycles for each birth-death interval, at each time point. This set of representative cycles is then used to track coverage holes over time, as well as to obtain size estimates (in conjunction with a hop-distance filtration) for the holes at each time point. This size information is then incorporated into the barcode, for a more complete description of the dynamic coverage of the network. While this method was developed with applications to dynamic sensor networks in mind, the algorithm may be also adapted to obtain an adaptive choice of representative cycles in any dimension, for any persistent homology or zigzag persistent homology computation, thus providing an area for future research.

A surprising amount of information can be gleaned about the time-varying coverage of the network using homological methods; and, all of this is achieved in a setting where no coordinate or edge-length information is required, and only a binary adjacency matrix for the network at each time point is used.

CHAPTER

3

ADAPTIVE TRACKING OF REPRESENTATIVE CYCLES IN ZIGZAG PERSISTENT HOMOLOGY

For the article this chapter is based on, see [21] or arXiv:1411.5442.

3.1 Introduction

The field of topological data analysis [6] has been blossoming in recent years, and many more statisticians, computer scientists and engineers are beginning to use topological tools to study their data.

The most popular and successful of these tools is persistent homology, a method which characterizes a space (or object) by using a multi-scale description of its topological features. These include characteristics like the number of connected components, holes, or voids. A variant of regular persistent homology is zigzag persistent homology, which describes the topological features as they vary over a sequence of spaces.

In this chapter, we propose an algorithm for obtaining specific representative cycles to track homological features over a sequence of simplicial complexes in a geometrically meaningful way.

In Section 3.2, we build up the foundational terminology and required notations for our discussion. This includes brief descriptions of simplicial homology and zigzag persistence. In Section 3.3 we describe our method, and propose an algorithm for implementing it on a sequence of simplicial complexes. Section 3.4 proves the correctness of our algorithm, as it was applied in the examples in Sections 2.3.3 and 2.5.

3.2 Terminology and notation

3.2.1 Simplicial complexes and homology

We will use ideas from simplicial homology theory throughout, using the same basic definitions and notations introduced in Section 2.2.1 for simplicial complexes, homology groups and classes. For a general reference on algebraic topology (including simplicial homology), see [24].

When writing an equation about the homology of a space, we use a general notation of $\mathsf{H}(K)$, which can be taken to mean that the total homology $H_*(K)$, or a specific $H_p(K)$, could be inserted into the equation in the place of $\mathsf{H}(K)$. Similarly, we use $\beta(K) = \text{rank}(\mathsf{H}(K))$ as a general notation for the associated Betti number. For the later applications to sensor networks, and for visualization purposes, it is convenient to think of $\mathsf{H}(K)$ to mean $H_1(K)$.

3.2.2 Zigzag persistence

The theory of zigzag persistent homology is concerned with how the homology changes over a sequence of spaces. There are mathematical results [8] showing that the changing homology of such a sequence can be expressed uniquely in terms of birth and death times of homological features in the sequence. There is also an algorithm [7] for computing this birth-death decomposition for a given sequence of simplicial complexes. In this section we expand on the brief introduction to zigzag persistence given in Sections 2.3.1 and 2.3.2.

Consider a sequence of simplicial complexes $K_1, K_2, \dots, K_n$, connected by either forward inclusion maps $K_i \to K_{i+1}$ or backward inclusion maps $K_i \leftarrow K_{i+1}$. We write this sequence as

$$K_1 \longleftrightarrow K_2 \longleftrightarrow \dots \longleftrightarrow K_n.$$

The inclusion maps induce linear maps between the associated homology spaces $V_i = \mathsf{H}(K_i)$, which we write as the *zigzag persistence module*

$$\mathbb{V} = V_1 \overset{p_1}{\longleftrightarrow} V_2 \overset{p_2}{\longleftrightarrow} \dots \overset{p_{n-1}}{\longleftrightarrow} V_n \xrightarrow{p_n} V_{n+1}, \tag{3.1}$$

where $K_i \longrightarrow K_{i+1}$ induces the forward map $V_i \xrightarrow{f_i} V_{i+1}$, and $K_i \longleftarrow K_{i+1}$ induces the backward map $V_i \xleftarrow{g_i} V_{i+1}$. Regardless of the direction, we use $i(\cdot)$ to denote the inclusion map between consecutive simplicial complexes. We further assume that consecutive simplicial complexes K_i and K_{i+1} differ by exactly one simplex, so $K_{i+1} = K_i \cup \{\sigma\}$ (in the forward case), or $K_{i+1} = K_i - \{\sigma\}$ (in the backward case).

When σ is a d-simplex, its addition results in either an increase in the dimension of the d-dimensional homology space, or a decrease in the dimension of the $(d-1)$-dimensional homology space. Similarly, the removal of a d-simplex σ results in either an increase in the $(d-1)$-dimensional homology, or a decrease in the d-dimensional homology. When the dimension of the homology

Table 3.1 The affected homology space when d-simplex added/removed.

Case	$V_i \xleftrightarrow{p_i} V_{i+1}$
1. Birth by addition	$H_d(K_i) \xrightarrow{f_i} H_d(K_{i+1})$
2. Birth by removal	$H_{d-1}(K_i) \xleftarrow{g_i} H_{d-1}(K_{i+1})$
3. Death by addition	$H_d(K_i) \xrightarrow{f_i} H_d(K_{i+1})$
4. Death by removal	$H_{d-1}(K_i) \xleftarrow{g_i} H_{d-1}(K_{i+1})$

space increases, we refer to this as a *birth*, and when the dimension decreases, we refer to this as a *death*.

$$\begin{aligned} Birth: \dim(V_{i+1}) &= \dim(V_i)+1 \\ Death: \dim(V_{i+1}) &= \dim(V_i)-1 \end{aligned}$$

Each inclusion between simplicial complexes will induce maps between the homology spaces of all dimensions, but these maps will be simple identity maps in all dimensions except for one. This will depend on whether the addition or removal of d-simplex σ results in a birth or a death. For the addition or removal of a d-simplex σ, the map $V_i \xleftrightarrow{p_i} V_{i+1}$ on the corresponding homology zigzag module, will be interpreted as a forward or backward linear map between the appropriate-dimensional homology spaces, as summarized in Table 3.1.

A main result from the theory of zigzag persistence, is that a zigzag module such as in Equation (3.1) has an interval decomposition,

$$\mathbb{V} \cong \mathbb{I}(b_1, d_1) \otimes \mathbb{I}(b_2, d_2) \otimes \ldots \otimes \mathbb{I}(b_m, d_m), \tag{3.2}$$

which is unique up to isomorphism, and is equivalently expressed as the multiset of pairings of births and deaths in the sequence, and represented as integer intervals, called the *zigzag persistence* of $\mathbb{V}$ [8]

$$\text{Pers}(\mathbb{V}) = \{[b_j, d_j] \mid j = 1, \ldots, m\}. \tag{3.3}$$

These are interpreted as birth and death times of homological features in the sequence.

3.2.3 Right filtration

Given a zigzag module $\mathbb{V}$ as in Equation (3.1), the zigzag persistence algorithm [7] computes the interval decomposition in Equation (3.2) by keeping track of a right filtration $R(\mathbb{V})$ on the spaces. The right filtration $R(\mathbb{V})$ is computed incrementally, and results in a filtration (a nested sequence of subspaces) on V_n, along with a birth time associated to each quotient space, as detailed below. A right filtration on V_i is denoted

$$\mathscr{R}_i = (R_i^0, R_i^1, \ldots, R_i^i), \tag{3.4}$$

where $R_i^0 \leq R_i^1 \leq \ldots \leq R_i^i$ and $R_i^i = V_i$. The quotients R_i^1/R_i^0, R_i^2/R_i^1, ..., R_i^i/R_i^{i-1} are each associated with a birth time b_i^j (for $j = 0, \ldots, i$), which are recorded in the vector

$$\mathbf{b}_i = (b_i^1, b_i^2, \ldots, b_i^i). \tag{3.5}$$

We may write the quotients as

$$\mathscr{R}'_i = (R_i^1/R_i^0, R_i^2/R_i^1, \ldots, R_i^i/R_i^{i-1}).$$

The computation of a right filtration is defined inductively, depending on whether the map from V_i to V_{i+1} is a forward map $\xrightarrow{f_i}$ or a backward map $\xleftarrow{g_i}$. For a single vector space V_1, we have the base case of $i = 1$, and we define

$$\mathscr{R}_1 = (\mathbf{0}, V_1) \text{ and } \mathbf{b}_1 = (0).$$

In the inductive step, if we are given $\mathscr{R}_i$ and $\mathbf{b}_i$ as in Equations (3.4) and (3.5) above, then

- If $V_i \xrightarrow{f_i} V_{i+1}$, then

$$\begin{aligned} \mathscr{R}_{i+1} &= (f_i(R_i^0), f_i(R_i^1), \ldots, f_i(R_i^i), V_{i+1}), \\ \mathbf{b}_{i+1} &= (b_i^1, b_i^2, \ldots, b_i^i, i+1). \end{aligned} \tag{3.6}$$

- If $V_i \xleftarrow{g_i} V_{i+1}$, then

$$\begin{aligned} \mathscr{R}_{i+1} &= (\mathbf{0}, g_i^{-1}(R_i^0), g_i^{-1}(R_i^1), \ldots, g_i^{-1}(R_i^i)), \\ \mathbf{b}_{i+1} &= (i+1, b_i^1, b_i^2, \ldots, b_i^i). \end{aligned} \tag{3.7}$$

Since we assume that consecutive simplicial complexes differ by at most one simplex, the change in dimension between V_i and V_{i+1} is at most 1. Similarly, the dimension of the quotient space R_i/R_{i+1} is either 0 or 1, for $i = 1, \ldots, n$, with their total dimension equaling that of V_i. The dimension of V_i is the rank of the homology group for K_i (the Betti number, $\beta(K_i)$), which is at most i:

$$\dim(V_i) = \operatorname{rank}(\mathsf{H}(K_i)) = \beta(K_i) \leq i.$$

For example, the dimension of the quotient spaces will be a sequence of 0's and 1's

$$\dim(R_i^1/R_i^0, R_i^2/R_i^1, \ldots, R_i^i/R_i^{i-1}) = (0, 0, 1, 1, 0, \ldots, 1, 0).$$

Note that choosing one homology class from each of the nonzero quotient spaces results in a basis for V_i. The right filtration on V_i can then be described using the unique subspaces in the right filtration (which have corresponding quotient spaces of dimension 1). Indexing the nonzero quotient spaces by $j_1, \ldots, j_{\beta(K_i)}$, define $W_i^k = R_i^{j_k}$ for the spaces $R_i^{j_1}, \ldots, R_i^{j_{\beta(K_i)}}$ to obtain a more compact representation of the right filtration $\mathscr{R}$:

$$\begin{aligned}\mathscr{W}_i &= (W_i^1, \ldots, W_i^{\beta(K_i)}) \\ &= (R_i^{j_1}, \ldots, R_i^{j_{\beta(K_i)}}),\end{aligned} \tag{3.8}$$

where the $R_i^{j_k}$ are those with $\dim(R_i^{j_k}/R_i^{j_k-1}) = 1$, therefore the quotient spaces W_i^j/W_i^{j-1} are all one-dimensional. We say that a basis $\{[w_i^j]\}_{j=1}^{\beta(K_i)}$ for V_i is *compatible* with the right filtration $\mathscr{W}_i$ if there is one basis element in each quotient space:

$$[w_i^j] \in W_i^j/W_i^{j-1},$$

for $j = 1, \ldots, \beta(K_i)$. We return to this concept in Section 3.3.2.

Additionally, let

$$\mathbf{b}_i^W = (b_i^{j_1}, b_i^{j_2}, \ldots, b_i^{j_{\beta(K_i)}}),$$

contain the birth times of the non-zero quotient spaces, which is the birth vector for $\mathscr{W}$. So $\mathbf{b}_i^W$ is a subset of the birth vector $\mathbf{b}_i$ for the full right filtration $\mathscr{R}$.

The zigzag persistence algorithm is implemented by determining whether a birth or a death is occurring with each simplex addition or deletion. The right filtration and birth vector are then updated accordingly, and when a death occurs, the quotient space R_i^j/R_i^{j-1} corresponding to it is determined, and the associated birth time b_i^j used to output the interval $[b_i^j, i]$.

While the output of intervals $\{[b_j, d_j] \mid j = 1, \ldots, m\}$ is unique, there may be more than one way to choose homology classes corresponding to each interval. In Section 3.3 we will propose a method for choosing a homology class (by choosing a specific representative cycle for it) for each interval at each time point in a way that is geometrically motivated, all the while compatible with the right filtration.

3.3 Tracking representative cycles

3.3.1 Motivation

Our interest in choosing and tracking representative cycles over a sequence of spaces stems from analysis of coverage holes in time-varying sensor networks. The idea of using homological methods to study coverage in sensor networks was proposed by de Silva and Ghrist ([48], [47]), and the use of zigzag persistent homology allows some of these ideas to be employed in the dynamic network setting. The set of intervals output from the zigzag persistence algorithm describes the birth and death times of homological features, and these features do not necessarily correspond to individual coverage holes [1]. Ideally, we are interested in tracking coverage holes over time, but this is not possible in general, given the constraints on the limited geometric information available with the adopted sensor network model. Instead, we try and obtain a 'good' representative cycle for a hole as it appears in the network, and then propagate this cycle over time as best as possible. Below we describe in more detail the model for the sensor network (3.3.1.1), the representative cycles we would ideally like to obtain (Section 3.3.1.3), and those that we are able to compute (Section 3.3.1.4).

3.3.1.1 Homology for sensor networks

A network consists of a set of sensors, each at the center of an isotropic coverage disk of radius r. The union of the disks yields the coverage region for the entire network, and we are interested in making statements about coverage properties of this network, as the sensors are allowed to move over time. A communication graph is constructed by connecting any two sensors by an edge when they are less than a distance $2r$ from one another, and the homology of the Rips complex of this graph is used to approximate the homology of the coverage region of the network. Figure 3.1 shows the coverage region (left), communication graph (center) and associated Rips complex (right) for a given sensor network. Note that a Rips complex is the maximal simplicial complex that can be built from a given graph, but since we are only interested in computing the first homology we only need to consider the 2-skeleton of the Rips complex. The Rips complex includes a 2-simplex defined by three sensors whenever their coverage disks have nonempty pairwise intersections, so if the disks have no triplet-wise intersection then a small hole may be present in the coverage region which is not detected by the Rips complex. See Figure 3.2 (left) for such an example. For our purposes we designate such holes as too small to be of importance, and work with the homology as it is defined by the Rips complex. We refer to [48] for an alternative approach, which allows false alarms (holes in the complex which do not exist in the coverage region), but is able to give coverage guarantees.

A final key result that we mention is by Chambers *et al.* [9], who show that the first homology of the Rips complex (a combinatorial object) is the same as the first homology of the projection of the Rips complex onto the plane (this projection is referred to as the Rips shadow). The Rips shadow corresponding to the sensor network from Figure 3.1 is shown in Figure 3.2 (right). For a Rips complex K, we denote its shadow by K^S.

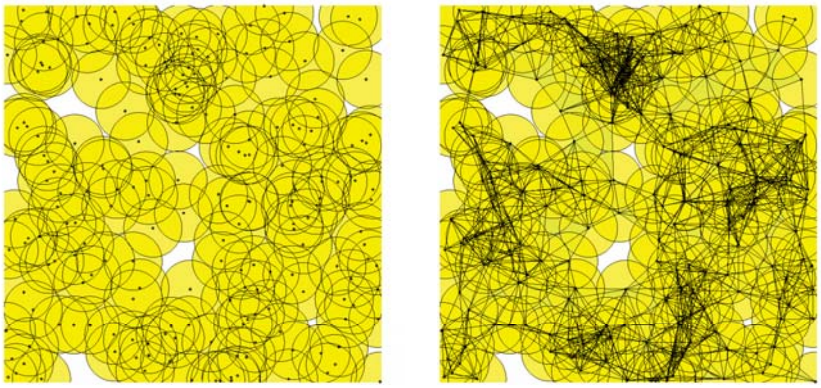

Figure 3.1 The coverage region and communication graph for a sensor network.

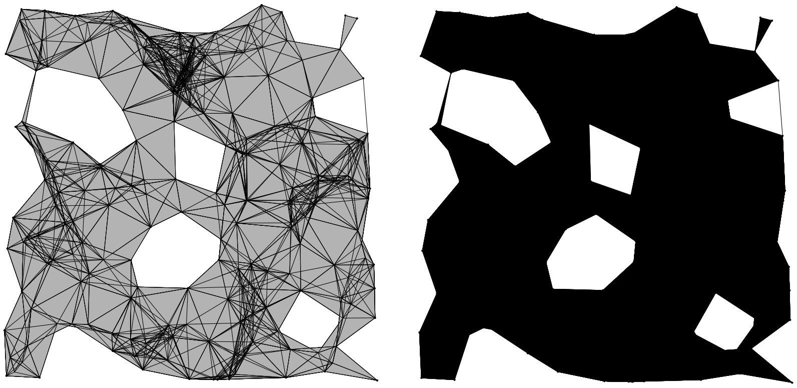

Figure 3.2 The Rips complex and Rips shadow of the communication graph.

3.3.1.2 Zigzag persistence for dynamic networks

In a time-varying network (represented by a sequence of simplicial complexes $K_{t_1}, \ldots, K_{t_T}$), homology classes may be tracked over time using zigzag persistent homology by mapping through the union complexes. So the sequence of simplicial complexes in Equation (2.3) gives rise to an associated zigzag persistence module.

For implementational and theoretical purposes the sequence in Equation (2.3) is broken down, with each forward map re-written as a series of single simplex additions, and each backward map as a series of single simplex deletions. This refinement induces the analogous refinement on the zigzag module.

3.3.1.3 Canonical basis

Given a compact region in the plane such as the Rips shadow K^S, there exists a 'canonical basis' for its first homology space, where each basis homology class surrounds a single hole. Consider $\overline{K^S}$, the complement of K^S in $\mathbb{R}^2$, then the number of separate components in $\overline{K^S}$ (ignoring the infinite component) is equal to the number of holes in K^S (i.e. the rank of $H_1(K^S)$). This result is a specific case of the more general principle of Alexander Duality (see, for example Ch. 5 of [39]), which for a certain class of spaces, relates the k-th reduced homology of a space to the $n-k-1$-th cohomology of the complement of the space (where n is the embedding dimension). We do not go into details here, but the salient point is that a canonical basis exists for the first homology of a space in the plane, with one homology class surrounding each hole.

Since the Rips complex K and its Rips shadow K^S have the same homology, a desirable goal would be to have a homology basis for the Rips complex, where projection of this basis onto the Rips shadow gives the canonical basis. In particular, we would like a representative cycle for each homology class in the basis, where the projection of the representative cycle onto the Rips shadow is homologous to the boundary of one of the holes. *In general, this desirable goal is not possible.* The Rips complex itself is not embeddable in two dimensions, so Alexander Duality cannot be applied to obtain a canonical basis for its first homology. Moreover, although K has the same homology as K^S, it is impossible to know whether a given homology basis for K corresponds to the canonical basis or not (without knowing coordinates for the vertices, or the projection map from K onto K^S).

We will see in Section 3.3.1.4 that taking the dynamic nature of the network into account, there are some cases where it is possible to make a canonical choice for a homology class (with corresponding representative cycle) at its birth or death time. In Section 3.3.2 we present a method for obtaining these cycles, and for updating them as the network evolves over time, along with an explicit algorithm for doing so.

3.3.1.4 Partial canonical information

As described in Table 3.1, a homology class can be born by either the addition or removal of a simplex, and similarly a death is caused by either the addition or removal of a simplex, resulting in four distinct cases for how the homology can change. In this section we illustrate the two cases corresponding to 'births', and how one of them allows a canonical choice of homology class. Our discussions here are with respect to the first homology, but the same principles hold for d-dimensional homology.

In the sequence of simplicial complexes, the birth of a homology class occurs at time i when either the forward map $V_i \xrightarrow{f_i} V_{i+1}$ has nonzero cokernel, or the backward map $V_i \xleftarrow{g_i} V_{i+1}$ has nonzero kernel. Of these two cases, $\ker(g_i) \neq \mathbf{0}$ is the only one which indicates the specific homology class that is being born.

Consider the case where the birth is in first homology. If a hole is formed by the removal of a 2-simplex, then there is a unique homology class (the one surrounding the hole) which is born. This homology class also the unique homology class in $\ker(g_i)$ (i.e.: the only homology class that is nontrivial in K_{i+1} but trivial in K_i). On the other hand, if a hole is formed by the addition of an edge, there are many choices for which homology class is being born, with no choice being canonical. For example, if a hole is split into two by the addition of an edge, then which of them is the 'new' hole? See the first two rows of Figure 3.3 for an illustration of these cases.

Our approach then, is to maintain a basis for the homology at each time point, making the canonical choice of homology class whenever available, and tracking that choice through the sequence of complexes as best as possible. Our method for implementing this, along with the specific basis we maintain and its relation to the zigzag algorithm, is detailed in Section 3.3.2.

3.3.2 Algorithm

As mentioned in Section 3.4, a zigzag module $\mathbb{V}$ (Equation (3.1)) has unique interval decomposition $\text{Pers}(\mathbb{V}) = \{[b_j, d_j] \mid j = 1, \ldots, m\}$, which describes the birth and death times of homological features in the sequence. This decomposition is determined through the maintenance of a right filtration $\mathscr{W}_i$ (Equation (3.8)) on the space V_i, and a birth vector $\mathbf{b}_i$ for $i = 1, \ldots, n$. The zigzag persistence algorithm performs this task by determining whether a birth or a death is occurring for each simplex σ being added or removed, and updating the right filtration and birth vector accordingly (and outputting the appropriate birth-death interval whenever a death occurs).

At each stage in our algorithm, we maintain a basis for the homology that attempts to approx-

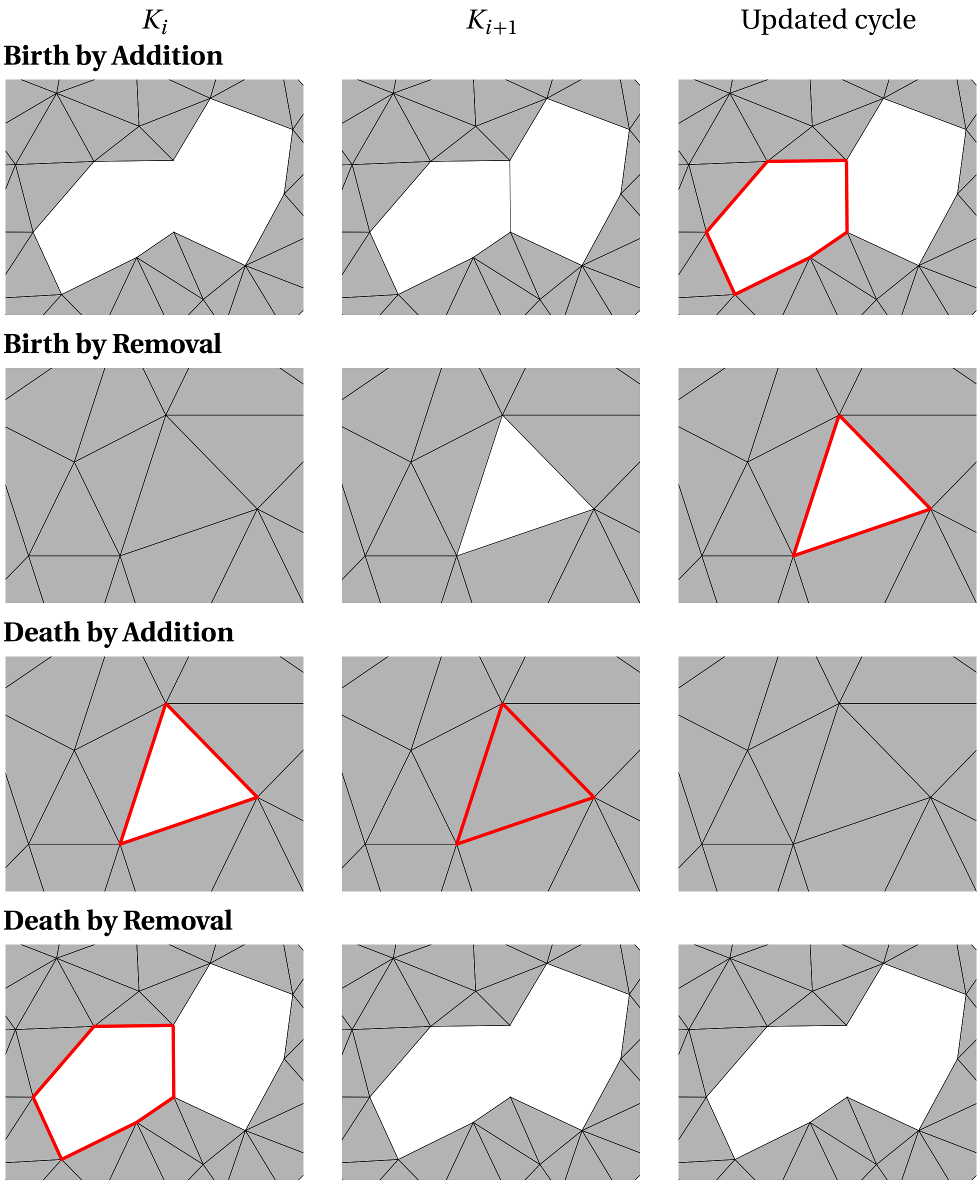

Figure 3.3 The four first homology changes, and corresponding representative cycles.

imate the canonical basis as best as possible. Further, the basis homology classes are compatible with the right filtration $\mathscr{W}_i$, in the sense that the j^{th} basis homology class is an element of the j^{th} quotient space W_i^j/W_i^{j-1}. This means that the span of the first j homology classes in the basis is equal to the j^{th} subspace W_i^j in the right filtration $\mathscr{W}_i$. This property is necessary if we wish to interpret our basis homology classes as corresponding to particular intervals in the birth-death decomposition. The intervals are really describing specific quotient spaces in the right filtration that have persisted over the sequence, so our homology classes need be assigned one-to-one to the quotient spaces. The proof that this property is maintained during the algorithm is presented in Section 3.3.2.

The method we present here is computed using the regular zigzag persistent homology algorithm, but keeps an explicit record of the homology basis chosen for the right filtration at each time point. The homology basis is stored by choosing a specific representative cycle for each basis homology class. The choice of basis homology classes is not unique, so it is made in a geometrically meaningful way, attempting to approximate the canonical basis. The zigzag persistence algorithm supplies information about whether the addition or removal of a simplex σ is causing a birth or a death. If it is a birth, σ is called a positive simplex, denoted σ^+, and if it is a death, σ is called a negative simplex, denoted σ^-.

When a birth occurs, we must add a new representative cycle to our list. As mentioned in Section 3.3.1.4, when a birth occurs due to the removal of a simplex σ, there is a canonical choice available for the new homology class. We choose the boundary $\partial\sigma$ of the removed simplex as the representative cycle for this homology class, since it is the shortest cycle surrounding the new hole. When the birth occurs due to the addition of a simplex σ, there is no canonical choice for which is the 'new' homology class, but any cycle containing σ will have its homology class in coker(f_i). For practical reasons, we choose the shortest cycle containing σ as the new representative cycle.

When a death occurs, we must remove a representative cycle from our list. Analogous to the two ways in which a birth can occur, a death occurs when either the forward map $V_i \xrightarrow{f_i} V_{i+1}$

has nonzero kernel, or the backward map $V_i \xleftarrow{g_i} V_{i+1}$ has nonzero cokernel. Of these two cases, $\ker(f_i) \neq \mathbf{0}$ is the only one which indicates the specific homology class $[c] = \ker(f_i)$ that is being killed (becoming trivial). In this case, the death occurs due to the addition of a simplex, and we reduce the matrix storing the representative cycles with respect to the boundary matrix ∂, and remove the cycle which becomes trivial. If the death occurs on account of the removal of a simplex σ, then the first representative cycle containing σ is removed, and a change of basis is performed to remove σ from any remaining representative cycles. This is done in the same way as the change of basis operation in the regular zigzag persistence algorithm.

We store the representative cycles for time i in the matrix W_i, which is retained for all time points. The algorithm is summarized below.

Algorithm for choosing and updating representative cycles

```
%Notation
$w_i^l =$ column $l$ of $W_i$
$w_i^j[\sigma] =$ coefficient of $\sigma$ in $w_i^j$
$\partial_d =$ boundary matrix

%Initialize
$W_0 = n \times 0$ matrix
$b_0 =$ empty vector

%Perform updates
for $i = 1$ to $n$
    if $K_i = K_{i-1} - \{\sigma\}$                %simplex removal
        if $\sigma^+$                              %birth
            $W_i = [\partial\sigma_i \; W_{i-1}]$          %prepend $\partial\sigma$
            $b_i = [b_{i-1} \; i]$
        else if $\sigma^-$                         %death
            $l =$ index of first nonzero entry in $r_\sigma$
            $c_l = r_\sigma(l)$, the coefficient of $\sigma$ in $w_l$
            for $j = 1$ to (# columns of $W_{i-1}$)            %change of basis
                $c_j = r_\sigma(j)$, the coefficient of $\sigma$ in $w_j$
                $w_j = w_j - \frac{c_j}{c} w_l$
            end
            $W_i = W_{i-1}$ with column $l$ and row $r_\sigma$ removed
            $b_i = b_{i-1}$ with entry $l$ removed
        end
    end
    if $K_{i+1} = K_i \cup \sigma$                 %simplex addition
        if $\sigma^+$                              %birth
            $W_i = [W_{i-1} \; (Cu - \sigma)]$       %append $Cu - \sigma$
            $b_i = [b_{i-1} \; i]$
        else if $\sigma^-$                         %death
            $l =$ index for col of $W_{i-1}$ trivial when $[\partial_d \; W_{i-1}]$ reduced
            $W_i = W_{i+1}$ with column $w_l$ removed
        end
    end
end
```

3.4 Correctness

Consider a sequence of simplicial complexes

$$K_1 \longleftrightarrow K_2 \longleftrightarrow \ldots \longleftrightarrow K_n.$$

connected by forward and backward inclusion maps, assuming without loss of generality that consecutive complexes differ by exactly one simplex. Each space V_i in the zigzag persistence module (Equation (3.1)) of this sequence has right filtration $\mathscr{W}_i$ (Equation (3.8)), with the j^{th} space in $\mathscr{W}_i$ denoted by W_i^j.

The adaptive representative cycles obtained using the algorithm described in Section 3.3.2 are stored as column vectors w_i^k in a matrix $\mathbf{W}_i$

$$\mathbf{W}_i = [w_i^1 \; w_i^2 \; \ldots \; w_i^{\beta(K_i)}].$$

Proposition 3.4.1. *The homology classes represented by the cycles w_i^k form a basis for V_i, and moveover, their order in $\mathbf{W}_i$ corresponds to the order of the right filtration $\mathscr{W}_i$ (Equation (3.8)). In other words, the span of the homology classes of the first j representative cycles is equal to the j^{th} space W_i^k in the filtration $\mathscr{W}_i$ of V_i.*

i.e.:

$$\text{span}\{[w_i^k]\}_{k=1}^{j} = W_i^j, \tag{3.9}$$

for $i = 1, \ldots, n$ and $j = 1, \ldots, \beta(K_i)$.

In the remainder of this section, we prove Proposition 3.4.1 by induction on i.

We begin with the base case of a single vector space $\mathbb{V} = V_1$, which results from a simplicial complex of one vertex $K_1 = \sigma$. This yields

$$\begin{aligned}\mathscr{W}_1 &= (V_1),\\ \mathbf{W}_1 &= [w_1^1],\end{aligned}$$

where $w_1^1 = [1]$ is the column vector of length 1 representing the cycle consisting of the vertex σ. The homology class $[w_1^1]$ spans the one-dimensional homology space $W_1^1 = V_1$.

In the inductive step, we assume that for

$$\mathscr{W}_i = (W_i^1, \ldots, W_i^{\beta(K_i)}), \tag{3.10}$$

$$\mathbf{W}_i = [w_i^1 \; w_i^2 \; \ldots \; w_i^{\beta(K_i)}], \tag{3.11}$$

we have (for $j = 1, \ldots, \beta(K_i)$)

$$\text{span}\{[w_i^k]\}_{k=1}^{j} = W_i^j,$$

We will show then that (for $j = 1, \ldots, \beta(K_{i+1})$)

$$\text{span}\{[w_{i+1}^k]\}_{k=1}^{j} = W_{i+1}^j, \tag{3.12}$$

for all four of the cases described in the algorithm (Section 3.3.2). In all cases we use σ to denote the d-simplex being added or removed, and the updates are performed on the representative cycles and right filtration of appropriate dimension (see Table 3.1).

1. Birth by addition. The map $V_i \xrightarrow{f_i} V_{i+1}$ has coker$(f_i) \neq \mathbf{0}$, and the new right filtration is

$$\mathscr{W}_{i+1} = \Big(f_i(W_i^0), f_i(W_i^1), \ldots, f_i(W_i^{\beta(K_i)}), V_{i+1}\Big),$$

where $V_{i+1}/f_i(W_i^{\beta(K_i)}) = \text{coker}(f_i)$.

The new list of representative cycles is

$$\mathbf{W}_{i+1} = [\mathbf{W}_i^{\sigma^+} \; w_{new}],$$

where $\mathbf{W}_i^{\sigma^+}$ is the matrix $\mathbf{W}_i$ with an additional row of zeros added, corresponding to simplex σ (so the cycles are now written in terms of simplices of K_{i+1} instead of simplices of K_i), and w_{new} is a cycle in K_{i+1} containing σ. There is no canonical choice for which cycle containing σ should be chosen, and our proof holds regardless of the choice. As mentioned in Section 3.3.2, we make this choice based on shortest hop length.

Since $w_{i+1}^k = w_i^k$ as chains (with the appropriate row for σ added containing a 0 coefficient), we get $[w_{i+1}^k] = f_i([w_i^k])$, for $k = 1, \ldots, \beta(K_i)$ because f_i is the map induced by inclusion. Therefore

$$\begin{aligned} W_{i+1}^j &= f_i(W_i^j) \\ &= f_i\left(\text{span}\{[w_i^k]\}_{k=1}^j\right) \\ &= \text{span}\{f_i([w_i^k])\}_{k=1}^j \\ &= \text{span}\{[w_{i+1}^k]\}_{k=1}^j, \end{aligned}$$

for $j = 1, \ldots, \beta(K_i)$.

Finally, we must show that $[w_{new}]$ is nontrivial and is in coker(f_i), and therefore linearly independent from $\{[w_i^j]\}_{j=1}^{\beta(K_i)}$, so they together span the $\beta(K_1) + 1 = \beta(K_{i+1})$-dimensional vector space $V_{i+1} = W_{i+1}^{\beta(K_{i+1})}$. First, note that having a nonzero coefficient for σ in w_{new}: that $[w_{new}] \neq 0$; and that any cycle c in the same homology class $[w_{new}]$ will also have a nonzero coefficient for σ. These are due to the fact that σ is not contained in the boundary of any other simplex, and the difference between homologous cycles must be written as a linear combination of boundaries (therefore the coefficient for σ is zero in the difference $c - w_{new}$, but is nonzero in w_{new}, so must also be nonzero in c). Moreover, note that $[w_{new}] \nsubseteq \operatorname{im}(f_i)$, since any homology class in $\operatorname{im}(f_i)$ must have a representative cycle in the image under inclusion $i(K_i) \subset K_{i+1}$, and all cycles in $[w_{new}]$ contain $\sigma \notin i(K_i)$. Therefore, we have

$$\begin{aligned} W_{i+1}^{\beta(K_{i+1})} &= V_i \\ &= \operatorname{im}(f_i) \oplus \operatorname{coker}(f_i) \\ &= \left(\operatorname{span}\{[w_{i+1}^k]\}_{k=1}^{\beta(K_i)}\right) \oplus [w_{new}] \\ &= \operatorname{span}\{[w_{i+1}^k]\}_{k=1}^{\beta(K_{i+1})}, \end{aligned}$$

as desired.

2. Birth by removal. The map $V_i \xleftarrow{g_i} V_{i+1}$ has $\ker(g_i) \neq \mathbf{0}$, and the new right filtration is

$$\mathscr{W}_{i+1} = \left(\ker(g_i), g_i^{-1}(W_i^1), g_i^{-1}(W_i^2), \ldots, g_i^{-1}(W_i^{\beta(K_i)})\right).$$

This is because in the full right filtration

$$\mathscr{R}_{i+1} = \left(\mathbf{0}, g_i^{-1}(R_i^0), g_i^{-1}(R_i^1), \ldots, g_i^{-1}(R_i^i)\right),$$

if R_i^j / R_i^{j-1} were nontrivial in $\mathscr{R}_i$, then $g_i^{-1}(R_i^j)/g_i^{-1}(R_i^{j-1})$ will be nontrivial in $\mathscr{R}_{i+1}$ for $j = 1, \ldots, i$. This means that if W_i^j is a subspace in $\mathscr{W}_i$ then $g_i^{-1}(W_i^j)$ is a subspace in $\mathscr{W}_{i+1}$. Also, the new term

$$g_i^{-1}(R_i^0)/\mathbf{0} = g_i^{-1}(\mathbf{0})/\mathbf{0} = \ker(g_i)/\mathbf{0} = \ker(g_i),$$

is nontrivial, and yields the first term $\ker(g_i)$ in $\mathscr{W}_{i+1}$.

The new list of representative cycles is

$$\mathbf{W}_{i+1} = [\partial \sigma \ \mathbf{W}_i],$$

where $\partial\sigma$ are the simplices that make up the boundary of σ, but considered in K_{i+1}, instead of K_i.

First we note that $\text{span}\{[\partial\sigma]\} = \ker(g_i)$. This is because under the backward inclusion map inclusion map $K_{i+1} \xleftarrow{i} K_i$, the image $i(\partial\sigma)$ is the boundary of simplex σ in K_i and hence homologous to zero, thus

$$g_i([\partial\sigma]_{i+1}) = [\partial\sigma]_i = \mathbf{0},$$

which means $[\partial\sigma] \subseteq \ker(g_i)$. The cycle $\partial\sigma$ is also nontrivial in K_{i+1}, because if there exists a d-chain c in K_{i+1} that had $\partial\sigma$ as its boundary, then in K_i the union of σ with $i(c)$ in K_i would form a d-cycle, and the removal of σ would result in the death of that d-cycle, instead of the birth of a $(d-1)$-cycle, which is a contradiction. Therefore, $[\partial\sigma]$ spans a one-dimensional subspace of the one-dimensional space $\ker(g_i)$, so $\text{span}\{[\partial\sigma]\} = \ker(g_i)$.

Now we show that $W^j_{i+1} = \text{span}\{[w^k_{i+1}]\}^j_{k=1}$ for $j = 1,\ldots,\beta(K_{i+1})$. First note the index change, so

$$w^{k+1}_{i+1} = w^k_i,$$

for $k = 1,\ldots,\beta(K_i)$. Consider the representative cycle w^k_i, and another cycle c which is homologous to w^k_i in K_i. Since c and w^k_i are both $(d-1)$-cycles, they are also present in K_{i+1}. Then $[c]_i = [w^k_i]_i$ implies $[c]_{i+1} = [w^k_i]_{i+1} + a[\partial\sigma]_{i+1}$, where $a = 0$ or 1. Therefore

$$g_i^{-1}([w^k_i]) = [w^{k+1}_{i+1}] \oplus [\partial\sigma].$$

So

$$\begin{aligned} W^j_{i+1} &= g_i^{-1}(W_i^{j-1}) \\ &= \text{span}\{g_i^{-1}[w^k_i]\}^{j-1}_{k=1} \\ &= \text{span}\left\{[w^{k+1}_{i+1}] \oplus [\partial\sigma]\right\}^{j-1}_{k=1}, \end{aligned}$$

for $j = 2, \ldots, \beta(K_{i+1})$. Combining this with

$$W_{i+1}^1 = \ker(g_i) = [\partial\sigma] = [w_{i+1}^1],$$

we obtain

$$W_{i+1}^j = \operatorname{span}\{[w_{i+1}^k]\}_{k=1}^j,$$

for $j = 1 \ldots, \beta(K_{i+1})$, as desired.

3. Death by addition. For the map $f_i : V_i \to V_{i+1}$ we get $\ker(f_i) = [\partial\sigma]$ with a similar proof to that of case **2** above.

Since $\ker(f_i) \neq \mathbf{0}$, we have $\operatorname{coker}(f_i) = \mathbf{0}$, so $V_{i+1}/f_i(V_i) = \mathbf{0}$. Also, there exists an index $l \in \{1, \ldots, \beta(K_i)\}$ such that $[\partial\sigma] \in W_i^l$, but $[\partial\sigma] \notin W_i^{l-1}$ (using the convention $W_i^0 = \mathbf{0}$), so

$$f_i(W_i^l / W_i^{l-1}) = \mathbf{0}.$$

This gives

$$\mathscr{W}_{i+1} = \left(f_i(W_i^1), \ldots, f_i(W_i^{l-1}), f_i(W_i^{l+1}), \ldots, f_i(W_i^{\beta(K_i)})\right),$$

so we have

$$W_{i+1}^j = \begin{cases} f_i(W_i^j) & \text{if } j < l; \\ f_i(W_i^{j+1}) & \text{if } j \geq l. \end{cases} \tag{3.13}$$

Considering now the representative cycles, we need to determine the index l. Since the elements $\{[w_i^k]\}_{k=1}^{\beta(K_i)}$ form a basis for V_i, we can write uniquely

$$[\partial\sigma] = \sum_{k=1}^{\beta(K_i)} \alpha_k [w_i^k]. \tag{3.14}$$

Then $[\partial\sigma] \in \operatorname{span}\{[w_i^k]\}_{k=1}^{l} = W_i^l$, but $[\partial\sigma] \notin \operatorname{span}\{[w_i^k]\}_{k=1}^{l-1} = W_i^{l-1}$ implies that α_l is the last nonzero coefficient in this sum. We now define

$$w_{remove} = w_i^l,$$

and obtain

$$\mathbf{W}_{i+1} = [w_i^1 \; \dots \; w_i^{l-1} \; w_i^{l+1} \; \dots \; w_i^{\beta(K_i)}],$$

noting that all of the simplices in the $(d-1)$-cycles w_i^k are present in K_{i+1}. Therefore the corresponding homology classes are related by

$$[w_{i+1}^j] = \begin{cases} f_i([w_i^j]) & \text{if } j < l; \\ f_i([w_i^{j+1}]) & \text{if } j \geq l, \end{cases}$$

for $j = 1, \dots, \beta(K_{i+1})$, since f_i is the map induced by inclusion. This, together with Equation (3.13) yields

$$W_{i+1}^j = \operatorname{span}\{[w_{i+1}^k]\}_{k=1}^{j},$$

for $j = 1, \dots, \beta(K_{i+1})$, as desired.

Note that the index l indicating the last nonzero coefficient in Equation (3.14) also determines the birth-death interval: $[\mathbf{b}_i^W[l], i]$.

4. Death by removal. The map $V_i \xleftarrow{g_i} V_{i+1}$ has $\operatorname{coker}(g_i) \neq \mathbf{0}$. There exists an index l such that $W_i^j \subseteq \operatorname{im}(g_i)$, for all $j < l$, but $W_i^l \nsubseteq \operatorname{im}(g_i)$. Then

$$g_i^{-1}(W_i^l / W_i^{l-1}) = \mathbf{0},$$

so

$$\mathscr{W}_{i+1} = (g_i^{-1}(W_i^1), \dots, g_i^{-1}(W_i^{l-1}), g_i^{-1}(W_i^{l+1}), \dots, g_i^{-1}(W_i^{\beta(K_i)})).$$

We note that the image of this in V_i is

$$\begin{aligned} g_i(\mathscr{W}_{i+1}) &= \mathscr{W}_i/_{\mathrm{coker}(g_i)} \\ &= (W_i^1, \dots, W_i^{l-1}, W_i^{l+1}/_{\mathrm{coker}(g_i)}, \dots, W_i^{\beta(K_i)}/_{\mathrm{coker}(g_i)}). \end{aligned} \tag{3.15}$$

Considering now the representative cycles, l is the index of the first representative cycle w_i^l which contains σ. To see that this is the same index l as described above, note that since w_i^k doesn't contain σ for $k < l$, we have $[w_i^k] \in \mathrm{im}(g_i)$, and $\mathrm{span}\{[w_i^k]\}_{k=1}^j = W_i^j \subseteq \mathrm{im}(g_i)$, for all $j < l$, but $\mathrm{span}\{[w_i^k]\}_{k=1}^l = W_i^l \nsubseteq \mathrm{im}(g_i)$.

Denoting the coefficient for σ in representative cycle w_i^k by $w_i^k[\sigma]$, we consider another set of representative cycles in K_i

$$\hat{w}_i^k = w_i^k - \frac{w_i^k[\sigma]}{w_i^l[\sigma]} w_i^l.$$

By definition, σ is not present in any $\hat{w}_i^k$, so we are able to define

$$w_{i+1}^k = \begin{cases} \hat{w}_i^k & \text{if } k < l; \\ \hat{w}_i^{k+1} & \text{if } k \geq l, \end{cases}$$

to be our representative cycles in K_{i+1}, with the row corresponding to σ removed. Then

$$\begin{aligned} \mathbf{W}_{i+1} &= [\hat{w}_i^1 \dots \hat{w}_i^{l-1}\ \hat{w}_i^{l+1} \dots \hat{w}_i^{\beta(K_i)}], \\ &= [w_{i+1}^1 \dots w_{i+1}^{l-1}\ w_{i+1}^l \dots w_{i+1}^{\beta(K_{i+1})}]. \end{aligned}$$

We proceed by showing that the $\hat{w}_i^k$ completely determine the quotiented filtration $\mathscr{W}_i/_{\mathrm{coker}(g_i)}$ in Equation (3.15), in the sense that

$$W_i^j/_{\mathrm{coker}(g_i)} = \mathrm{span}\{[\hat{w}_i^k]\}_{k=1}^j, \tag{3.16}$$

for $j = 1, \dots, \beta(K_{i+1})$.

To show that Equation (3.16) holds, we show it separately for $j < l$, $j = l$, and $j > l$. For the first case, note that when w_i^k does not contain σ, we have $\hat{w}_i^k = w_i^k$. In particular, for $k < l$ we have $\hat{w}_i^k = w_i^k$, therefore

$$W_i^j /_{\text{coker}(g_i)} = W_i^j = \text{span}\{[w_i^k]\}_{k=1}^j = \text{span}\{[\hat{w}_i^k]\}_{k=1}^j,$$

for $j = 1, \dots, l-1$.

By assumption $W_i^l \nsubseteq \text{im}(g_i)$, but $W_i^{l-1} \subseteq \text{im}(g_i)$, so

$$W_i^l /_{\text{coker}(g_i)} = W_i^{l-1} = \text{span}\{[\hat{w}_i^k]\}_{k=1}^{l-1} = \text{span}\{[\hat{w}_i^k]\}_{k=1}^{l},$$

since $\hat{w}_i^l = \vec{0}$.

For $j > l$, we first note that the homology elements $\{[\hat{w}_i^k]\}$ are linearly independent for $k \in \{1, \dots, l-1, l+1, \dots, \beta(K_i)\}$. This is because each $[\hat{w}_i^k]$ is a subset of $[w_i^k] \oplus [w_i^l]$ (but not equal to $[w_i^l]$), and the $\{[w_i^k]\}$ are linearly independent. So $\{[\hat{w}_i^k]\}_{k=1}^j$ span a $(j-1)$-dimensional space when $j > l$ (since $[w_i^l]$ is trivial). Also, because all the $\hat{w}_i^k$ have a zero coefficient for σ, they are not in the coker(g_i). So

$$\text{span}\{[\hat{w}_i^k]\}_{k=1}^j \subseteq W_i^j /_{\text{coker}(g_i)}.$$

Moreover, we note that $W_i^j /_{\text{coker}(g_i)}$ is also a $(j-1)$-dimensional space for $j > l$. So $\text{span}\{[\hat{w}_i^k]\}_{k=1}^j = W_i^j /_{\text{coker}(g_i)}$.

It now follows that since

$$g_i([w_{i+1}^k]) = \begin{cases} [\hat{w}_i^k] & \text{if } k < l; \\ [\hat{w}_i^{k+1}] & \text{if } k \geq l, \end{cases}$$

then

$$W_{i+1}^{j} = \begin{cases} g_i^{-1}(W_i^{j}) = g_i^{-1}(W_i^{j}/_{\mathrm{coker}(g_i)}) & \text{if } j < l; \\ g_i^{-1}(W_i^{j+1}) = g_i^{-1}(W_i^{j+1}/_{\mathrm{coker}(g_i)}) & \text{if } j \geq l \end{cases}$$
$$= \mathrm{span}\{[w_{i+1}^{j}]\}_{i+1}^{j},$$

completing the induction.

3.5 Conclusion

Persistent homology and zigzag persistent homology represent the dynamic homology of a sequence of spaces by computing a set of intervals, describing the birth and death times of homological features in the sequence. In this chapter we presented a method for assigning a representative cycle at each time point to each interval in this decomposition. The original choice and method for updating these representative cycles are geometrically motivated, so they are interpreted as 'tracking' homological features. To be compatible with the birth-death decomposition obtained from zigzag persistent homology, there must exist an ordering on the representative cycles such that they form a basis for each space in the right filtration, and we proved that our representative cycles do, in fact, satisfy this property.

Some applications of the method to track coverage holes in time-varying sensor networks were presented in Section 2.5. For spaces in the plane, this method of tracking attempts to approximate the canonical basis for the first homology (where one homology class surrounds each hole), as best as possible, while still being compatible with the birth-death decomposition. Having chosen a specific representative cycle for each interval at each time point, additional features (such as estimates of hole size) can be attached onto the barcode, for a more comprehensive description of the dynamic coverage of the network.

CHAPTER

4

NODE DOMINANCE: REVEALING CORE-PERIPHERY STRUCTURE IN SOCIAL NETWORKS

4.1 Introduction

In this chapter, we present the local property of *node dominance* as a method for network analysis. We will show why node dominance is such a useful criterion, by defining a node-dominance-based algorithm for the core-periphery decomposition of a network, as well as by deriving its relation to the network community structure.

For our settings, the node dominance criterion may be checked by simply considering the neighbor set of two nodes. A node v is dominated by node w if all nodes that share and edge with v, also share an edge with w. The formal definition of node dominance is based on a simplicial complex (as opposed to graph) structure, and will be discussed in detail later. If we iteratively collapse dominated nodes, the resulting set (the network *core*) is shown to consist of nodes that are important with respect to the network flow, community structure, and global network structure. One especially important property of the core is the preservation of shortest paths, so a shortest path between any two nodes in the core is also a shortest path between them in the original network. The network *periphery* (the complement to the core, consisting of dominated nodes) is seen to consist of many disconnected components, including all the nodes in the network through which no shortest paths pass. These peripheral components also play a key role in the community structure of the network.

The intuitive notion that a network naturally decomposes into a core and periphery has appeared many times in the social network literature over the decades. Researchers have proposed different interpretations about what such a decomposition should look like, but it is commonly suggested that a 'core' should be central to the network (with respect to information flow, or shortest paths) [26], have high average degree [13], and be relatively well-connected both internally, and to the periphery [5] [61]. In contrast, the periphery should be connected to the core, but extremely sparsely connected amongst itself.

Borgatti and Everett [5] were the first to attempt to analytically describe these intuitive properties. They proposed an 'idealized core-periphery', wherein every core node is connected to every other core node, each peripheral node is connected to the core, and no peripheral nodes are connected to each other. They would then learn the core-periphery structure for a given network by assigning each node as 'core' or 'periphery' in the way that best correlated with this idealized structure. This method assumes explicitly that the probability of two nodes being joined by an edge is only a function of their 'core-ness', as opposed to some other characteristics, such as community membership. In this sense, the traditional core-periphery network model is in contrast to more common network models based on community structure. Both core-periphery and community network structures can be expressed using a stochastic blockmodel approach [61], but with different parameters, so under these models a given network will not display both structures simultaneously.

Another approach, by Rombach *et al.* [46] presents a more flexible generalization of Borgatti and Everett's philosophy, where a *core score* is computed for each node, using a range of possible core sizes and continuous/discrete transitions between core and periphery. Here, they admit that both core-periphery and community structure are often present in real-world networks, but still propose the core-periphery decomposition as an alternative/complementary analysis to the more common community detection methods. In Della Rossa *et al.* [14], an approach to periphery detection based on random walks is taken, where is it assumed that due to the extremely sparse connectivity of the periphery, a random walk will exit the set of peripheral nodes very quickly. Thus, a core-periphery profile for the network, along with a coreness value for each node, is computed using a greedy algorithm that adds nodes to the periphery one-by-one in a way that minimizes the expected time until a random walk exits the set. Again, this method focuses very heavily on the sparsity of the periphery, and is somewhat unrelated to any community structure that may be present in the network. For a good review of existing methods of core-periphery network decomposition, see the survey by Csermely *et al.* [13], or the introductory sections in [46].

Traditionally, approaches to community detection in networks have assumed that communities form a partition of the network, with each node belonging to exactly one community. A foundational method has been the Girvan-Newman algorithm [42], where communities are detected though iterative removal of nodes with high centrality. They defined the notion of 'modularity' as a stopping criterion for their algorithm, and many subsequent algorithms attempt to partition a network in such a way that optimizes (usually approximately) modularity [41], or cut ratio (approximated using spectral clustering) [10]. Fortunato provides an excellent overview of the breadth and depth of approaches to the community detection problem in his 100 page survey paper [18]. In more recent years, researchers are determining that partition-based methods are often somewhat unrealistic, since real-world networks with ground-truth communities typically display overlapping community structure [57], where one node may have multiple community memberships. See Xie *et al.* [55] for a survey of methods for overlapping community detection, including clique percolation, link clustering, and fuzzy detection methods using mixed-membership stochastic block models, or nonnegative matrix factorization.

A particularly realistic model for overlapping community detection is Yang and Leskovec's community affiliation graph model (AGM) [56] [59]. This model considers communities as 'overlapping tiles', and its distinguishing feature is that regions of community overlaps are *more* densely connected than regions involving single communities. Precisely, the probability of an edge existing between two vertices is based on the communities they share, with higher probability when they have more community memberships in common. This assumption is validated on data sets with ground-truth community memberships available, where higher edge densities are observed in community intersections [56]. AGM, and the other methods for overlapping community detection are more realistic than the partition-based methods, but they do not scale up well to networks larger than a few thousand nodes. A recent relaxation of AGM, referred to as Cluster Affiliation Model for Big Networks (BIGCLAM) [58], allows nodes to have continuous-valued community memberships, indicating their degree of involvement in a given community. This reduces the combinatorial optimization in AGM to a continuous optimization that can be solved using nonnegative matrix factorization. Therefore BIGCLAM is scalable to networks with hundreds of thousands, or millions of nodes. We will return to this model in Section 4.4.3.

In the current chapter, we will see how a core-periphery structure and a community structure are both present in real-world networks, and how node dominance informs us about both. The relationship between the core-periphery and community structure of a network has been touched upon previously by Leskovec *et al.* [34], where they also noted the presence of a network periphery, defined in terms of *whiskers* (clusters of nodes that are separable from the main network by removing a single edge), which were interpreted as small communities, weakly connected to the remaining network "core". In the AGM model mentioned above [59], Yang and Leskovec refer to the overlapping portions of communities as the "core" of the network. We will see that this interpretation does in fact concur with our notion of core and periphery, where in networks with ground-truth communities available, the nodes in the core obtained using node dominance typically have multiple community memberships, while the nodes in the periphery have fewer community memberships (often just one).

Iterative node dominance collapses were originally proposed independently by Wilkerson *et al.* [54] and Barmak and Minian [4], as a homology/homotopy-preserving simplification of a simplicial complex, with the distributed version described in [53]. Here, we explore much more deeply the use of this simplification as a network core, and describe the relationship between the core-periphery decomposition, and the community structure, global structure, and network flow properties.

In Section 4.2, we will first describe the relevant information for the simplicial complex representation of a network, and the background and definition of the node dominance criterion. We follow this in Section 4.3 by statements and derivations of the resulting properties of our core-periphery decomposition, and propose an algorithm for the use of peripheral components in community detection. In Section 4.4, we illustrate our method with two real-world network data sets which contain ground-truth community information. We not only observationally verify the importance of core nodes with respect to network flow and global structure, but see that our proposed use of the peripheral components for community detection performs very well (better than BIGCLAM, considered the state-of-the-art method for overlapping community detection in large networks). Finally, in Section 4.5 we draw some conclusions, and discuss the limitations of our method, as well as some directions for future research.

4.2 Background

4.2.1 Simplicial homology

In the fields of graph theory or network analysis, a graph $G = G(V, E)$ is defined by a list, V, of its vertices, as well as a list, E, of the pairs of vertices that are joined by an edge. An implicit assumption in this is that an edge $e = (v_i, v_j) \in E$ can only be present in G if both of its vertices v_i and v_j are in V. The notion of a simplicial complex is exactly a higher-order generalization of a graph, while similarly preserving this 'closed under subsets' property.

Definition (Simplicial complex): A k*-simplex* $\sigma = (v_0, v_1, \ldots, v_k)$ is a set of $(k+1)$ singleton elements (called *vertices*). A *simplicial complex* K is a set of simplices (i.e. a set of sets of vertices) such that

(i) if $\sigma, \tau \in K$, then $\sigma \cap \tau \in K$

(ii) if $\tau \leq \sigma$, then $\tau \in K$

where $\leq$ indicates the subset relation. If $\tau \leq \sigma$, we call τ a *face* of σ.

A simplex σ is *maximal* if there are no $\tau \in K$ such that $\sigma \leq \tau$. A k-simplex has dimension k. The *dimension* of simplicial complex K is the maximum dimension of any simplex in K

$$\dim(K) = \max_{\sigma \in K} \dim(\sigma).$$

A subset K' of a simplicial complex K is called a *subcomplex*, if K' is itself a simplicial complex (satisfying properties (i) and (ii) above). The k-skeleton of K is the subcomplex formed by all simplices in K with dimension at most k

$$k\text{-skeleton of } K = \{\sigma \in K \mid \dim(\sigma) \leq k\}$$

Given a graph $G = G(V, E)$, we can think of G as the 1-skeleton of a simplicial complex, whose higher-dimensional simplices have not been directly observed. The maximal simplicial complex whose 1-skeleton is equal to G is called the *flag complex*.

Definition (Flag complex): Given a graph $G = G(V, E)$, the simplicial complex

$$X(G) = \{\sigma = (v_{i_0}, v_{i_1}, \ldots, v_{i_{\dim\sigma}}) \mid (v_{i_j}, v_{i_k}) \in E \text{ for all } 0 \leq j, k \leq \dim\sigma\}$$

contains a simplex σ whenever all pairs of vertices in σ are connected by an edge in E. $X(G)$ is called the *flag complex* of G.

As we will see in Section 4.2.2.1, if we have additional information about the k-tuple relations in G, we may build a simplicial complex using that information, adding k-simplex σ whenever its vertices satisfy a k-tuple relation, and all faces of the simplex are also present. In the absence of such information, when only the graph G is given, we propose the use of the flag complex, and see that it can be very informative. Note that the Rips complex described in Section 2.2.1 was simply the flag complex on the communication graph of our sensor network.

A final notion we will review here is the definition of the *homology* of a simplicial complex.

Definition (Homology): We encode the structure of simplicial complex X through *boundary maps* $\{\partial_k\}_{k=1}^{\dim(X)}$, where ∂_k gives the oriented connectivity information between k-simplices and $(k-1)$-simplices. Then the k-th homology group of X is

$$H_k(X) = \ker(\partial_k) / \operatorname{im}(\partial_{k+1})$$

See Section 2.2.1 for a more detailed definition. Intuitively, the dimension of the k-th homology space counts the number of k-dimensional "holes" in the simplicial complex. These can be thought of as $(k+1)$-dimensional voids enclosed by k-simplices, so H_1 counts the number of loops which are not "filled-in" by triangles, and H_2 counts the number of voids. The interpretation of H_0 is slightly different: it counts the number of connected components of X (which may be interpreted as cycles of dimension zero).

For our purposes, we will not be computing any homology directly, but we will see that by preserving homology during our node dominance collapse, we will in fact be preserving important global structure of the network.

4.2.2 Node dominance

We will be representing a network using its flag complex, and in that setting, *node dominance* is characterized by the following definition.

Definition (neighbor set): The *neighbor set* of a node v, is the set of all nodes sharing an edge with v, as well as v itself:

$$\mathcal{N}(v) := \{u \in V \mid (u, v) \in E\} \cup \{v\}.$$

A node v is *dominated by* one of its neighbors w, if and only if

$$\mathcal{N}(v) \subseteq \mathcal{N}(w)$$

i.e.) all the neighbors of v are also neighbors of w.

To understand the importance and relevance of this definition, we will explore a bit of its history, and related concepts.

4.2.2.1 Homology of a relation

Definition (relation): A *relation* on two sets A and B is a function $r : A \times B \to \{0, 1\}$. We say that elements $a_i, a_j \in A$ are *related* (through element b) if there exists an element $b \in B$ such that $r(a_i, b) = 1$ and $r(a_j, b) = 1$. Similarly, $b_i, b_j \in B$ are related if there exists an $a \in A$ such that

$r(a,b_i) = 1$ and $r(a,b_j) = 1$. For A and B finite, the relation r can be represented by an $|A| \times |B|$ binary matrix $R = (r_{ij})$, where $r_{ij} = r(a_i, b_j)$.

As an example, the elements of set A could be actors, and the elements of set B could be movies, with $r(a,b) = 1$ whenever actor a appears in movie b.

Given a relation, there are two ways to encode its structure as a simplicial complex. The first way, which we will denote as $X_R(A,B)$, the elements of A are represented as vertices, and vertices $\{a_{i_0}, a_{i_1}, \ldots, a_{i_k}\}$ are spanned by a k-simplex whenever there exists a $b \in B$ such that $r(a_{i_l}, b) = 1$ for all $l = 0, 1, \ldots, k$. The second way, which we will denote as $X_R(B,A)$, the elements of B are represented as vertices, and $\{b_{j_0}, b_{j_1}, \ldots, b_{j_k}\}$ are similarly spanned by a k-simplex whenever they are all related by the same $a \in A$. Note also that for any simplicial complex X (even if it wasn't constructed using a relation) one may form its dual complex $\hat{X}$, by letting each maximal simplex in X correspond to a vertex in $\hat{X}$. In that case, a set of vertices in $\hat{X}$ are spanned by a simplex if their associated simplices in X all had a vertex in common.

In the example with actors and movies, this means that we can represent their relationships by building a simplicial complex where actors are vertices, and simplices are formed between actors who are in the same movie; or alternatively, we can encode it by using movies as vertices and spanning a set of movies by a simplex when they all feature the same actor.

Note that these two simplicial complexes may have drastically different structure (different number of vertices, different dimension), but Dowker [15] proved that the two complexes have exactly the same homology (in the sense that the k^{th} homology groups of the two complexes are isomorphic, for all k).

Theorem 4.2.1 (Dowker)**.** *If R is a relation on sets A and B, with associated simplicial complexes $X_R(A,B)$ and $X_R(B,A)$, then*

$$H_k(X_R(A,B)) \cong H_k(X_R(B,A)) \textit{ for all } k$$

4.2.2.2 Node dominance and equivalent notions

In light of the dual simplicial complexes presented in Section 4.2.2.1, we can now give the more general definition of node dominance.

Definition (Node dominance): Given simplicial complex X and its dual complex $\hat{X}$, each vertex $v \in X$ has an associated simplex $\sigma_v \in \hat{X}$. We say a vertex v is *dominated* by vertex w, if σ_v is a face of σ_w. This occurs exactly when the set of simplices incident to (i.e. containing) v is a subset of the set of simplices incident to w (in X).

When the simplicial complex of interest is a flag complex, we know that the presence of a higher dimensional simplex is determined by the presence of its constituent edges. This is why we are able to check the node dominance criterion using only the neighbor sets of our vertices, in the flag complex setting: if the neighbors of v are all neighbors of w, then the set of simplices incident to v is a subset of the set of simplices incident to w.

To illustrate the concept of node dominance using the example of actors and movies, consider two actors, represented by separate vertices a_i and a_j in $X_R(A,B)$. If the movies featuring actor a_i is a (proper) subset of the movies featuring actor a_j (i.e. a_i is dominated by a_j), then in the dual complex $X_R(B,A)$, the simplex σ_{a_i} will be a (proper) face of simplex σ_{a_j}. Thus, removing actor a_i (and all its incident simplices) completely, will not change the simplicial structure of the dual complex $X_R(B,A)$ at all, and thus will not change the homology of the original complex $X_R(A,B)$.

The insight that removing dominated nodes does not change the homology of the simplicial complex, suggests an algorithm, as originally proposed (independently) by [54] and [4], to simplify a simplicial complex by iteratively removing such vertices. In the work by Barmak and Minian [4], they term the removal of a dominated node a *strong homotopy collapse*, node dominance is a stricter condition than that required for a regular homotopy-preserving simplicial collapse [52].

In Figure 4.1, vertex v is dominated by vertex w, where vertex w could have additional connections in the network which are not shown. The removal of vertex v does not create or destroy any connected components, loops, or voids (preserves homology), and does not affect shortest path lengths between other nodes (see Section 4.3.1).

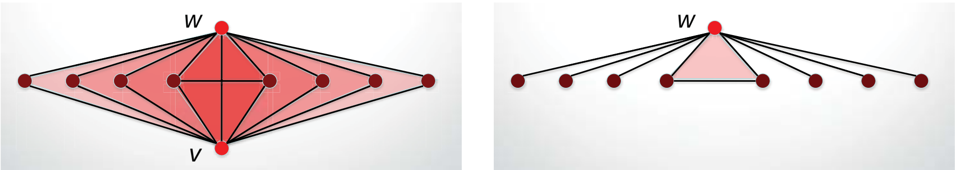

Figure 4.1 Node v dominated by node w.

One more definition we will note is that of a *2-hop neighbor set*, which is the neighbor set of a node that also contains all "friends of friends", instead of just immediate neighbors:

$$\mathcal{N}_2(v) = \{u \in V \mid (u,v) \in E, \text{ or } (u,v_i) \in E \text{ for some } v_i \in \mathcal{N}(v)\}$$

Performing the node dominance collapse using the 2-hop neighbor set can allow greater collapsability in networks with few dominated nodes. It also allows small holes in the flag complex (i.e. those with hop length ≤ 6) to be "filled in", so only larger homological features are preserved. We will use this version of the node dominance collapse on one of the data sets in Section 4.4.

4.2.2.3 Distributed algorithm for flag complexes

Assuming a flag complex structure, the node dominance collapse can be performed referring only to its 1-skeleton (the original graph under analysis). Moveover, the criterion for determining node dominance requires only local information, making the algorithm of distributed nature. This algorithm was first presented in [53].

Each node v has the list of its neighbor set $\mathcal{N}(v)$, and it then executes the following steps during each iteration:

```
Distributed algorithm for node dominance collapse
Broadcast $\mathcal{N}(v)$ to neighbors
for $v_i \in \mathcal{N}(v), v_i \neq v$
    Receive $\mathcal{N}(v_i)$
    if $\mathcal{N}(v_i) \subseteq \mathcal{N}(v)$
        Broadcast OFF to $v_i$
        if OFF received from $v_i$
            Handshake to determine if $v$ or $v_i$ turns off
        end if
    end if
end for
if OFF received OR Handshake determined $v$ turns off
    $v$ designated OFF
else
    Update $\mathcal{N}(v)$, omitting OFF neighbors
```

A very similar distributed algorithm is also possible in the non-flag complex setting, where there exists some *a priori* information about which k-tuples of simplices are related. An example of this would be the list of movies and actors, or some other relation (eg. authors/papers). In that case three actors (vertices) are only spanned by a triangle when there is a single movie they all appeared in together, not only if they had all appeared in movies together pairwise, as in the flag complex case. To compute node dominance in that setting, we only need to assume that each node has access to its *list of maximal simplices* (eg. an actor has its movie list, an author has its paper list, etc.). Then the algorithm above can proceed exactly as written, with $\mathcal{N}(v)$ replaced by the maximal simplex list of v.

4.3 Properties of core and periphery

In this section, we will outline both the theoretical and observed properties of the core-periphery decomposition obtained through the iterative node dominance collapse. Examples of the observed properties on real-world data sets are presented in Section 4.4.1.

Analytical properties:

1. Shortest paths in the core are shortest paths in the original network. *(Network flow)*
2. Nodes with betweenness centrality zero are not in the core *(Network flow)*
3. A node is dominated (with high probability) by a node sharing its community membership(s) *(Community structure)*
4. The homology of the flag complex of the core is the same as the homology of the flag complex of the entire network *(Global structure)*
5. The structure of the core is unique (all possible cores for a given network are isomorphic as simplicial complexes) *(Global structure)*

Observed properties:

- Core nodes typically have high degree and high betweenness centrality. ‘Hub’ nodes are in the core. *(Network flow)*
- Nodes with multiple ground-truth community membership labels tend to be in the core, while nodes with just one (or no) community labels are usually in the periphery. *(Community structure)*
- Using the peripheral groups, we can obtain candidate sets that are seen to contain a large proportion of ground-truth communities. See Section 4.4.3 for details, and our use of these candidate sets for community detection. *(Community structure)*

- The core is stable. In real-world networks, a very high proportion of nodes in the core are always there, regardless of random order collapses are executed in. *(Global structure)*

Throughout this section, for a graph $G = G(V, E)$, the core $G_C = G(V_C, E_C)$ is the graph induced by the set of nodes $V_C \subseteq V$ which remain upon an iterative and total removal of dominated nodes from V. Note that the set V_C (and thus the core itself) is not necessarily unique, because of a potential random 'handshake' in the Algorithm from Section 4.2.2.3. The statements given below are valid for any core obtained by the procedure of iterative node dominance collapse. As we will discuss further in Section 4.3.3 below, all possible cores obtained from the same initial graph have the exact same structure (are isomorphic) [38].

4.3.1 Network flow

The properties in this subsection involve statements about shortest paths between given nodes in the network. An outline of a proof similar to Property 4.3.1 is given in [54], and we include the complete proof here for completeness.

Definition (Shortest paths): Given a graph $G' = G(V, E)$, for any pair of points $v_i, v_j, \in V$, a *path* $p = (v_i = v_1, v_2, \ldots, v_l = v_j)$ is a sequence of vertices such that $(v_k, v_{k+1}) \in E$ for all $k = 1, \ldots, l-1$. The path has length $|p| = l$, and p is a *shortest path* if $l \leq |p'|$ for any other path p' from v_i to v_j. The set of all shortest paths from v_i to v_j, in the graph G' is denoted $SP_{G'}(v_i, v_j)$.

Property 4.3.1 (Shortest paths in the core are shortest paths in the original network.)**.** *For $v_1, v_2 \in V_C$, if $p \in SP_{G_C}(v_1, v_2)$, then $p \in SP_G(v_1, v_2)$.*

Proof. For any graph G', let v_j be dominated by its neighbor v_i. Consider any shortest path $p = (\ldots, v_k, v_j, v_l, \ldots)$ passing through v_j. Note that $k, l \neq j$ [Proof by contradiction: $p = (\ldots, v_i, v_j, v_l, \ldots)$ could be replaced by shorter path $(\ldots, v_i, v_l, \ldots)$, since $\mathcal{N}(v_j) \subseteq \mathcal{N}(v_i)$ so $v_l \in \mathcal{N}(v_j) \Rightarrow v_l \in \mathcal{N}(v_i)$]. So $p = (\ldots, v_k, v_j, v_l, \ldots)$ can be replaced by $p' = (\ldots, v_k, v_i, v_l, \ldots)$, which is the same length as p, but doesn't contain v_j.

Therefore, the length of all shortest paths in G' (where v_j is not the source or destination) are preserved when v_j is removed. □

Definition (Betweenness centrality): The betweenness centrality of a node v is a measure of the number of shortest paths passing through v. It is defined as the proportion of shortest paths between nodes s and t that pass through v, summed over all pairs $s, t \neq v$. i.e.)

$$\mathrm{bc}(v) = \sum_{s,t \neq v} \frac{|\{p \in SP_G(s,t) | v \in p\}|}{|SP_G(s,t)|}$$

Property 4.3.2 (Nodes with betweenness centrality zero are not in the core)**.**

$$\mathrm{bc}(v) = 0 \Rightarrow v \notin V_c$$

This can be equivalently stated as: nodes with betweenness centrality zero are dominated.

Proof. Using the definition of betweenness centrality above, we can see that

$$bc(v) = 0 \Rightarrow |\{p \in SP_G(s,t) | v \in p\}| = 0 \; \forall s, t \neq v.$$

Therefore, either

(i) $\deg(v) = 1$

(ii) $\forall s, t, \in \mathcal{N}(v), (s,t) \in E$ (so that $\ldots, s, v, t, \ldots$ will not be in any shortest path)

If (i), then v is dominated.

If (ii), then $\mathcal{N}(v)$ is a clique, so for any $w \in \mathcal{N}(v)$ with $w \neq v$, $\mathcal{N}(v) \subseteq \mathcal{N}(w)$, so v is dominated.

Therefore $v \notin V_C$. □

Both of these properties speak to the 'centrality' of the nodes in the core, with respect to the original network. Property 4.3.1 tells us that there is no way to shortcut through the periphery when traveling between two nodes in the core, and Property 4.3.2 says the nodes that are not involved in any shortest paths are guaranteed to be contained in the periphery. Together, we can conclude that the node dominance collapse only has local effects (with respect to shortest paths in the network), in that only shortest paths beginning or ending at the dominated node are affected.

Empirically we see that nodes with high betweeness centrality and nodes with high degree will lie in the core (see Section 4.4.1 for concrete examples). These are 'hub' nodes, in terms of network flow properties, so removal of nodes in the core have a much greater impact on network information flow than removal of nodes from the periphery.

4.3.2 Community structure

The community affiliation graph model (AGM) proposed by Yang and Leskovec [56] assumes that the probability of an edge forming between two nodes depends on the community membership(s) of the nodes under consideration. This is similar to the traditional stochastic blockmodel (which require communities to form a partition of the network), or generalizations [2] of the stochastic blockmodel that allow for overlapping communities, with the notable exception that under AGM the edge density in the intersections of communities is *higher* than the edge density in the non-overlapping portions of communities.

For notation, consider the set $C = \{c_k\}_{k=1}^m$ defining the m communities in the network, where c_k is the set of nodes belonging to the k^{th} community. Note that each node in V may belong to zero, one, or multiple communities. For two nodes $u, v \in V$, let $C_{uv} = \{c \in C \mid u, v \in c\}$ denote the set of communities containing both u and v. We will also use the more general notation $C_S = \{c \in C \mid \exists v \in S \text{ s.t. } v \in c\}$ to denote the set of community memberships for nodes in a given set S. Under AGM, an edge forms between u and v, independently, with probability p_c for each of the communities $c \in C_{uv}$. In other words, denoting the probability of an edge between u and v by

$p(u,v) = P[(u,v) \in E]$, we have

$$p(u,v) = 1 - \prod_{c \in C_{uv}} (1 - p_c). \tag{4.1}$$

Further, Yang and Leskovec define a baseline edge probability $\varepsilon = p(u,v)$ for u, v with no communities in common. They choose $\varepsilon = \frac{2|E|}{|V|(|V|-1)}$, which is typically a number of orders of magnitude smaller than the p_c probabilities. For the proof of the following result, we assume the AGM model for network community structure, however the result would still hold for any model that bases the probability of an edge between two nodes on the community membership of the nodes, where the probability of an edge is significantly higher for nodes sharing communities than nodes not sharing communities.

Property 4.3.3 (Nodes are dominated, with high probability, by nodes that share the community memberships of their neighbor set)**.** *If v is dominated by w, then with high probability, $C_{\mathcal{N}(v)} \subseteq C_w$.*

Proof.

$$\begin{aligned} P[v \text{ dominated by } w] &= \prod_{v_i \in \mathcal{N}(v)} p(w, v_i) \\ &= \left(\prod_{\substack{v_i \in \mathcal{N}(v) \\ C_{wv_i} \neq \emptyset}} \left[1 - \prod_{c \in C_{wv_i}} (1 - p_c) \right] \right) \prod_{\substack{v_i \in \mathcal{N}(v) \\ C_{wv_i} = \emptyset}} \varepsilon \end{aligned}$$

In other words, v will be dominated by w, only if there exist edges between w and all $v_i \in \mathcal{N}(v)$. Each of these edges occurs independently, with probability $p(w, v_i)$, with the value given in Equation (4.1) if w and v_i share community membership(s) (i.e. if $C_{wv_i} \neq \emptyset$), and $p(w, v_i) = \varepsilon$ otherwise. Since $\varepsilon \ll p_k$ for all k,

$$P[(w, v_i) \in E \mid C_{wv_i} \neq \emptyset] \gg P[(w, v_i) \in E \mid C_{wv_i} = \emptyset]$$

Therefore

$$P[v \text{ dominated by } w \mid C_{\mathcal{N}(v)} \subseteq C_w] \gg P[v \text{ dominated by } w \mid C_{\mathcal{N}(v)} \nsubseteq C_w]$$

□

Observationally (as described in Section 4.4.1), nodes in the periphery typically have one (or no) community membership(s), while nodes in the core have multiple community memberships, and lie in the intersections of communities. In Section 4.4.3, we will take this interpretation further, by proposing a method for using the peripheral components to obtain candidate sets which are likely to contain communities of the network. We can think of the peripheral components as the non-overlapping portions of the communities, in which case the true network communities would consist of a peripheral component, along with adjoining nodes in the core. It is also possible that a single community could have non-overlapping portions which “stick out” from the core in multiple places, on account of which we propose a method of combining peripheral components according to which core nodes they connect to. This yields an algorithm for obtaining “candidate sets” which intended to contain the true network communities. This method is discussed further in Section 4.4.3.

4.3.3 Global structure

As described in Section 4.2.2, when the flag complex representations of the original network and the core network are used, the core is seen to have the exact same homology as the original complex, in the sense that their homology spaces are isomorphic in all dimensions.

Property 4.3.4 (Homology is preserved in the core)**.**

$$H_k(X(G_C)) \cong H_k(X(G)) \textit{ for all } k$$

Proof. This property follows immediately from the result of Dowker's Theorem (that a simplicial complex and its dual complex have the same homology), combined with the observation that if a vertex is dominated, its corresponding simplex in the dual complex will be a face of the simplex corresponding to the dominating node, and thus will not contribute to the structure of the dual complex.

An alternative formulation and proof is available in [4]. □

A corollary of Property 4.3.1 is that at least one shortest cycle for each homology class is retained in the core. Thus, not only is the dimension of each homology space preserved, but the 'hole locations' in the network are also preserved. It is this additional property that truly allows us to interpret the core as the global scaffolding for the network.

Property 4.3.4, together with Property 4.3.3 tell us that nodes with diverse friend sets (including bridging ties) will be in the core. If they are not, it is only because they are dominated by another node with all the same diverse connections. In real-world networks, we see that the average clustering coefficient for nodes in the core is much lower than in the network as a whole (see Section 4.4.1), which supports the 'diverse friend set' interpretation, because the friends of a core node are usually not friends with each other.

4.4 Analysis of real-world networks

We will use two data sets in this section as a running illustration, both obtained from the Stanford SNAP network database [32]. The first is a coauthorship network built from the DBLP computer science bibliography, and the second is a co-purchasing network from Amazon. The networks were originally analyzed by Yang and Leskovec [57] in one of the first papers to systematically analyze the properties of ground-truth communities (abbreviated in figures as GTCs) in real-world networks. Both communities have ground-truth community labels: 13,477 ground-truth community labels in DBLP, defined as connected components of authors within the same publication venue; and

271,570 ground-truth community labels in Amazon, defined using product categories. Additionally, Yang and Leskovec labeled 5000 of the communities in each data set as “best” in terms of having community-like properties such as low conductance or high triangle-participation ratio. We computed the core-periphery decomposition for both networks using the iterative node dominance collapse algorithm described in Section 4.2.2.3. For the Amazon co-purchasing network, the periphery consisted of 70716 nodes (accounting for only 21% of the nodes in the network), each of which were singletons, connected only to the core and not to other peripheral nodes. To allow further collapse, we re-computed the core using the 2-hop neighbor sets $\mathscr{N}_2(v)$ described in Section 4.2.2.2. This yielded 193,195 nodes in the periphery (57.7% of the nodes in the network), with 70716 peripheral components, of which 20136 were non-singletons (of varying sizes). All analysis presented below uses the regular node dominance collapse on the DBLP data set, and the node dominance collapse based on 2-hop neighbor sets for the Amazon data set.

Descriptive statistics for the networks, as well as for their associated core-periphery partitions, are presented in Table 4.1. For the computations of average degree and clustering coefficient, the values were computed with respect to the entire network, and again with respect to the induced subgraph under consideration (either the core or periphery).

To verify the stability of the core under multiple realizations of the node dominance collapse algorithm, we performed the following randomization: For one realization of the iterated node dominance collapse, we would compute the set of dominated nodes, pick one at random to collapse, add the newly dominated nodes to the set of dominated nodes, randomly pick the next dominated node to collapse, and so on. After performing 100 realizations of the core-periphery decomposition on the two data sets, we found that 99.58% (DBLP) and 99.43% (Amazon) of the nodes in the core were present in the core on every realization. The set of nodes that appeared in the core on some (but not all) realizations was 0.89% (DBLP) 1.24% (Amazon) the size of the core. Thus, not only is the shape of the core unique, but the actual nodes composing it are very stable in these real-world data sets.

Table 4.1 Descriptive statistics for real-world networks.

	DBLP	Amazon
Nodes in core:	71,018	141,688
Nodes in periphery:	246,062	193,195
Nodes (total):	**317,080**	**334,863**
Edges within core:	318,741	347,527
Edges within periphery:	274,367	218,237
Edges between core and periphery:	456,758	360,108
Edges (total):	**1,049,866**	**925,872**
Mean degree:		
Entire network	6.62	5.53
Core (w.r.t entire network)	15.41	7.45
Core (w.r.t. core)	8.98	4.91
Periphery (w.r.t entire network)	4.09	4.12
Periphery (w.r.t periphery)	2.23	2.26
Clustering coefficient:		
Entire network	0.632	0.397
Core (w.r.t entire network)	0.285	0.219
Core (w.r.t. core)	0.255	0.182
Periphery (w.r.t entire network)	0.733	0.527
Periphery (w.r.t periphery)	0.385	0.293
Communities (total):		
Number	13,477	271,570
Average size	53.41	11.67
Standard deviation of size	257.58	273.66
Communities (best):		
Number	5000	5000
Average size	22.45	13.49
Standard deviation of size	201.08	17.52

4.4.1 Relationship of core-periphery to network structure

For both data sets, we observe (Table 4.1) that nodes in the core have higher degree than nodes in the periphery, with the difference especially pronounced in the DBLP network. Additionally, nodes in the core have lower clustering coefficient, which corroborates our intuition that core nodes have "diverse friend sets", so their friends are not all friends with each other. Along with their high degree, this is also interpretable as having reach outside of their local community.

Scatterplots showing the natural logarithm of betweenness centrality versus node degree are shown in Figure 4.2, with the two plots of the same data alternating whether core or periphery is plotted on top, to help display the region of overlap. As mentioned in Section 4.3.1, all nodes with betweenness centrality of zero (i.e. nodes through which no shortest paths pass) are guaranteed to be in the periphery, and we observe that additionally, all of the nodes with highest betweenness centrality are in the core. For example, in Figure 4.2, it can be seen that in the DBLP data set there is a threshold betweenness centrality value (around $\ln(\text{bc}) = 17$), above which all nodes are in the core, while in the Amazon data set, it is the nodes with both high degree and high betweenness centrality that appear exclusively in the core.

Figure 4.3 shows the number of ground-truth community (GTC) assignments per node in the core and periphery. Out of all the nodes in the periphery, 22.11% had no ground-truth community membership labels, 57.39% had exactly one, and 20.49% had more than one GTC membership label. On the other hand, out of the nodes in the core 85.02% had multiple GTC membership labels, while 12.65% had a single community, and only 2.33% had no GTC label. From another perspective, the periphery contained 97.05% of the nodes without a GTC label, 94.02% of the nodes with a single label, but 45.51% of the nodes with multiple labels (however of those nodes multiply labeled, the average number of labels was 2.9 in the periphery, but 7.0 in the core). A similar behavior is observed in the Amazon network, albeit to a lesser extent, and is likely due to the average number of labels per node being much higher.

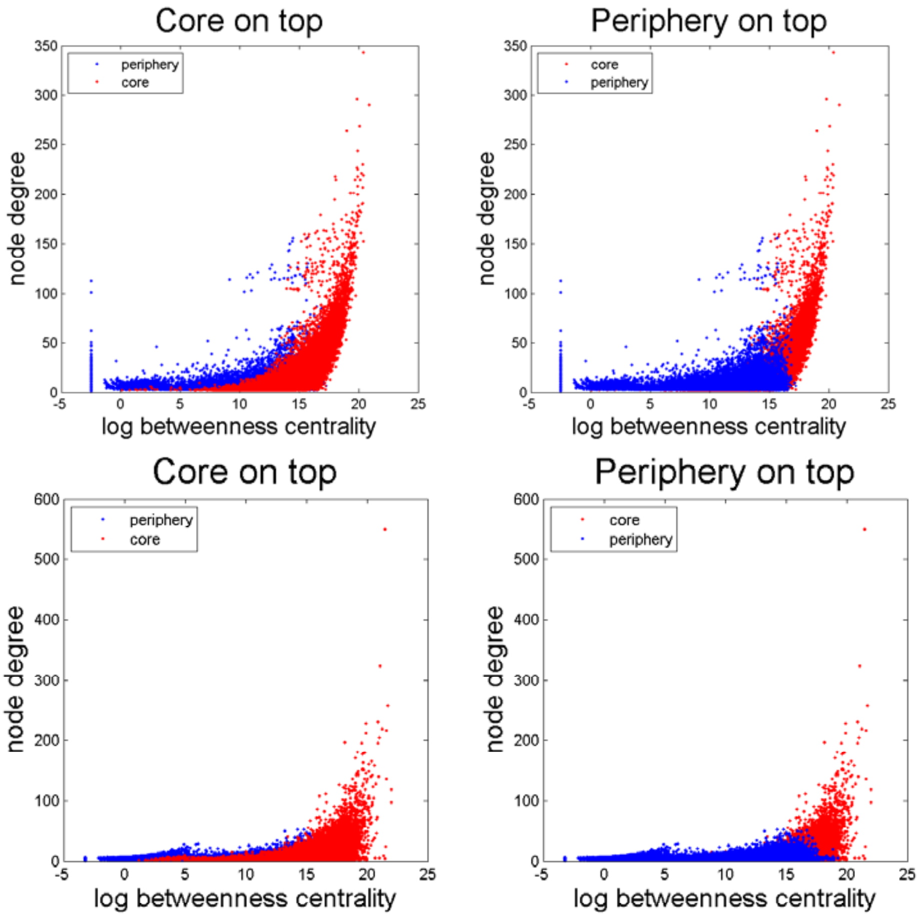

Figure 4.2 Log betweenness centrality vs degree (DBLP-top, Amazon-bottom).

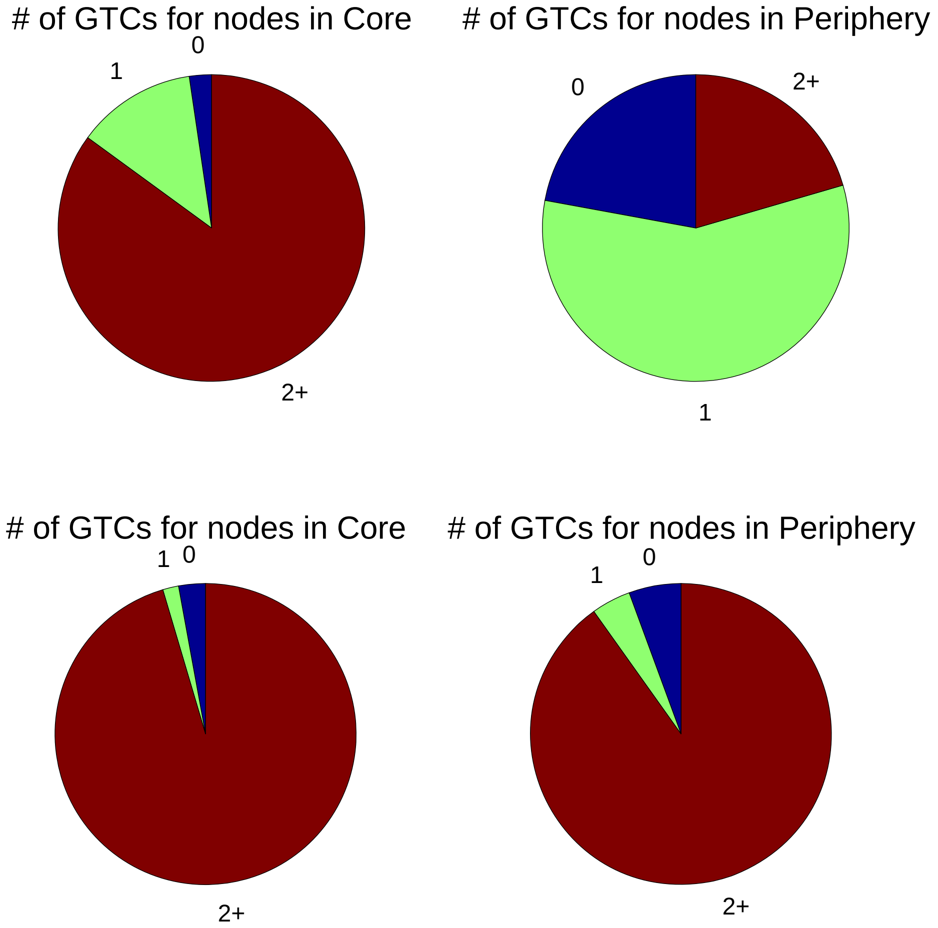

Figure 4.3 Number of community memberships (DBLP-top, Amazon-bottom)

4.4.2 Role of core in network flow

To demonstrate the key role our core nodes play in information flow over the network, we computed their contribution to the shortest paths of the network. For each network, we randomly chose 1000 pairs of nodes, and computed shortest paths between them. Since 100% of these paths contain at least one node from the core, we computed the proportion of each path that is in the core. It is worth noting here that for any two nodes which do not belong to the same peripheral component, all shortest paths between them will pass through the core. For comparison, we chose three sets of nodes, each with the same number of nodes as the core: chosen uniformly randomly; using the nodes of highest degree; and using the nodes with highest betweenness centrality. Then, using the same 1000 shortest paths, we computed the proportion of nodes from each path belonging to each of these sets. Taking the average over all 1000 paths, the mean proportion of each path contained in the four sets (Core, Highest BC, Highest Degree, and Random) are shown in Table 4.2. Since betweenness centrality measures how many shortest paths pass through a node, the nodes with highest betweenness centrality should be the optimal choice for this measure (if considering all shortest paths in the entire network), so it is not surprising that they have the highest proportion of shortest path nodes. What is somewhat more surprising, is that for both data sets, the nodes in the core out-perform the nodes with highest degree, so a greater proportion of nodes in shortest paths belong to the core, than belong to the equal-sized set of highest degree nodes. The proportion of nodes in the shortest paths that belong to the Random set give us a baseline probability from which to compare the other choices of “important” nodes. Recall also, that betweennness centrality is very computationally expensive, requiring global information, so it is useful that the distributed core-periphery computation be nearly comparable at obtaining nodes central to network flow.

Table 4.2 Proportion of important nodes in shortest paths.

Proportion of nodes in shortest paths belonging to important sets		
	DBLP	Amazon
Highest BC	0.785	0.892
Core	**0.753**	**0.841**
Highest degree	0.739	0.698
Random	0.222	0.427

4.4.3 Community detection

The findings of this study are consistent with the community affiliation graph model (AGM) of Yang and Leskovec [56, 59], in the sense that it supports an overlapping community model for social and information networks where the probability of an edge between two nodes is related to their common community membership(s), with higher probabilities of edges between nodes that have multiple communities in common. Under this model, we showed that nodes are only dominated (with very high probability) by nodes which share their community memberships. Interpreting our peripheral components with respect to this model, they appear to be the ‘non-overlapping’ parts of communities that stick out of the network. Figure 4.4 shows embeddings of some peripheral components from the DBLP data set as examples, where the peripheral component is drawn in black, while the core nodes and connecting edges are grey. The internal structure and connectivity to the core can vary considerably between peripheral components.

In light of the interpretation of peripheral components as non-overlapping portions of communities, we propose an algorithm which consists of taking unions of these peripheral components, along with their neighboring nodes in the core, to obtain candidate sets for community detection.

More precisely, let $PC = \{pc_i\}_{i=1}^{|PC|}$ denote the set of peripheral components in the network, where each node in the periphery is in exactly one peripheral component, pc_i. Then define the

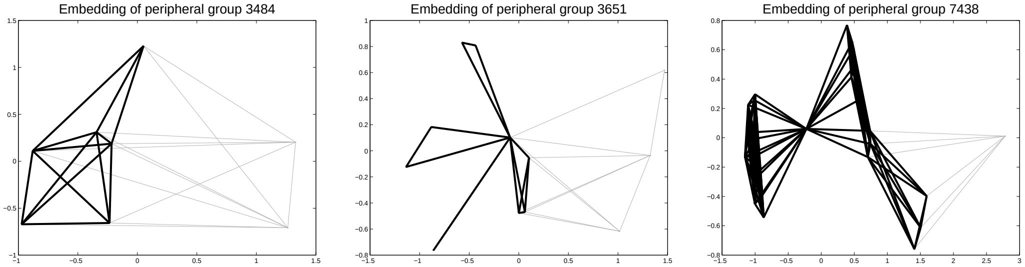

Figure 4.4 Example peripheral components.

extended peripheral components $PC^+ = \{pc_i^+\}_{i=1}^{|PC|}$ where

$$pc_i^+ = \{v \in V_c \mid \exists\ v_j \in pc_i \text{ s.t. } (v_j, v) \in E\} \cup pc_i,$$

so each extended peripheral component additionally contains all the nodes in the core that share an edge with a vertex of the peripheral component. The extended peripheral components are meant to approximate ground-truth communities in the data set, however there are large numbers of very small size (such as those consisting of an isolated peripheral node and its single neighboring core node). We consolidate extended peripheral components into "candidate sets" by taking, for each $v \in V_C$, the union of all extended peripheral groups that include v. So we obtain $\{cs_v\}_{v \in V_C}$, where

$$cs_v = \bigcup_{\substack{pc_i^+ \in PC^+ \\ v \in pc_i^+}} pc_i^+.$$

For example, if there were many peripheral nodes connected to a single core node (but not connected amongst each other), this group would be consolidated into a single candidate set. We then remove any candidate sets cs_v that are repetitions or subsets of other candidate sets, to obtain our final set of maximal candidate sets: CS. Intuitively, our candidate sets are meant to approximate ground truth communities, or unions of ground truth communities (that overlap on common core nodes).

To judge the performance of our candidate sets for the purposes of community detection, we also ran the BIGCLAM algorithm [58] on the DBLP data set. Popular methods for detection overlapping communities include clique percolation, link clustering, and fuzzy detection methods using mixed-membership stochastic block models (see [55] for a survey), however none of these methods scale up well to networks with hundreds of thousands or millions of nodes. The recent exception to this is Yang and Leskovec's BIGCLAM algorithm, which can estimate the overlapping community structure for large networks. Their algorithm (available in the SNAP C++ package [33]) allows the user to input the expected number of communities, but runs into memory problems if the number of communities is larger than a few hundred. It also has an option for the algorithm to learn the appropriate number of communities, with a default to test between 5 and 100 communities. Therefore, to obtain a set of communities of the same order as the number of ground-truth communities (13,477 for the DBLP data set), we performed BIGCLAM in a nested manner. First obtaining 100 communities, and then further subdividing each of these, where the optimal number of subcommunities was most often also 100. This yielded a total of 9904 detected communities from the BIGCLAM algorithm. We used the same method for analysis of the Amazon data set, yielding 8899 BIGCLAM communities, even though that network has a much larger number of ground-truth communities (271,570). For both data sets, the number of candidate sets obtained using our method was around 40,000 (47,134 for DBLP and 37,449 for Amazon).

To measure the fit of the candidate sets and BIGCLAM communities to the ground-truth communities, we used precision, recall, and average F1 score. For a detected community C_1 and ground truth community C_2 (the target), the *precision* is the proportion of detected nodes that belong to the target:

$$precision(C_1, C_2) = \frac{|C_1 \cap C_2|}{|C_1|},$$

the *recall* is the proportion of target nodes captured in the detected community:

$$recall(C_1, C_2) = \frac{|C_1 \cap C_2|}{|C_2|},$$

and the F1-score is the harmonic mean of precision and recall:

$$F1(C_1, C_2) = \frac{2 \cdot precision(C_1, C_2) \cdot recall(C_1, C_2)}{(precision(C_1, C_2) + recall(C_1, C_2))}.$$

These three values for a given ground-truth community are obtained by maximizing each over all candidate sets (or BIGCLAM communities), and an average precision, recall, and F1-score for the ground-truth communities is obtained. Similarly, the three values are obtained for each candidate set (or BIGCLAM community) by thinking of it as the “target” community, and maximizing precision, recall, and F1-score over all ground-truth communities, and then taking the average of these maxima.

Using all three of these values (precision, recall, and F1-score) helps offset some of the discrepancies caused by the varying numbers of ground-truth communities, candidate sets, and BIGCLAM communities. Since the matching of ground-truth communities onto detected communities, but also the matching of detected communities onto ground-truth communities, are considered, having more candidate sets than BIGCLAM communities will not necessarily be an advantage.

Table 4.3 gives the values for recall, precision and F1-score when comparing the ground-truth communities to our candidate sets (left three columns), and to the BIGCLAM communities (right three columns). The performance using candidate sets and BIGCLAM communities are compared for each measure (eg. “ground-truth community recall”, or “ average precision”), with the values in boldface indicating the method (candidate sets or BIGCLAM) with superior performance in that measure. The column “ground-truth” gives the average values for the ground truth communities (when maximized over the detected communities), and the column “detected” gives the average

Table 4.3 Detection of all GTCs by candidate sets and BIGCLAM communities.

	DBLP (all 13,477 communities)					
	Candidate sets			BIGCLAM		
	ground-truth	detected	average	ground-truth	detected	average
Recall	**0.7620**	**0.5401**	**0.6511**	0.7418	0.4478	0.5948
Precision	**0.4319**	0.4960	**0.4640**	0.2366	**0.6261**	0.4314
F1-score	**0.4233**	0.2565	**0.3399**	0.2696	**0.2721**	0.2709

	Amazon (all 271,570 communities)					
	Candidate sets			BIGCLAM		
	ground-truth	detected	average	ground-truth	detected	average
Recall	0.8481	**0.8721**	0.8601	**0.9213**	0.8203	**0.8708**
Precision	**0.2545**	0.8728	**0.5636**	0.1124	**0.9861**	0.5492
F1-score	**0.3218**	**0.4815**	**0.4017**	0.1611	0.4685	0.3148

for the detected communities (when maximized over ground-truth communities).

Our candidate sets give better overall community detection performance than the BIGCLAM communities (as measured by the average F1-score). For the DBLP data set, the ground-truth communities were contained in the candidate sets (based on higher ground-truth recall scores), more so than the candidate sets found strongly-matching ground-truth communities (although it is worth noting, as Yang and Leskovec did, that not all “true” ground-truth communities necessarily have ground-truth community labels in this data set). The performance on the Amazon data set is quite good, with very high ground-truth recall and detected recall and precision for both the candidate sets and the BIGCLAM methods, although our candidate sets out-performed BIGCLAM in detected recall, as well as ground-truth, detected and average F1-scores.

The analysis was repeated using only the 5000 “best” ground-truth communities, and again the candidate sets resulted in higher average F1-scores than the BIGCLAM communities. The main difference was that recall for the ground-truth communities increased (on average, each ground-truth community had a candidate set it was 94% contained in), while recall and precision for the

Table 4.4 Detection of 5000 best GTCs by candidate sets and BIGCLAM communities.

	DBLP (5000 best communities)					
	Candidate sets			BIGCLAM		
	ground-truth	detected	average	ground-truth	detected	average
Recall	**0.9414**	0.2559	**0.5987**	0.9054	**0.2678**	0.5866
Precision	**0.4313**	0.3121	**0.3717**	0.3065	**0.4216**	0.3640
F1-score	**0.5221**	0.1446	**0.3333**	0.3840	**0.1913**	0.2877

	Amazon (5000 best communities)					
	Candidate sets			BIGCLAM		
	ground-truth	detected	average	ground-truth	detected	average
Recall	**0.9893**	0.0222	**0.5058**	0.9072	**0.0728**	0.4900
Precision	**0.4781**	0.0404	0.2593	0.4535	**0.1224**	**0.2880**
F1-score	**0.5753**	0.0241	**0.2997**	0.5100	**0.0753**	0.2927

candidate sets decreased (since there were fewer ground-truth communities to match to, fewer detected had a well-matched ground-truth community). It is also worth noting that for the DBLP data set 81.7% of the best ground-truth communities were completely contained in at least one candidate set, while 73.8% of the best ground-truth communities were completely contained in at least one BIGCLAM community. For the Amazon data set, these values were 94.8% for the candidate sets, and 82.8% for the BIGCLAM communities.

The challenge of detecting many thousands of overlapping communities from a large network is formidable. Currently there are no available methods which achieve excellent performance when comparing detected to ground-truth communities. Based on the analysis of two large, real-world data sets with ground-truth community information, our proposed algorithm of obtaining candidate sets from the peripheral components of the core-periphery decomposition, yielded better community detection results than the state-of-the-art BIGCLAM algorithm for overlapping community detection.

4.5 Conclusion

This study posed the question “How does the concept of node dominance relate to local and global properties of a network?”. Previous work determined that iteratively removing dominated nodes is a homology-preserving way to perform a collapse/simplification of a simplicial complex [4] [54]. This was extended into a distributed algorithm for the case of flag complexes [53]. Here, we undertook an investigation of the theoretical and practical properties of performing such a collapse on social and information networks, and discovered that it has implications for both a core-periphery decomposition of the network, as well as uncovering network community structure.

The properties of the core and periphery that we developed in Section 4.3, and observed in Section 4.4, lead to the interpretation that nodes in the core obtained using node dominance collapse are important with respect to network flow, to the global structure of the network, and to the network community structure.

The core nodes are essential to network flow because of two properties: a shortest path between any two points in the core is contained in the core; and nodes with betweenness centrality zero (through which no shortest paths pass) are never in the core. Observationally, ‘hub’ nodes are contained in the core, and core nodes often have high degree and high betweenness centrality.

The global structure of the network is preserved in the core because the homology of the core is the same as the homology of the entire network, when considering the respective flag complexes. This can be interpreted as node dominance collapses only having ‘local’ effects, and that nodes with diverse neighbor sets (including bridging ties) are members of the core, maintaining a scaffolding for the global structure of the network. The observation that each core node typically has a diverse neighbor set (their friends are not all friends with each other) is also quantified by their relatively low clustering coefficient values.

Finally, the core is related to the community structure of the network because under community membership models where within-community connections have significantly higher probability than cross-community connections, we see that nodes are dominated (with high probability) by nodes that share their community membership(s). In real-world networks with overlapping ground-truth community labels, this is observed through nodes with multiple community memberships typically residing in the core, and through nodes with single (or no) community labels occupying the periphery.

The result relating the core-periphery to the community structure of the network gives us an additional application: the use of the peripheral components to generate “candidate sets” which are likely to contain the true network communities. Many state-of-the-art community detection algorithms which allow for overlapping communities, are not scalable past network sizes of a few thousand nodes. The notable recent exception is Yang and Leskovec’s BIGCLAM algorithm, which our method is shown to outperform on their DBLP and Amazon data sets.

Implications of this work may be of interest not only to researchers explicitly interested in a core-periphery decomposition of complex networks, but to anyone studying community structure, or key nodes for network flow. Hopefully this work will also serve to further popularize the node dominance collapse for use in general contexts where data is represented using a simplicial complex structure.

One limitation of our method is that some networks don’t collapse significantly using node dominance. For example, on Facebook there are very few people who have a friend list completely contained in the friend list of another person. One option for future research in this direction would involve performing the node dominance collapse locally on ego networks, and consolidating the resulting communities. Another potential drawback is the nondeterministic nature of the node dominance collapse algorithm. Perhaps under some circumstances it would be wise to consider the set of nodes that are “ever in the core”, or “always in the core”, under repeated realizations of the algorithm. In practice however (Section 4.4.1), we have seen that these two sets are quite similar.

One other area for future research is in the study of the core under a graph evolution. Either using observed or model-generated dynamic networks, studying how the core varies over time could be used to help evaluate or predict community structure and key players in the network.

CHAPTER

5

CONCLUSIONS AND FUTURE WORK

5.1 Conclusions

This dissertation explored how a computational topology approach can be used to study dynamic and complex networks. We showed how techniques such as simplicial complex representations, in computing and tracking homology classes, and in performing homology-preserving simplifications, can be greatly fruitful in the context of network analysis.

In a sensor network setting, we developed a method for choosing specific representative cycles when tracking the homology of a time-varying sensor network with zigzag persistent homology. The barcode output of zigzag persistence can be used as a descriptor of the dynamic coverage of the network, and we showed that it can be used to distinguish between types of stochastic mobility patterns for the mobile nodes. Additionally, using our chosen representative cycles along with a hop-distance filtration, we were able to attach size estimates onto the bars at each time point, thus providing a richer quantitative descriptor of the dynamic network coverage, as well as an informative visualization tool in the weighted barcode. All of this was carried out while only analyzing the network through a series of snapshots of its communication graph, with no information about coordinates, or edge lengths.

Turning to social networks, we found that the node dominance condition [54] (i.e. strong homotopy collapse [4]) and the iterative application of node dominance removal for network simplification [53] resulted in a core-periphery decomposition of a network that had very relevant features. As summarized in Section 4.5, nodes in the core, display both theoretical and observational importance with respect to network flow, and form the global scaffolding of the network. The peripheral components are seen to be related to the overlapping community structure of a network, in that they are the non-overlapping parts sticking out from the core, thus yielding an algorithm for using the peripheral components and their adjoining nodes in the core. This, in turn, builds “candidate sets” which are seen to perform better than a state-of-the-art method for overlapping community detection in large networks.

In both cases, using the flag complex representation of the network, and studying properties related to its homology, gave us insights into the data that would not have been possible using the traditional graph and network analysis viewpoint.

5.2 Future Work

The methods presented here have been developed specifically for network data, because traditional methods from topological data analysis were more geared towards point cloud data observed in a metric space (such as $\mathbb{R}^n$). Perhaps somewhat paradoxically, we are now considering whether these methods for network analysis could be extended back for purposes of general data analysis. First, consider a setting where the networks under analysis have edge weights. This naturally induces a filtration, or nested sequence of graphs, by including all edges with weights below (or above) some threshold, and obtaining the resulting flag complexes. Persistent homology analyzes such a nested sequence through the changing homology. Perhaps without explicitly computing the homology, simply studying the core-periphery decomposition at each level in the filtration, would be informative. Indeed, such a procedure could be applied to the nested sequence of simplicial complexes obtained from point cloud data.

Similarly, the method of tracking representative cycles through a (regular or zigzag) persistence computation, has a clear geometric interpretation for sensors roaming in the plane, but could still prove useful for more general settings. For a regular persistent homology analysis, the semi-canonical choice of representative cycles is only available (through Alexander Duality) in the (top-minus-one)-dimensional homology, so situations where features of this dimension are of interest seem to have the greatest potential for applications. For example, instead of studying cycles in the plane, as was done for the coverage problem, it could be applied to studying voids in data that is innately three-dimensional (which is often of interest in materials science).

In terms of social network analysis, there are a number of generative models for network growth and change over time. Studying the evolution of the core under such models can shed light on how “key players” in a network develop over time. Additionally, comparing the model-based evolution of the core to the evolution of the core observed in real-world networks, could highlight aspects which may be unrealistic, or help validate the generative model.

BIBLIOGRAPHY

[1] Adams, H. & Carlsson, G. “Evasion paths in mobile sensor networks”. *The International Journal of Robotics Research* **34**.1 (2015), pp. 90–104.

[2] Airoldi, E. M. et al. “Mixed membership stochastic blockmodels”. *Advances in Neural Information Processing Systems*. 2009, pp. 33–40.

[3] Akyildiz, I. et al. “A survey on sensor networks”. *Communications Magazine, IEEE* **40**.8 (2002), pp. 102–114.

[4] Barmak, J. A. & Minian, E. G. “Strong homotopy types, nerves and collapses”. *Discrete & Computational Geometry* **47**.2 (2012), pp. 301–328.

[5] Borgatti, S. P. & Everett, M. G. “Models of core/periphery structures”. *Social networks* **21**.4 (2000), pp. 375–395.

[6] Carlsson, G. “Topology and data”. *Bulletin of the American Mathematical Society* **46** (2009), pp. 255–308.

[7] Carlsson, G. et al. “Zigzag persistent homology and real-valued functions”. *Proc. 25th Annual Symposium on Computational Geometry (SoCG)* (2009), pp. 247–256.

[8] Carlsson, G. & Silva, V. de. “Zigzag persistence”. *Foundations of Computational Mathematics* **10**.4 (2010), pp. 367–405.

[9] Chambers, E. W. et al. “Vietoris–rips complexes of planar point sets”. *Discrete & Computational Geometry* **44**.1 (2010), pp. 75–90.

[10] Chan, P. K. et al. “Spectral k-way ratio-cut partitioning and clustering”. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* **13**.9 (1994), pp. 1088–1096.

[11] Cohen-Steiner, D. et al. “Stability of persistence diagrams”. *Discrete & Computational Geometry* **37**.1 (2007), pp. 103–120.

[12] Cohen-Steiner, D. et al. “Lipschitz functions have L p-stable persistence”. *Foundations of Computational Mathematics* **10**.2 (2010), pp. 127–139.

[13] Csermely, P. et al. “Structure and dynamics of core/periphery networks”. *Journal of Complex Networks* **1**.2 (2013), pp. 93–123.

[14] Della Rossa, F. et al. “Profiling core-periphery network structure by random walkers”. *Scientific reports* **3** (2013).

[15] Dowker, C. “Homology groups of relations”. *Annals of mathematics* (1952), pp. 84–95.

[16] Edelsbrunner, H. et al. “Topological persistence and simplification”. *Discrete & Computational Geometry* **28** (2002), pp. 511–533.

[17] Edelsbrunner, H. *Weighted alpha shapes*. University of Illinois at Urbana-Champaign, Department of Computer Science, 1992.

[18] Fortunato, S. “Community detection in graphs”. *Physics Reports* **486**.3 (2010), pp. 75–174.

[19] Frosini, P. & Landi, C. “Size theory as a topological tool for computer vision”. *Pattern Recognition and Image Analysis* **9**.4 (1999), pp. 596–603.

[20] Funke, S. “Topological hole detection in wireless sensor networks and its applications”. *Proceedings of the 2005 joint workshop on Foundations of mobile computing*. ACM. 2005, pp. 44–53.

[21] Gamble, J. et al. “Adaptive tracking of representative cycles in regular and zigzag persistent homology”. Preprint: `arxiv:1411.5442`. 2014.

[22] Gamble, J. et al. “Coordinate-free quantification of coverage in dynamic sensor networks”. *Signal Processing* **114**.0 (2015), pp. 1 –18.

[23] Ghrist, R. “Barcodes: The persistent topology of data”. *Bulletin of the American Mathematical Society* **45** (2007), pp. 61–75.

[24] Hatcher, A. *Algebraic Topology*. 1st. Cambridge University Press, 2001.

[25] Hatcher, A. *Algebraic Topology*. Cambridge University Press, 2002.

[26] Holme, P. “Core-periphery organization of complex networks”. *Physical Review E* **72**.4 (2005), p. 046111.

[27] Howard, A. et al. “Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem”. *Proceedings of the 6th International Symposium on Distributed Autonomous Robotics Systems (DARS02)*. Citeseer. 2002, pp. 299–308.

[28] Huang, C.-F. & Tseng, Y.-C. “The coverage problem in a wireless sensor network”. *Mobile Networks and Applications* **10**.4 (2005), pp. 519–528.

[29] Khedr, A. M. et al. “Perimeter discovery in wireless sensor networks”. *Journal of Parallel and Distributed Computing* **69**.11 (2009), pp. 922–929.

[30] Kröller, A. et al. “Deterministic boundary recognition and topology extraction for large sensor networks”. *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. ACM. 2006, pp. 1000–1009.

[31] Le Boudec, J.-Y. & Vojnovic, M. “Perfect simulation and stationarity of a class of mobility models”. *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*. Vol. 4. 2005, 2743–2754 vol. 4.

[32] Leskovec, J. & Krevl, A. *SNAP Datasets: Stanford Large Network Dataset Collection* . `http://snap.stanford.edu/data`. 2014.

[33] Leskovec, J. & Sosič, R. *SNAP: A general purpose network analysis and graph mining library in C++*. `http://snap.stanford.edu/snap`. 2014.

[34] Leskovec, J. et al. “Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters”. *Internet Mathematics* **6**.1 (2009), pp. 29–123.

[35] Li, X. et al. “WLC12-1: Distributed coordinate-free hole detection and recovery”. *Global Telecommunications Conference, 2006. GLOBECOM’06. IEEE*. IEEE. 2006, pp. 1–5.

[36] Liu, B. et al. “Mobility improves coverage of sensor networks”. *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*. ACM. 2005, pp. 300–308.

[37] Liu, B. et al. “Dynamic Coverage of Mobile Sensor Networks”. *Parallel and Distributed Systems, IEEE Transactions on* **24**.2 (2013), pp. 301–311.

[38] Matoušek, J. “LC reductions yield isomorphic simplicial complexes”. *Contributions to Discrete Mathematics* **3**.2 (2008).

[39] Miller, E. & Sturmfels, B. *Combinatorial commutative algebra*. Vol. 227. Springer, 2005.

[40] Mischaikow, K. & Nanda, V. “Morse theory for filtrations and efficient computation of persistent homology”. *Discrete & Computational Geometry* **50**.2 (2013), pp. 330–353.

[41] Newman, M. E. “Fast algorithm for detecting community structure in networks”. *Physical review E* **69**.6 (2004), p. 066133.

[42] Newman, M. E. & Girvan, M. “Finding and evaluating community structure in networks”. *Physical review E* **69**.2 (2004), p. 026113.

[43] Peres, Y. et al. “Mobile geometric graphs: Detection, coverage and percolation”. *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2011, pp. 412–428.

[44] Poduri, S. & Sukhatme, G. S. “Constrained coverage for mobile sensor networks”. *Robotics and Automation, 2004. Proceedings. ICRA’04. 2004 IEEE International Conference on*. Vol. 1. IEEE. 2004, pp. 165–171.

[45] Robins, V. “Towards computing homology from finite approximations”. *Topology Proceedings*. Vol. 24. 1. 1999, pp. 503–532.

[46] Rombach, M. P. et al. “Core-periphery structure in networks”. *SIAM Journal on Applied mathematics* **74**.1 (2014), pp. 167–190.

[47] Silva, V. de & Ghrist, R. “Homological sensor networks”. *Notices of the American Mathematical Society* **54**.1 (2007), pp. 10–17.

[48] Silva, V. de & Ghrist, R. “Coordinate-free coverage in sensor networks with controlled boundaries via homology”. *The International Journal of Robotics Research* **25**.12 (2006), pp. 1205–1222.

[49] Singh, G. et al. “Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition.” *SPBG*. 2007, pp. 91–100.

[50] Wang, B. “Coverage problems in sensor networks: A survey”. *ACM Computing Surveys (CSUR)* **43**.4 (2011), p. 32.

[51] Wang, Y. et al. “Boundary recognition in sensor networks by topological methods”. *Proceedings of the 12th annual international conference on Mobile computing and networking*. ACM. 2006, pp. 122–133.

[52] Whitehead, J. H. C. “Simplicial spaces, nuclei and m-groups”. *Proceedings of the London mathematical society* **2**.1 (1939), pp. 243–327.

[53] Wilkerson, A. C. et al. “A distributed collapse of a network’s dimensionality”. *Proceedings of the IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE. 2013, pp. 595–598.

[54] Wilkerson, A. C. et al. “Simplifying the homology of networks via strong collapses”. *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE. 2013, pp. 5258–5262.

[55] Xie, J. et al. “Overlapping community detection in networks: The state-of-the-art and comparative study”. *ACM Computing Surveys (CSUR)* **45**.4 (2013), p. 43.

[56] Yang, J. & Leskovec, J. “Community-affiliation graph model for overlapping network community detection”. *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE. 2012, pp. 1170–1175.

[57] Yang, J. & Leskovec, J. “Defining and evaluating network communities based on ground-truth”. *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*. ACM. 2012, p. 3.

[58] Yang, J. & Leskovec, J. “Overlapping community detection at scale: a nonnegative matrix factorization approach”. *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM. 2013, pp. 587–596.

[59] Yang, J. & Leskovec, J. “Overlapping Communities Explain Core–Periphery Organization of Networks” (2014).

[60] Yick, J. et al. “Wireless sensor network survey”. *Computer networks* **52**.12 (2008), pp. 2292–2330.

[61] Zhang, X. et al. “Identification of core-periphery structure in networks”. *arXiv:1409.4813* (2014).

[62] Zomorodian, A. & Carlsson, G. “Computing persistent homology”. *Discrete & Computational Geometry* **33** (2005), pp. 249–274.