ABSTRACT

Title of dissertation: TOPOLOGICAL DATA ANALYSIS DIMENSION REDUCTION AND COMPUTATIONAL EFFICIENCY

Nathaniel Monson
Doctor of Philosophy, 2022

Dissertation directed by: Professor Wojciech Czaja
Professor Patrick Brosnan
Department of Mathematics

In this dissertation, we present a novel stability result for the persistent homology of the Rips complex associated to a point cloud. Our theorem is narrower than the classic result of Cohen-Steiner, Edelsbrunner, and Harer in that it does not apply to Čech complexes, nor to functions which are not measuring distance to a point cloud. It is broader than the classic result in that it is "local"; if a function approximately preserves distances in some range, but is contractionary below or expansionary above that range, our result still applies. The novel stability result is paired with the Johnson-Lindenstrauss Lemma to show that, with high probability, random projection approximately preserves persistent homology. An experimental analysis is given of the computational speedup granted by this dimension reduction. This is followed by some observations suggesting that even when the theoretical bound is loose enough that we have no guarantee of homology preservation, there is still a high chance that significant features of the dataset are preserved.

TOPOLOGICAL DATA ANALYSIS, DIMENSION REDUCTION, AND COMPUTATIONAL EFFICIENCY

by

Nathaniel Burton Monson

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2022

Advisory Committee:
Professor Wojciech Czaja, Chair/Advisor
Professor Patrick Brosnan, Co-Advisor
Professor Jonathan Rosenberg
Professor Christian Zickert
Professor Leila de Floriani (Dean's representative)

Dedication

To my grandmother, Nancy Moebus Heuston. Gran, I stand all amazed at the nigh-overwhelming support, generosity, and kindness you have offered me—oh, it is wonderful. I know you love me so much you would read this whole dissertation if I asked you to. And I love you so much that I won't ask.

Acknowledgments

I have been incredibly fortunate in my life to have wonderful teachers, friends, and family. As I write this, I am filled with more gratitude than I can express for all of them. This page is too small to contain every name—indeed, this entire dissertation is too small to contain the entirety of my thanks. Thank you, thank you all, from the bottom of my heart. I cannot thank you enough.

While a multitude of people have enriched my life, both mathematically and otherwise, there are six people without whom this dissertation would literally never have happened. In chronological order of entry into my life, I'd like to start by thanking my mother, Kary Patricia Heuston. Some of my earliest memories are of being in the lab, "helping" you as you worked on your own dissertation [86]. You were a model of spontaneous kindness and generosity, and determination in the face of overwhelming adversity. I love you and miss you.

My thanks also to my father, Roland Hintze Monson. You were largely responsible for beginning my love of math, by explaining everything from Cantor's diagonalization argument, to Alexander's horned sphere, to Fourier series to me, all before high school. You have been a pillar of love and support for me in almost every way I can imagine. If I ever have a child, I hope I am for that child what you've been for me. Thanks also to my stepmother, Dian Saderup Monson. While your influence on my mathematical life was limited to helping me learn my times tables in second grade, your presence in the rest of my life has been possibly the greatest blessing in a life filled with many.

Jumping forward several decades, my thanks to my advisors, Wojciech Klaudiusz Czaja and Patrick Gerald Brosnan. You have been patient when I required patience, and pushed me when I needed to be pushed. I am a better mathematician, better teacher, and even a better person due to your influence. You've been excellent mentors, and deserve much of the credit for any part of this dissertation that future readers find worthwhile.

Finally, my thanks to my friend Thomas Faraday Harper. In the marathon of my dissertation, you have done everything short of slinging me bodily over your shoulder and hauling me for twenty miles. Thank you for breakfasts, emotional support, ice water, evening planning, cakes, and the love which they all represent. Without your literally hundreds of hours of encouragement and logistical support, this dissertation would not exist.

My thanks also to my mathematical community. Thomas Hunter, Jonathan Rosenberg, Walter Stromquist, and Larry Washington have all been wonderful professors for me. I first learned about persistent homology from Nathan Dykas at an event kindly organized by Niranjan Ramachandran. In dealing with the Graduate School of University of Maryland, Tom Haines and Cristina Garcia took in what could have been a nightmare of red tape for me, and gave me smoother sailing than I could have hoped for. Steve Balady, Rebecca Black, JP Burelle, Jon Cohen, Ran Cui, Michael Kreisel, Adam Lizzi, Tim Mercure, Richard Rast, Catie Schwartz, and many others have gone from being peers and fellow grad students to friends.

I began this dissertation without any programming skills. Rose Lynn Embry, Liam Fowl, Josh Gleason, Ilya Kavalerov, Stephen Tratz, and Ben Warren all pro-

vided some combination of programming assistance and friendship, both of which were deeply appreciated.

My brother Boyd Monson spent many hours keeping me company while I worked on this, and provided food and encouragement. My sister Olivia Monson gave input on many important questions, such as "is Paul Bunyan a kaiju?" She is also the artist responsible for Figures 2.6 and 3.14. Both of their kindness and good cheer have been invaluble.

Christopher Green, we did for each other what we could. Maybe it was not what we have asked for, when we have spoken. But a man can only give what he has, being what he is. Thank you for breakfasts, dinners, board gaming, interesting conversations, and your relentless and uncompromising support. Carry on.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Summary of results

The primary results of this dissertation are twofold. First, we present a novel stability result for Rips complexes in Theorem 4.2.11. Our result differs from the classic result of Cohen-Steiner, Edelsbrunner, and Harer (Theorem 2.4.2) in several ways. Their result is more general in that it concerns the stability of the persistence diagram of any tame function on a triangulable space, while ours is restricted to the case of functions which are the distance to a finite point cloud, and ours only approximates the persistent homology using the Rips complex, while the Čech complex is the true object of interest. While this may sound a significant restriction, almost all practical applications we have seen in the literature are computing Rips complexes from finite point clouds. Indeed, we have only found a single computer implementation of an algorithm which can support computation of persistent homology via Čech complexes—all others default to using the Rips complex. Given that the sacrifice of Čech to Rips has already been made, our Theorem 4.2.11 is thus more directly applicable. Moreover, our result is an "element by element" result which can be applied locally in some sense, while their result only applies if there is a single uniform bound on the motion of the entire point cloud. We pair Theorem

4.2.11 with the Johnson-Lindenstrauss Lemma (Theorem 4.3.1) to show that, with high probability, random projection approximately preserves persistent homology.

In the second major contribution of this dissertation, we give an experimental analysis of the computational speedup granted by dimension reduction through random projection. We also give some observations suggesting that even when the theoretical bound is loose enough that we have no guarantee of homology preservation, there is still a high chance that significant features of the dataset are preserved.

1.2 Literature review

The subfield of computational topology known as persistent homology has grown quickly since its birth two decades ago. There are many survey articles on persistent homology. The author recommends Ghrist's "Barcodes: the persistent topology of data" [78] as an especially welcoming introduction to the subject. Very good alternatives include articles by Edelsbrunner and Harer [65], (and its successor [69]) Carlsson [34] or Weinberger [125]. While the field of applied topology, for which persistent homology is the exemplar, is too young to have many textbooks, those that exist [66, 79, 109, 116] are uniformly good.[1]

The paper that introduced persistent homology in its modern incarnation was "Topological persistence and simplification," by Edelsbrunner, Letscher, and Zomorodian [68]. This paper drew on the three associated geometric notions of

[1] An honorable mention should be made here of [129], which was before its time, and thus does not contain many important later results. A further honorable mention to Chambers et al.'s *Research in Computational Topology* [39]. While neither a textbook nor elementary, it collects many interesting papers springing from research conducted at a workshop in 2018.

Voronoi regions, Delauney triangulations, and alpha complexes to define the persistent homology of a filtered complex and introduce the index-persistence plane for visualizing their results, and also discussed a reasonably efficient algorithm for pairing simplices in order to calculate the persistent homology. It also included a proof-of-concept example of using persistent homology for feature detection, calculations of the persistent homology of five objects of interest (e.g., a certain protein and a small statue). This perspective was refined shortly thereafter in "Computing persistent homology," by Zomorodian and Carlsson [128]. An important contribution of their paper is the identification of persistent homology of a filtered complex with the standard homology of a certain graded module over a polynomial ring. This implies that persistent homology with field coefficients can be simply classified, in contrast to persistent homology with $\mathbb{Z}$ coefficients. The last piece of the core of persistent homology was built by Cohen-Steiner, Edelsbrunner, and Harer, in [45], which showed that persistence is stable under perturbations of the underlying function, a result which our Theorem 4.2.11 is closely tied to. This stability theorem of Cohen-Steiner, Edelsbrunner, and Harer (Theorem 2.4.2), was further refined by Cohen-Steiner et al in [46] and [47], and by Chazal et al in [40].

Even before the 2000 paper of Edelsbrunner, Letscher, and Zomorodian, several mathematicians were already converging on these ideas. In 1990 Frosini introduced "size functions" [76] which are equivalent to 0-dimensional persistent homology. Robins [115] described persistent Betti numbers in 1999. While Edelsbrunner et al independently discovered persistent homology the next year, and their language and formalism has persisted, the fundamental ideas are all present in Robins'

work. Also in 1999, Cagliari et al. reinterpreted and categorified size functions into a language which is quite close to the modern formulation of persistence in [30].

The insights and theory underlying persistent homology have been extended to a number of wider frameworks. In [38] Carlsson and Zomorodian explored multi-dimensional persistence, and prove that it is necessarily unsatisfactory. In [35, 36], Carlsson, De Silva, and Morozov give an explanation of and details for computing zigzag persistence for a chain of spaces connected by maps that are not simply inclusions. In [17] Bendich, Edelsbrunner, Morozov and Patel give a way to measure the robustness of a level set, and relate it to well groups and persistent homology. In [59] De Silva, Morozov, and Vejdemo-Johansson define "circular persistence" for S^1-valued functions. Bubenik categorifies persistent homology in [28], drawing on the work of [48] and [91]. He uses this approach to give a highly general stability result in [25]. Buchet, Chazal, Oudot, and Sheehy extend persistent homology from functions to measures in [29]. Basu, in [13], gives an explicit formula relating the persistent homology of a filtered space to the spectral sequence induced by that same filtration. And Dey, in [60], presents traditional persistent homology as dependent on a map to $\mathbb{R}$ as a topological space covered by intervals, and extends this formulation to a map whose codomain is any manifold with any cover.

There has been an explosion of interest in the last ten years on the applications of persistent homology across many fields. A full accounting is beyond the scope of this dissertation. Thus the following examples from medical research, image processing, and robotics should be taken as non-exhaustive representative samples of the broad research using computational topology.

Topological data analysis has seen considerable use in the study of medical data and genomic data, with [102] as a notable early example in which topological analysis identified a previously unknown subgroup of breast cancers clinically distinct from others. Neuronal structures in the visual cortex have been studied this way as well [121]. Topological analysis has also been used in image processing [1, 10] and robotic path planning [90].

In addition to the direct uses above, numerous papers have tried to bridge the gap between theory and practice—papers whose results are theoretical, but whose purpose is clearly to make applications more attractive or tractable. The following list highlights a few of these demonstrating the development of the field over time.

1. De Silva, Ghrist, and Muhammad have used persistent homology to demonstrate that a swarm of robots with no localization capabilities can determine if it covers an area [57].

2. Carlsson, Ishkhanov, De Silva, and Zomorodian studied the space of natural images, using persistent homology to identify a Klein bottle structure within it, and gave reason to think this has the potential to be useful for image compression [37].

3. Possibly most importantly on this list is the 2008 paper by Niyogi, Smale, and Weinberger "Finding the homology of submanifolds with high confidence from random samples" [103], a foundational paper in manifold learning, in which the authors demonstrate that random finite discrete samples from a manifold allow recovery of its homology with high confidence. Niyogi et al.

later extended their results to relate more explicitly to unsupervised machine learning [104].

4. Bubenik and Kim have related this statistical perspective of Niyogi et al. more directly to persistent homology [27].

5. Adler et al. discussed manifold learning, as well as applications of persistent homology to statistics and probability; they drew parallels between sublevel sets and excursion probabilities [2].

6. Walker has used persistent homology to establish coverage of sensor networks over a domain [124].

7. Fasy et al. have used statistics to give a detailed account of how much a given feature detected by persistent homology matters [74].

8. Sheehy has given an algorithm for a linear-size approximation of the Vietoris-Rips complex [119], and has paired the Johnson-Lindenstrauss Lemma (Theorem 4.3.1) with the stability theorem of Cohen-Steiner, Edelsbrunner, and Harer (Theorem 2.4.2) to give an algorithm for approximating persistent homology somewhat faster at the cost of some accuracy [120].[2]

9. Bubenik et al. have introduced persistence landscapes [26], a way to make persistent homology more useful for machine learning applications, while Adams

[2]Although Oudot, in [109], gives some reason to believe the speedup is small enough to be not worth it in most applications. Chapter 5 of this dissertation contains some experiments to evaluate these perspectives along with a discussion.

et. al refined this into the idea of persistence images [1], which have some desirable properties that persistence landscapes lack.

10. Finally, Mandal [94] has demonstrated the usage of sparsification methods to assist persistent homology calculations in computational biology.

In addition to the above results on the applications of persistent homology, there is an additional current of research flowing in the direction of increasing the computational speed of various aspects of persistent homology. There is not a sharp division between the papers listed here and those listed above, merely a change in emphasis; the papers below are more narrowly focused on limitations of computer speed and memory. Before giving this list, the author wishes to give special notice to the 2017 paper "A Roadmap for the Computation of Persistent Homology," by Otter, Porter, Tillmann, Grindrod, and Harrington [107]. An excellent paper which is highly accessible, it serves both as an inviting introduction to persistent homology as used in practice, and as a framework for understanding many other contributions and areas for further study. It also contains a variety of datasets suitable for benchmarking computational speeds of persistent homology calculations on, which the authors have made publicly available. This dissertation makes use of many of these datasets in Chapter 5. If the reader wishes to know more about computational topology, this is the first paper they should read.

- In 2004, Kaczynski, Mischaikow, and Mrozek published *Computational Homology* [88]. While this book does not deal with persistence directly, it addresses questions of speed of homology computations in great detail, and provides a

number of novel results.

- In 2013, Attali, Lieutier, and Salinas [9] give explicit conditions for the Rips complex of a point cloud to be collapsible to the same homotopy type as the underlying shape, and two years later, in [8], the first two authors give a (non-constructive) result that sheds light on why collapses tend to not get stuck in "house with two rooms"[3] situations.

- Also in 2013, Kerber and Sharathkumar [89] give a relatively quick algorithm which yields an approximation of the Čech filtration of any point cloud.

- Again in 2013 (a highly productive year for this area) Mischaikow and Nanda, in [98], adapt discrete Morse theory for filtrations and efficient computations of persistent homology. This perspective was built on by Curry, Ghrist, and Nanda in [50] for more efficient computations of sheaf cohomology, which includes persistent homology as a special case, and also by Henselman and Ghrist in [85] which observes a novel relationship between discrete Morse theory, matrix factorizations, and matroids.

- In 2020, Espinoza, Hernández-Amador, Hernández-Hernández, and Ramonetti-Valencia in [72] present an algorithm for computing the Čech complex of a planar collection of discs with different radii, and assess its performance.

- Finally, also in 2020, Malott, Sens, and Wilsey in [93] evaluate several data reduction methods, including random point sampling vs cluster centroids.

[3]The "house with two rooms" is an example of a simplicial complex which is contractible, but has many simplices and no available collapses.

The bridge connecting the idea of a stability result, such as our Chapter 4 to a speedup of computations, as we investigate in Chapter 5, is compressive sensing, a field of math concerned with reconstructing high dimensional information with high probability from low dimensional projections. The foundation on which all of compressed sensing rests is the Johnson-Lindenstrauss Lemma (Theorem 4.3.1), from their paper "Extensions of Lipschitz mappings into a Hilbert space" [87]. While the original proof is quite technical, Dasgupta and Gupta discovered an elementary proof in the appropriately-named "An elementary proof of a theorem of Johnson and Lindenstrauss" [54]. More recently, an explosion of interest has followed the papers "Near optimal signal recovery from random projections: Universal encoding strategies?" of Candes and Tao [32], closely followed by "Stable signal recovery from incomplete and inaccurate measurements" [33] and "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," [31] both by Candes, Romberg, and Tao. See Clark [11], Cloninger [44] Doster [62], Hafftka [81] or Murphy [101]for more recent examples of compressed sensing in action.

A recent example of using compressed sensing for a similar purpose to that we put it in this paper is given by Halevy, in [82], wherein he speeds the computation of Laplacian eigenmaps using random projection.

1.3 Dissertation organization

In Chapter 2, we present some mathematical preliminaries, primarily topological, with no original theorems.

In Chapter 3, we discuss applications of persistent homology to path planning, based largely on a paper by Ghrist and Battacharya. We include some information about experiments done with DTRA funding, which the author wrote some (ultimately unused) computer code (implementing the algorithm described by Ghrist and Battacharya) for.

In Chapter 4, we give a proof of a novel stability result for persistent homology, which yields the most commonly used application of the stability theorem of Cohen-Steiner, Edelsbrunner, and Harer (Theorem 2.4.2) as a special case. Following a suggestion by Baraniuk and Wakin [12], we pair it with the Johnson-Lindenstrauss Lemma (Theorem 4.3.1) to show that random projection preserves homology of point clouds with sufficiently high persistence, and has the potential to do better than that if the points lie on a manifold. We include some directions for future research.

In Chapter 5, we examine the implications of the dimension reductions of Chapter 4 for the speed of computing persistent homology, with an emphasis on computing the lower dimensional homology groups. We discuss the results of some experiments done with various software packages.

Chapter 2: Topological preliminaries

This Chapter is devoted to an exposition of the topological results that form the necessary background for the rest of this dissertation, in particular the field of persistent homology. First we introduce simplicial homology, then we relate it to point clouds (finite sets of points in Euclidean space). We then give a simplistic formulation of "naive" persistent homology, which will arise naturally from our discussions on point clouds. We then mention two extensions of this formulation—one that aids in the speed and ease of computation, and another that extends the definition of persistent homology to a much more general class of objects than finite point sets in $\mathbb{R}^d$. This language will allow us to quote the stability theorem of Cohen-Steiner, Edelsbrunner, and Harer. In Chapter 4 we will prove a theorem which, in the case of point clouds in $\mathbb{R}^d$, serves as a substantial generalization of their stability theorem.

While none of the theorems or definitions we give in this Chapter are original to this work, some of our exposition takes a perspective we have not seen before in the literature—in particular, the counterexample described in Figure 2.10 is one the author wishes he had seen.

2.1 Simplicial homology

Homology has an illustrious history; distant ancestors appear in 1758 [73], where Euler first discussed the eponymous Euler characteristic, as well as 1857 [114], where Riemann defined the genus of a surface and 1870 [18], when Betti first defined the invariants of a surface that would later come to be called Betti numbers . It was given the name "homology" in 1895 by Poincaré in [111]. The modern formulation of algebraic topology, the idea of a group of cycles modulo a group of boundaries, was first given by Noether, between 1925 and 1928.[1]

Our discussion here follows the standard background material; we recall some basic definitions and state some theorems for reference. For more details, any text which provides an introduction to algebraic topology will give a more thorough discussion than we do here, as well as contain more proofs. Hatcher's *Algebraic Topology* [83] has a very clear discussion, and has the further advantage of being available free online.[2] Other standard references include Armstrong [6], Bredon [24], and, for the brave who appreciate a minimalistic text with full generality, May [96].

We take the classical perspective of homology as measuring the shape of a space, in the sense of how many holes it has. The 0^{th} homology measures how many connected components an object has, the first measures how many non-trivial loops it has and so forth. An d-dimensional sphere has a hole in the d^{th} dimension (and

[1] She seems to have never written a paper focusing on this. However, in the summers of 1926 and 1927, she attended a series of lectures by Hopf and her discussions and comments there are agreed to have been original and insightful. Her viewpoints were adopted by Hopf and others shortly thereafter. See, e.g., [61] for more details of her contributions.

[2] It was also consulted frequently while writing this Section.

no other holes) so we intuitively expect homology to "notice" this, and it does—the homology of the d-sphere is trivial in all dimensions except the 0^{th} and the d^{th}, and is non-trivial there. In fact, it is rank 1 in each of those, noticing a single connected component and a single d-dimensional hole.

2.1.1 Simplicies and simplicial complexes

Our eventual goal is to perform large-scale homological computations. Of the several approaches to homology, that of simplicial complexes is most sympatico with this goal, so we forego the lure of Whitehead's CW-complexes approach [126] in favor of this more combinatorial language. Simplicial complexes were used by Eilenberg and Zilber in [71] to introduce simplicial sets, which have been used to great effect by Quillen [112], May [97], Bousfield and Kan [22], Segal [118], and many others. Standard references include Massey [95], Armstrong [6], and Spanier [122] (which we use here). Fomenko [75], while less well-known, covers similar material with a visual focus, and has some very useful illustrations. Following in their footsteps, we now define a simplex. The following definition is equivalent to that given in Spanier (page 108) [122].[3]

Definition 2.1.1 (Abstract n-simplex; faces)**.** An *abstract n-simplex*[4] is a set of $n+1$ elements, $[v_0, v_1, \ldots, v_n]$. These v_i's should be thought of and referred to as *points* or *vertices*. If simplex σ_1 is a subset of simplex σ_2, then we say σ_1 is a *face* of σ_2. If the set of vertices is equipped with an order, we call it an *oriented* n-simplex.

[3] Although he does not refer to them as abstract.
[4] In some references these are called *combinatorial* n-simplices.

We consider two oriented simplices equivalent if the orders of their vertices differ by an even permutation. If σ is an oriented simplex, we write $\bar{\sigma}$ for the oriented simplex with the same vertices as σ, with an opposite orientation.

Definition 2.1.2 (Abstract simplicial complex)**.** An *abstract simplicial complex* is a collection, Δ, of abstract simplices with the property that any face of a simplex in Δ is itself also in Δ.

Importantly, we do not insist that the vertices of the simplices which compose the simplicial complex be distinct. (Indeed, if we did it would be impossible to have any simplex with more than one vertex in the simplicial complex; its faces are, by definition, not disjoint from it). For instance,

$$\{\{1,2,3\},\{1,2\},\{2,3\},\{1,3\},\{2,4\},\{1,4\},\{1\},\{2\},\{3\},\{4\},\varnothing\}$$

is an abstract simplicial complex—it is the smallest simplicial complex containing the simplices $\{1,2,3\},\{1,4\},$ and $\{2,4\}$. Equivalently, it is the closure of $\{\{1,2,3\}$, $\{1,4\}$, $\{2,4\}\}$ under the operation of taking subsets.

These abstract objects are purely combinatorial but we are ultimately interested in working with topological spaces such as subsets of $\mathbb{R}^d$. To connect the two, we make the following definitions:

Definition 2.1.3 (Standard n-simplex; realizations of n-simplices)**.**

a) The *standard n-simplex* is a subset of $\mathbb{R}^{n+1}$; it is the convex hull of $\{\vec{e}_1,\ldots,\vec{e}_{n+1}\}$, the standard unit vectors. These are, of course, the vertices of the simplex.

b) A *realization* of an abstract n-simplex is a subset of $\mathbb{R}^d$ which is the image of the standard n-simplex under an affine map; the realization is considered degenerate if it is not homeomorphic.[5]

In geometrically realizing simplicial complexes, just as in the above case of simplices, we shall define a standard realization and then generalize it with appropriate homeomorphisms.

Definition 2.1.4 (Realizations of simplicial complexes)**.**

a) If Δ is an abstract simplicial complex with n points, a *standard realization* of Δ is a subset D of $\mathbb{R}^n$. It is uniquely determined by a bijection between the points of Δ and the standard unit vectors $\vec{e}_1, \cdots, \vec{e}_n$ in $\mathbb{R}^n$. Given such a bijection, a point x of $\mathbb{R}^n$ is in D if and only if it is in the convex hull of a set of $\vec{e}_i$'s whose corresponding points form a simplex in Δ; i.e., each simplex of Δ is realized in a way consistent with the bijection.

b) More generally, if D is a standard realization of a simplicial complex Δ then $K \subset \mathbb{R}^k$ is a (non-degenerate) *realization* if there exists a homeomorphism $f : D \to K$ such that the restriction of f to the portion of D corresponding to any simplex of Δ is an affine map.

From here on, when we refer to a realization, we mean a non-degenerate realization unless otherwise specified. See Figure 2.1 for a simple example of a realization.

[5]The standard n-simplex is, of course, a (non-degenerate) realization of an abstract n-simplex via the identity map.

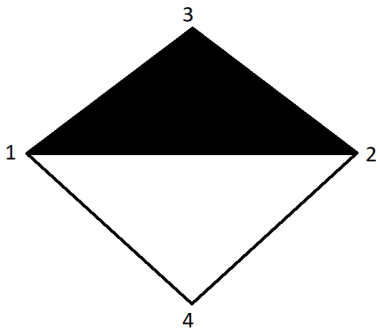


Figure 2.1: A geometric realization of the simplicial complex generated by $\{\{1,2,3\},\{1,4\},\{2,4\}\}$.

Simplicial complexes are quite easy to work with, and spaces that are sufficiently similar to them deserve a name.

Definition 2.1.5 (Triangulable space)**.** A space is called *triangulable* if it is homeomorphic to the realization of some finite simplicial complex.

In particular, a single simplex is triangulable. As the n-sphere, S^n, is homeomorphic to the union of faces of an n-simplex, the n-sphere is triangulable.

2.1.2 Chain complexes

We now take these geometric, combinatorial objects and use them to generate algebraic structures through the following definitions. The following definition is equivalent to that given in Spanier (page 159) [122].

Definition 2.1.6 (Chain groups; chain complexes)**.**

a) Given a simplicial complex, Δ, the associated *chain groups* are, for every natural number i, an abelian group C_i which is the set of formal sums (with integer

coefficients unless otherwise specified) of oriented i-simplices of Δ, subject to the relation $\sigma = -\bar{\sigma}$. There is a map, $\partial_n : C_n \to C_{n-1}$, called the boundary map. On the level of an individual oriented simplex, $\sigma = \{v_0, v_1, \ldots, v_n\}$, we define $\partial_n(\sigma) = \sum_{i=0}^{n}(-1)^i\{v_0, \ldots, \hat{v_i}, \ldots, v_n\}$ (where the hat means exclude the corresponding vertex), and we define ∂ of a formal sum of simplices to be the formal sum of ∂ of the individual simplices.

b) Piecing these together, we have the *chain complex*, which is the sequence of abelian groups

$$\cdots \to C_{n+1} \xrightarrow{\partial_{n+1}} C_n \xrightarrow{\partial_n} C_{n-1} \to \cdots \to C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0,$$

which has the property that $\partial_n(\partial_{n+1}) = 0$ for all n—that is to say, the image of ∂_{n+1} lies within the kernel of ∂_n. In symbols, Im $\partial_{n+1} \lhd$ Ker ∂_n.

We are now in a position to define the homology groups of a chain complex, and by extension, the homology groups of a simplicial complex.

Definition 2.1.7 (Homology groups)**.** The n^{th} *homology group of a chain complex* X *with coefficients in* $\mathbb{Z}$, usually written $\mathrm{H}_n(X, \mathbb{Z})$ is Ker ∂_n/Im ∂_{n+1}.[6] We will call the n-index the dimension of the homology group. If, in the construction of the chain complex, we take formal sums with coefficients in an abelian group G other than the integers, we write the result as $\mathrm{H}_n(X, G)$.

We will frequently take homology with coefficients in a field, in which case the

[6]Note that these are both subgroups of C_n.

chain complex will be a sequence of vector spaces connected by linear maps. When it will not hamper clarity to do so, we will omit the coefficients and write only $\mathrm{H}(X)$. We will use $\mathrm{H}_*(X)$ to refer to the direct sum of the homology over all dimensions if we wish to emphasize the direct sum; otherwise, we may omit the subscript entirely. If X is any object with a canonical chain complex associated to it (e.g., a simplicial complex), we will write $\mathrm{H}(X)$ for the homology of the chain complex associated to X, rather than add unnecessary clutter by creating a separate symbol for the chain complex itself.

Homology has a variety of very useful properties. It is a homotopy invariant: any two simplicial complexes which are homotopy equivalent to each other have the same homology groups. Because of this, we extend our definition of homology to include any space which is homotopy equivalent to a simplicial complex. Most importantly for our purposes, homology is a functor—that is, a map between spaces induces a family of maps between the corresponding homology groups. The case that will be of most interest to us will be the fact that if $U \subset X$, the inclusion map $i : U \hookrightarrow X$ will induce a map on homology, $i_* : \mathrm{H}(U) \to \mathrm{H}(X)$. Note that this induced map is not guaranteed to be an inclusion. Indeed, it will usually not be; by a theorem of Whitney [127] any smooth manifold can be embedded into $\mathbb{R}^d$ for a sufficiently high d, which has trivial homology above dimension 0, but most smooth manifolds have non-trivial higher homology groups.

2.2 Point clouds

Homology is the tool of our study, and real-world datasets are its object. In particular, we would like to study the homology of spaces derived from finite sets of points inside $\mathbb{R}^d$, as many datasets have a somewhat natural interpretation into this setting.

Definition 2.2.1 (Point cloud)**.** A *point cloud in dimension d* is a finite subset of $\mathbb{R}^d$. We will denote a point cloud by $X = \{x_1, x_2, ..., x_n\}$. We denote the open ball centered at x_i of radius r by $B_r(x_i)$. We use X_r to denote the union of open balls of radius r around each point in X.

We will frequently think of and refer to r as a time parameter—imagining the balls expanding over time. Figure 2.2 shows an example of a simple point cloud and two of its expansions.

In order to study point clouds with the tool of homology, we must find a way to view them as simplicial complexes. To that end, we make the following two definitions (see Croom [49] for more details):

Definition 2.2.2 (Good cover)**.** A *good cover* of a topological space is a collection of open sets (whose union is the whole space) such that each open set, each pairwise intersection, each 3-way intersection, etc. are all contractible.

For any r, the space X_r has a canonical good cover $\{B_r(x_i)\}$. This cover is good because balls are convex, intersections of convex sets are convex, and convex sets are contractible.

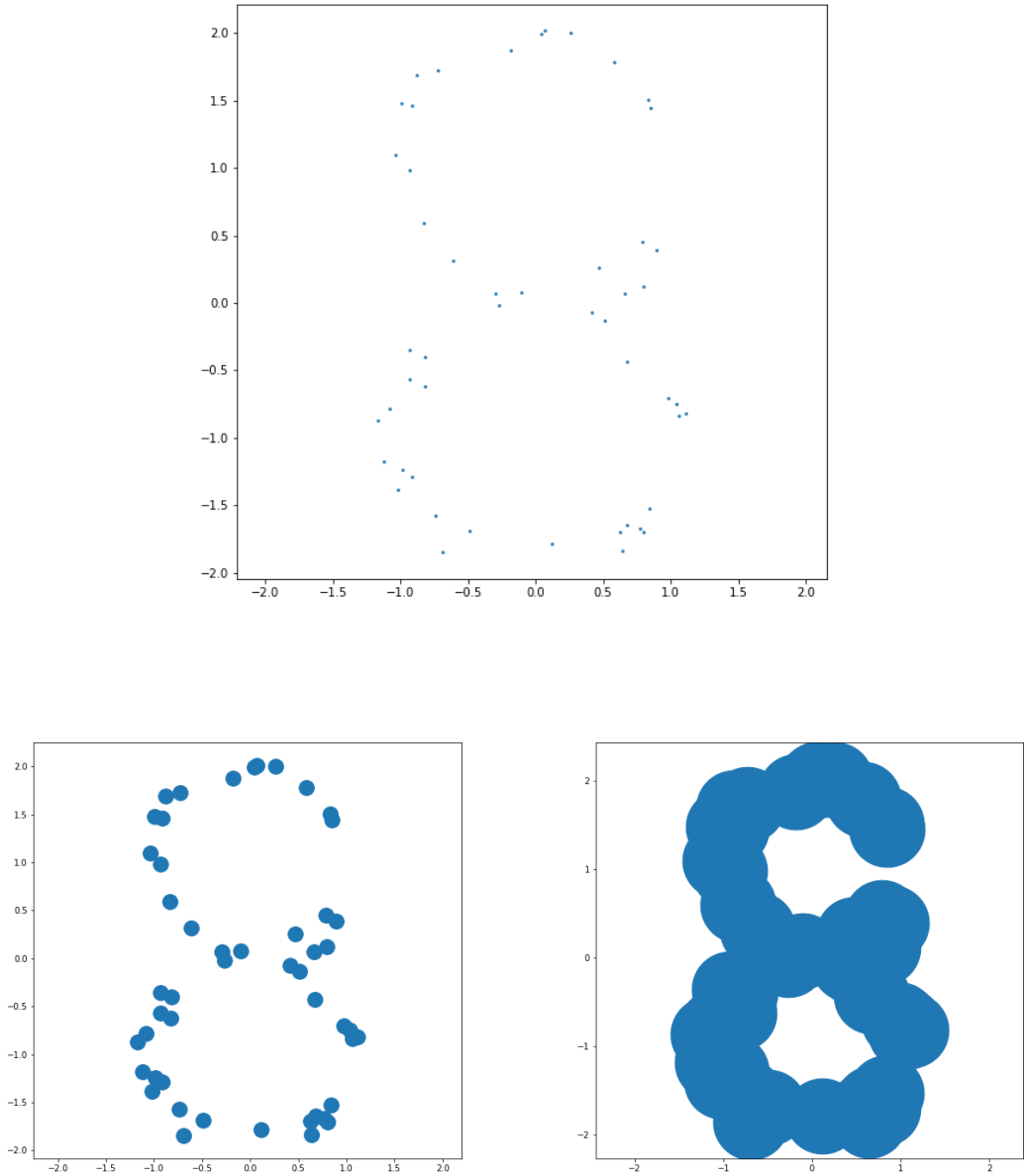


Figure 2.2: An example of a 2-dimensional point cloud, X, (top) along with two of its expansions, $X_{.1}$ (bottom left) and $X_{.5}$ (bottom right).

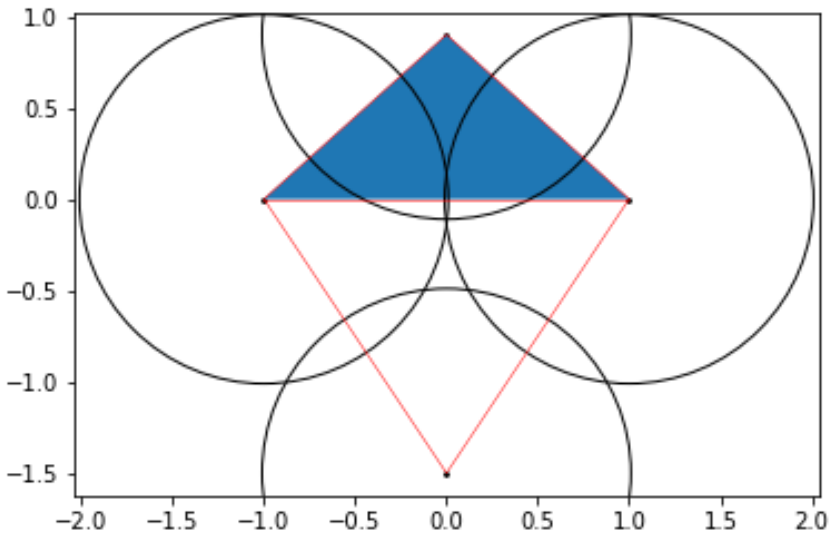


Figure 2.3: A Čech complex at scale 1.01, generated by four points in the plane.

Definition 2.2.3 (Čech complex)**.** Given a topological space and an open cover of it, the *Čech complex* (also called the nerve) of the cover is the simplicial complex formed by a 0-simplex for each element of the cover, and an n-simplex for each n-way intersection in the obvious way. We will use $C(X_r)$ to refer to the Čech complex associated with the canonical good cover of X_r.

An example Čech complex generated from 4 points is shown in Figure 2.3.

We will occasionally refer to the Čech complex associated to X_∞—this is the maximal Čech complex, given by a simplex for every subset of the vertices. As a topological space, it is contractible.

Importantly, the Nerve Theorem is true:

Theorem 2.2.4 (Nerve Theorem, Borsuk [21])**.** *Given a subset of* $\mathbb{R}^d$*,* X*, and a*

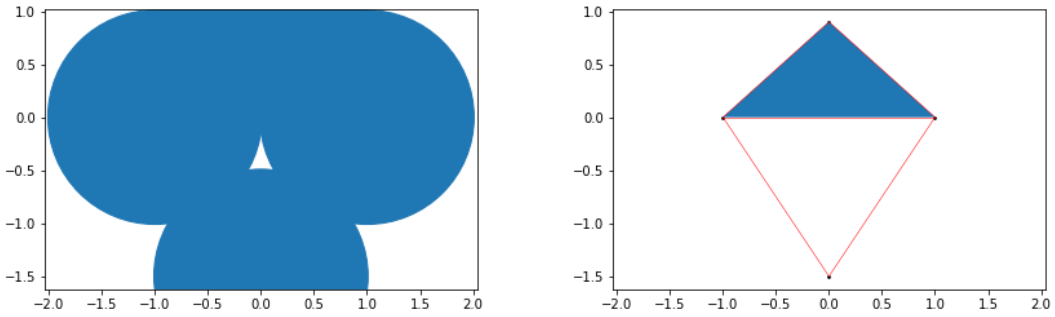


Figure 2.4: Here we see two interpretations of Figure 2.3. By the Nerve Theorem (Theorem 2.2.4), the topological space $X_{1.01}$ (top) and the Čech complex generated by its natural good cover (bottom) are homotopy-equivalent. In this case, both are equivalent to S^1, the one-dimensional circle.

good cover $\{U_i\}$ *of* X*, the Čech complex of* $\{U_i\}$ *is homotopy equivalent to* X*.*[7]

We can geometrically realize the nerve of X_r with X, and a straight line between each pair of points of X within $2r$ of each other, triangles filling in appropriate triples of lines, etc. (See Figure 2.4).

Note that if r_1 and r_2 are positive real numbers with $r_1 < r_2$, there is a natural inclusion map $i : X_{r_1} \hookrightarrow X_{r_2}$. By functoriality, this induces a map on homology, $i_* : \mathrm{H}(X_{r_1}) \to \mathrm{H}(X_{r_2})$. Indeed, the parameterized family of spaces, X_r, yields a parameterized family of simplicial complexes, each homotopy equivalent to the corresponding space, which in turn yields a parameterized family of homology groups. This parameterized family might be constant for some long range of r's. We thus introduce the following definition:

Definition 2.2.5 (Stable homology)**.** [8] Let X be a point cloud. If, for all s, t with

[7] This is in fact true for most topological spaces one might run into in practice, although there are some pathological counterexamples within point-set topology that prevent us from asserting it for all topological spaces.

[8] We note there is a different sense of this phrase in the literature which usually applies to the

$r_1 < s < t < r_2$, the inclusion $i : X_s \to X_t$ is a homotopy equivalence, we say X has *stable homology* from r_1 to r_2.

Point clouds have only a finite number of radii at which the homology might change. If $X = \{x_1, x_2, \cdots x_n\}$ is a point cloud, then for any $I \subset \{1, 2, \cdots, n\}$ define

$$d_I := \inf\{r \in \mathbb{R}^+ | \bigcap_{i \in I} B_r(x_i) \neq \varnothing\}.$$

Thus, d_I is the smallest distance for which there exists a point (of our ambient space) which is within d_I of every point of the subset of X indexed by I. We can order all the d_{I_j}'s:

$$0 \leq d_{I_1} \leq d_{I_2} \leq \cdots \leq d_{I_{2^n}}.$$

X will have stable homology from d_{I_i} to $d_{I_{i+1}}$. The Čech complex generated at each of the radii between consecutive d_I's will, in fact, be identical. X may have stable homology for a longer range than that, but the Čech complexes will not be identical—see Figure 2.5.

The values where the homology changes are worth naming. It is easiest to define the intervals where homology does not change and then name their complements. For reasons that will become apparent later, we focus on the function which gives the distance from any point to our point cloud.

Definition 2.2.6 (Homological regular value; homological critical value)**.** [9]

homology of groups rather than spaces. In this dissertation we shall never take the homology of a group nor use this sense of 'stable,' so this should result in no confusion.

[9]The definition which follows is due to Bubenik and Scott. An earlier definition for the term "homological critical value" was due to Cohen-Steiner, Edelsbrunner, and Harer [45]. However,

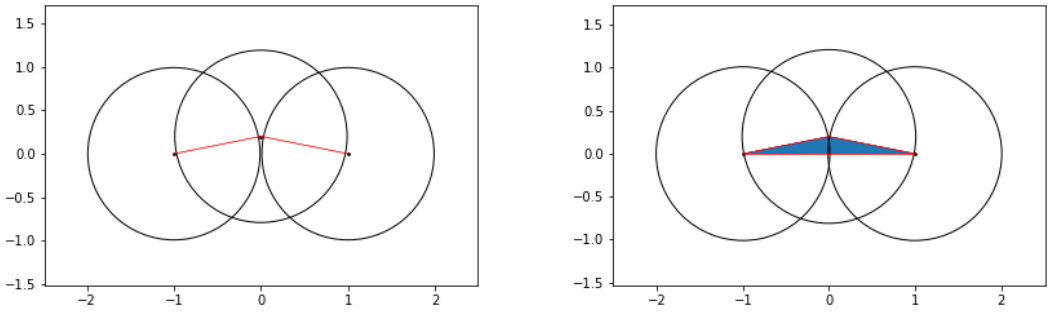


Figure 2.5: A point cloud X and the Čech complexes induced by the natural good covers of $X_{.99}$ (left) and $X_{1.01}$ (right). While the Čech complex changes, the homology does not. In both cases, the induced complex is contractible.

a) A real number a is a *homological regular value* of a function $f : \mathbb{R}^d \to \mathbb{R}$ if there exists an open interval I which contains a such that, for every r_1, r_2 with $[r_1, r_2] \subset I$, the map induced by inclusion $i_* : \mathrm{H}(f^{-1}[-\infty, r_1]) \to \mathrm{H}(f^{-1}[-\infty, r_2])$ is an isomorphism.

b) Any value of f which is not a homological regular value we will call a *homological critical value*. We will say r is a homological critical value of a point cloud X if it is a homological critical value of the function which measures distance to the nearest point of X.

To restate what we said earlier with our new language, all of the homological critical values of a point cloud X occur as one of the d_I's. The converse is not true—there exists d_I's which are not homological critical values.

We can use this idea to translate the time progression, which is continuous, to

in [28], Bubenik and Scott give a counterexample (which is further explored by Govc in [80]) to a theorem in Cohen-Steiner et al.'s paper, and suggest a minor correction to the definition under which the original theorem holds. In the case we are concerned with, that of persistent homology of finite point clouds, the two definitions are equivalent.

a discrete structure:

Definition 2.2.7 (Filtration)**.** Given a simplicial complex, Δ, an expanding nested sequence of subcomplexes is called a *filtration* of Δ. Thus we may write

$$\Delta_0 \subset \Delta_1 \subset \cdots \subset \Delta_{n-1} \subset \Delta_n = \Delta$$

for a filtration of Δ.

A few things to note about the above definition. The containments are not necessarily proper—Δ_i could well be equal to Δ_{i-1} or to Δ_{i+1}, or to both. A filtration may begin with the empty set, or indeed with several copies of the empty set. If $\Gamma \subset \Delta$, the above filtration of Δ induces a filtration of Γ:

$$(\Delta_0 \cap \Gamma) \subset (\Delta_1 \cap \Gamma) \subset \cdots \subset (\Delta_{n-1} \cap \Gamma) \subset (\Delta_n \cap \Gamma) = \Gamma.$$

Any point cloud X has a natural filtration of the fully connected Čech complex associated to X_∞, given by taking a Čech complex between each homological critical value.

It is reasonable to think of the parameter describing which step of the filtration we are referring to as a time parameter, and we shall do so. This is compatible with the notion of the subscript of X_t as a time parameter, in the sense that they both occur in the same order. It is, however, not identical to it as the filtration time parameter is always a positive integer, while the radius time parameter could be any positive real. We will be explicit in pointing out when we switch from one to

the other to avoid confusion.

2.3 Naive persistent homology

Let us review for a moment what we have done. As this list of steps will serve as a template for further refinements, we include some passive steps.

1. We began with a point cloud, which we left alone but will discuss modifying in Chapter 5.

2. From this point cloud we obtained a filtration of a simplicial complex.

3. We then had a parameterized family of simplicial complexes. Again, we will discuss modifying this in Chapter 5.

4. By taking the homology, we obtained a parameterized family of homology groups.

5. We then had a parameterized family of homology groups, which we will want to display in some easily analyzable format.

Why might we have done this? Frequently, we are in a situation where we have a point cloud that represents a sample of some important data which we have good reason to believe all lies on some manifold or algebraic variety (or finite union thereof). Ideally, we would be able to derive knowledge of the exact manifold from the point cloud, but this is obviously impossible—there are an infinite number of C^∞ manifolds which contain any given finite collection of points. But as long as the

embedding of the manifold is not too sharply angled, and our point cloud represents a sufficiently large and evenly distributed sample of points, we can still expect to calculate the major features of the manifold with high probability. In particular, if X has stable homology from a to b, with $a << b$, we might well conclude that X "really looks like" $X_{(b-a)/2}$. There are a number of difficulties with this. One is that any point cloud will have stable homology from a to ∞ for any a greater than the diameter of the point cloud—specifically, the X_r's will be contractible. Thus, we can't just choose the largest interval with stable homology—the homology on that interval is trivial.

Another difficulty is a bit more subtle. Consider a subset of the cylinder $x^2 + y^2 = 4, 0 \leq z \leq 2$ consisting of finitely many vertically oriented circles with radii between .1 and 1, each tangent to at least one circle whose center is clockwise of it and one whose center is counterclockwise.[10] Now let X be a random sample of many points from this space (See Figure 2.6). X_r is contractible for $r > 2$, and X has stable homology from 1 to 2 (and is homotopy equivalent to a circle in that range). But between 0 and 1, the homology of X can fluctuate wildly. Despite this, there should be an element of the homology of X_ϵ whose representative is also a non-zero element of $\mathrm{H}(X_{1.5})$, even for very small ϵ. A *piece* of the first homology is quite stable, even if the first homology group as a whole is not.

Both of the problems can be dealt with by focusing on the generators of homology instead of the entire collection of groups. This method for trying to extract these individual features of the structure underlying the data is called *persistent*

[10] Geometrically, these will be slight deformations of circles, due to the curvature of the cylinder.

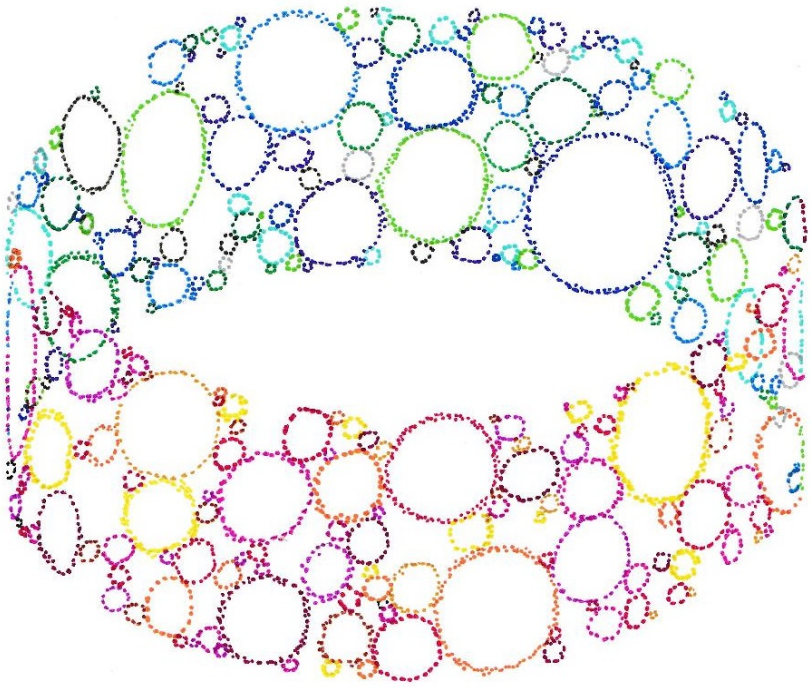


Figure 2.6: An image of the point set described in the text. A point cloud with a generator of homology more stable than its homology groups.

homology.

2.3.1 Birth-death matching

Suppose we have a parameterized family of homology groups, each mapping into the next. If we are given an element h of the homology at time t, $h \in \mathrm{H}^t(X)$, the length of time it will survive is well defined—its image under the time progression map, $\mathrm{H}^t(X) \to \mathrm{H}^{t+s}(X)$ will either be zero or not. Its "age," the amount of time it existed before time t is, however, not well defined. If the preimage under the time progression map is empty, then it clearly didn't exist earlier, and if the preimage is unique it clearly did, but what about a preimage with multiple elements?

Unfortunately, if we are working with coefficients in $\mathbb{Z}$, things are pretty dire for reasons we will return to. If, however, we are working with coefficients in a field F, the situation is much more tractable.

We mentioned earlier that we will be explicit when switching from using "time" to refer to the real-valued, radius-like parameter to using it instead for the discrete, integer-valued filtration parameter. Here is such a place. In the two paragraphs above, we intended time to refer to the former, and below, we will use it for the latter. Given an n-stage filtration of a d-dimensional simplicial complex, for which we use upper indices to avoid confusion with the lower indices indicating dimension, $\Delta^0 \hookrightarrow \Delta^1 \hookrightarrow \cdots \hookrightarrow \Delta^n$, the associated chain complexes have the structure shown in Figure 2.7.

Consider the direct sum of all the homology groups that result with coefficients

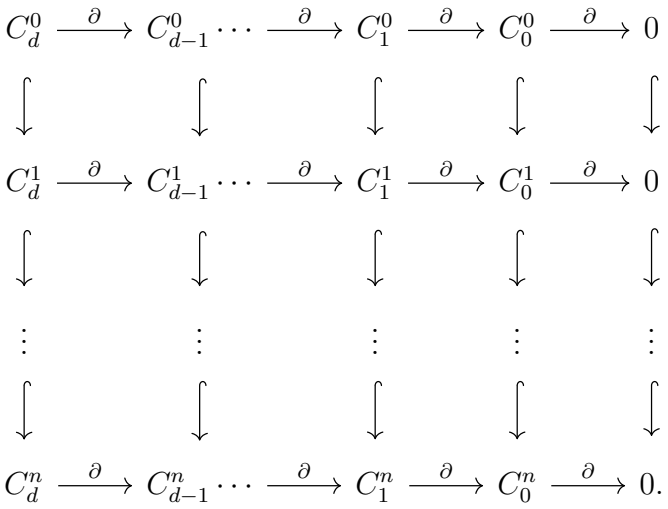


Figure 2.7: Here we show a diagram of the chain groups associated to an n-stage filtration of a d-dimensional simplicial complex. By the nature of a filtration, all the vertical maps are simple inclusions.

in a field F. The homology groups are vector spaces over F. The diagram above commutes, as all the vertical maps are simple inclusions, and the ∂ maps do not change due to these inclusions. Thus the vertical "time progression" maps are well-defined on homology.

This direct sum, together with the time-progression maps, thus contains all the information of all the homology groups at every time. We therefore name it.

Definition 2.3.1 (Persistence module)**.** Given a filtration of a simplicial complex and a field F, the *persistence module* associated to the filtration is an $F[x]$ module. It is the direct sum of the homology groups with coefficients in F of all the spaces in the filtration, with the usual F-action of a vector space. The action of x is moving to the next step in the filtration.

As F is a field, $F[x]$ is an integral domain in which every ideal can be generated by a single element. Such rings are called principal ideal domains, or PIDs. It would take us too far afield to go deeply into the theory of PIDs, but suffice it to say there is a well-understood theory of how to classify finitely generated modules over a PID.[11] There is essentially only one way of decomposing this module—thus, there is a preferred basis for the module. This preferred basis gives us a canonical pairing of birth/death times for each generator of the homology. This is precisely why the situation is worse with integer coordinates, as we mentioned earlier—the ring $\mathbb{Z}$[x] is not a PID, so the modules over it don't break down in the same easily classifiable way. Attempts to develop a satisfying theory of multiparameter persistent homology have encountered difficulties for a similar reason: $\mathbb{F}[x, y]$ is also not a PID.

In our much more easily classifiable case, we have the following:

Theorem 2.3.2 (Carlsson and Zomorodian [128])**.** *A finitely generated persistence module over a field, F, will decompose as*

$$(\bigoplus_i x^{t_i} \cdot F[x]) \oplus (\bigoplus_j x^{r_j} \cdot (F[x]/(x^{s_j} \cdot F[x]))),$$

where the i-indexed terms, which are free $F[x]$-modules, represent pieces of the persistence module that begin to exist at the t_i^{th} time step, and never stop existing. The j-indexed terms are those which begin to exist at time r_j and which then stop existing at time $r_j + s_j$.

[11] See, for instance, Chapter 12 of Dummit and Foote [63], or any other first-year text on graduate algebra.

Note that, as our point clouds are finite sets, all our persistence modules automatically satisfy the finitely generated condition. Earlier we mentioned a preferred basis. Note that if not all t_i or r_j, s_j pairs are unique, the preferred basis will not be either. In this case, we may pick any of the preferred bases, and consider its elements as the "generators" we mentioned earlier.

After taking advantage of the discrete structure of the time steps, we can now translate our integer-valued, discrete, ordinal times back into continuous, real valued times. A note of caution: there is some inconsistency in the literature and in the use of computer programs, with some sources translating the discrete times into radius-like times, and others translating them into diameter-like times, which is a natural thing to do when using the Rips complex, discussed later in this Chapter. For the remainder of this dissertation, **we shall use diameter-like times**, as that seems to be most consistent with the authors and software packages. As a result, the following definition is not uniform throughout the literature.

Definition 2.3.3 (Birth time, death time)**.** A generator of the persistence module of X which first appears as an element of $\mathrm{H}(X_{t_b/2})$, and has an image which is first 0 in $\mathrm{H}(X_{t_d/2})$ is said to have *birth time* t_b and *death time* t_d.

The generators of homology now have well-defined times at which we can consider them to have died or been born. We can graphically represent this information in a few ways. The two most common are a persistence diagram or a barcode.

Definition 2.3.4 (Persistence diagram)**.** A *persistence diagram for dimension* d is

a subset[12] of $\mathbb{R}^2$ which consists of a point for every generator of the d-dimensional homology. The x-coordinate of the point represents the t value at which the homology generator was born, and the y-coordinate represents the t value at which it dies. It is convenient to count the line $x = y$ as also being part of the diagram.[13]

Note that all points in a persistence diagram lie above the line $x = y$, as everything must be born before it dies. While we may occasionally refer merely to "a persistence diagram," omitting the identification of dimension, it is important to remember that a persistence diagram contains information about only a single dimension. Figure 2.8 displays a point cloud and the associated persistence diagrams for dimensions 0 and 1.

Another way of presenting the information contained in a persistence diagram is the barcode: a multiset of intervals, one interval for each generator of homology, with endpoints representing the birth and death times. Just as in the case of persistence diagrams, we consider the barcodes for each dimension to be seperate objects. Figure 2.9 shows a point cloud and associated bar code.

How can we compare two persistence diagrams? One method is to look at all possible bijections between the points of the two, and find the bijection which moves the most-moved point the least. The distance this least-moved point moves is called the *bottleneck distance*.

Definition 2.3.5 (Bottleneck distance)**.** Given two persistence diagrams, P_1 and

[12] Technically a multisubset, as we allow points to have multiplicity greater than one.
[13] And indeed to consider all points on it as having infinite multiplicity.

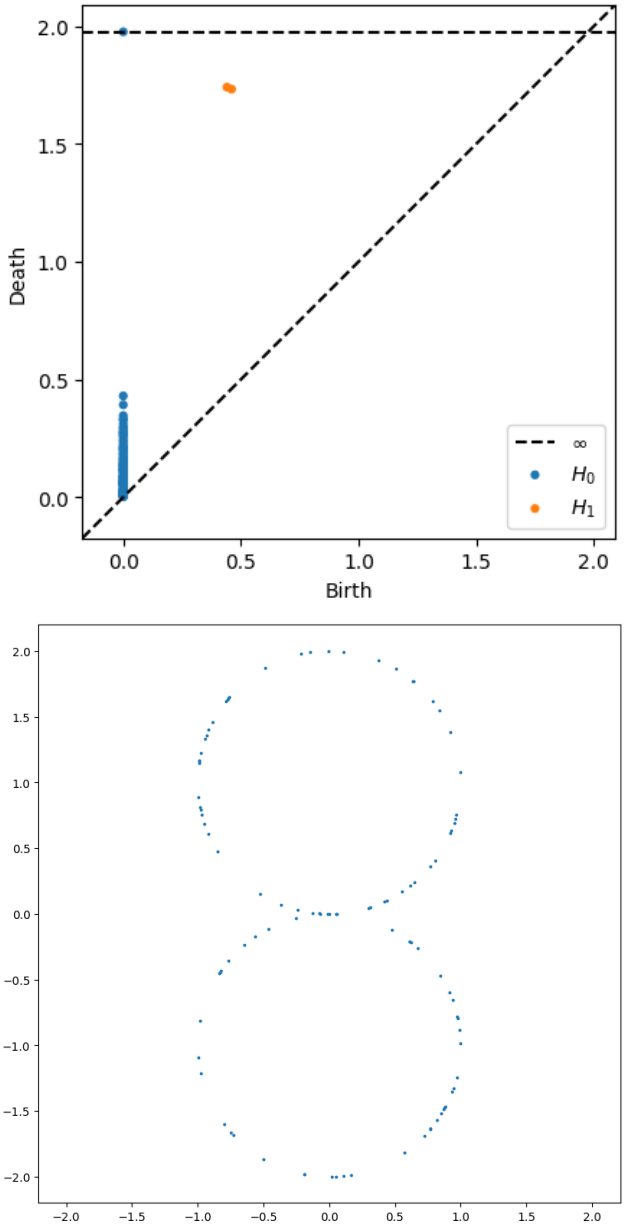


Figure 2.8: Top, we have the persistence diagrams for dimension 0 and 1, which display the persistent homology of a point cloud. As with any point cloud, the dimension 0 homology, shown here as blue points, is all born at time 0, and has a single point which never dies (displayed as a death time of 2 here, when the computer stopped sampling). The two high off-diagonal points of H_1 are suggestive of a point cloud with 2 large loops. Bottom shows the point cloud, which consists of 100 points picked at random from a figure-8, with 2 large loops as predicted.

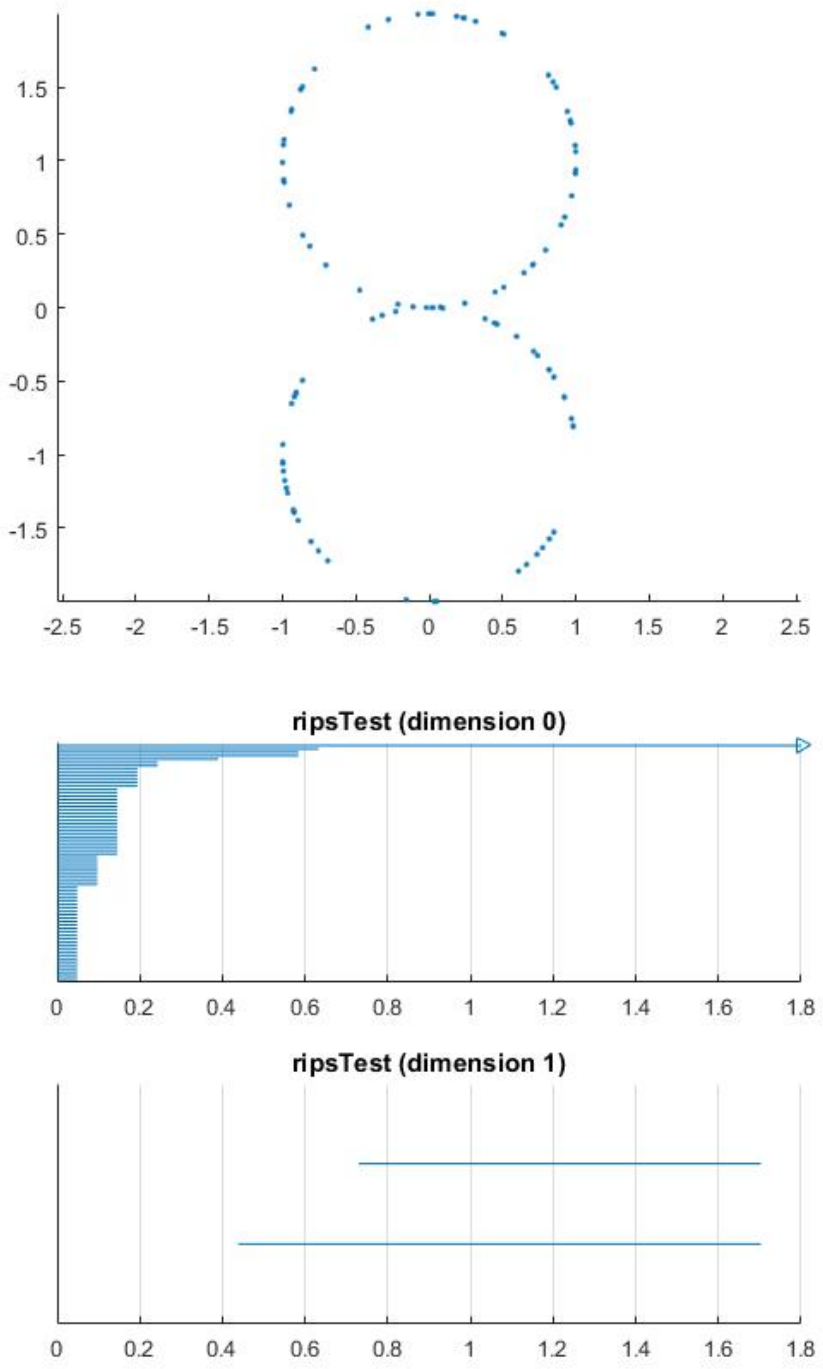


Figure 2.9: Top shows a cloud of 100 points picked at random from a figure-8. Bottom is the bar code displaying the persistent homology (calculated with 37 divisions) of the point cloud. The two long bars in dimension 1 correspond to the two circles of the figure-8.

P_2, the *bottleneck distance* (denoted D_B) between them is

$$D_B(P_1, P_2) := \inf_{\psi} \sup_{x} \|x - \psi(x)\|_\infty,$$

where ψ ranges over all bijections between P_1 and P_2, and x ranges over all points in P_1.

As calculating the bottleneck distance involves quantifying over all possible bijections, it can be quite unwieldy. Thus we make another definition:

Definition 2.3.6 (Hausdorff distance)**.** Given two persistence diagrams, P_1 and P_2, the *Hausdorff distance* (denoted $D_H(P_1, P_2)$) between them is

$$D_H(P_1, P_2) := \max\{\sup_{x} \inf_{y} \|x - y\|_\infty, \sup_{y} \inf_{x} \|y - x\|_\infty\},$$

where x ranges over all points in P_1 and y ranges over all points in P_2.

Note that the Hausdorff distance can never be greater than the bottleneck distance; one of the options the Hausdorff distance is taking the infimum over is the matched point $\psi(x)$ in the bottleneck distance.

It may aid clarity to translate these distance considerations back into the language of barcodes. If we are attempting to compare two barcodes, one method is to put every bar of the first barcode close to whichever bar of the second barcode it most resembles, (in the sense of endpoint-location), and then find the worst match (the element of the first barcode whose left endpoint is farthest from its matches left endpoint, or whose right endpoint is farthest from its right endpoint). This

corresponds to the Hausdorff distance between persistence diagrams. If each bar may only correspond to a single other bar, this added restriction corresponds to bottleneck distance. In either case, we allow any bar to be matched with the "empty" bar whose birth and death are both the midpoint of the non-empty bar. In a persistence diagram, this corresponds with matching a point off the diagonal with one on the diagonal. The closest diagonal point will be obtainable by decreasing the death time by half of the total duration and increasing the birth time by the same amount.

2.4 Advanced persistent homology

Persistent homology, as we have introduced it above, is a useful tool with some major limitations. One major limitation is that it can only be used to understand finite point sets; another is that it is very computationally expensive. In this Section we take a different perspective, one which allows calculation of the persistence of other objects. We also look at some alternative routes to the computation of persistent homology.

2.4.1 Replacing point clouds with height functions

While the construction we have described above is sufficient for most real-world applications, the full framework of persistent homology is usually approached differently. Instead of trying to find the "shape" of a finite set of points embedded in $\mathbb{R}^d$, we instead try to find the "shape" of a real-valued function. If we allow an

arbitrary function, some pathologies will be problematic. We therefore restrict our attention to a class of functions for which many of our previous statements hold, and note that (on a bounded set) most functions the reader will come across in nature are sufficiently well-behaved.

Definition 2.4.1 (Tame function)**.** A real-valued function f on a topological space is called *tame* if it satisfies the following two properties:

1. f has a finite number of homological critical points
2. Every homology group of every sublevel set of f, that is, a set of the form $f^{-1}((-\infty, a])$ for some real a, is finitely generated.

Now we will discuss taking the persistent homology of a tame function. Just as before, we have a parameterized family of spaces (X_r in the naive construction, the inverse image $f^{-1}([-\infty, r])$ in the more advanced) and we can ask for what ranges of values of the parameter are the generators of the homology of the space stable. The naive construction is a special case of this more subtle framework; if the function we are investigating is "distance from a point to the set X", we obtain the naive construction of persistent homology. The two tameness conditions allow us to again make use of the fundamental theorem of finitely generated modules over a PID.

From a computational perspective, we don't have a general method for computing the inverse image of an arbitrary tame function. If our space is finite, we can check each point individually, but $\mathbb{R}^d$, which is a frequent domain of interest, is

not finite, nor even bounded. Arbitrarily large subsets are, however, triangulable, which makes $\mathbb{R}^d$ somewhat amenable to brute-force algorithms.

We might hope that persistent homology is stable under small perturbations, and indeed it is:

Theorem 2.4.2 (Stability theorem of Cohen-Steiner, Edelsbrunner, and Harer, 2005 [45])**.** *Let $\mathbb{X}$ be a triangulable space with continuous tame functions $f, g : \mathbb{X} \to \mathbb{R}$. Then the bottleneck distance between the persistence diagrams $D_B(f, g)$ is at most twice the L^∞ distance*[14] *between f and g:*

$$d_B(D(f), D(g)) \leq 2\|f - g\|_\infty.$$

The factor of 2 is not present in the original paper, as the authors are using a radius-like scale in their persistence diagrams, unlike our diameter-like scale. In the case of distance from a finite point set, we can rephrase this in terms of the bar code and obtain:

Corollary 2.4.3. *Suppose $\psi : X \to X'$ is a bijection of point clouds, with $d(x, \psi(x)) < \epsilon$ for all $x \in X$. Then the persistence bar code of X' is identical to that of X, with 2 differences. First, the endpoints of each element of the barcode might move by 2ϵ at either end (and thus any element of X's barcode which persisted for less than 4ϵ might die). Second, an arbitrary number of new elements may be added, although*

[14] Recall the L^∞ distance between two continuous real-valued functions is the supremum (taken over all inputs) of their difference.

each will be of size less than 4ϵ.

Proof. Let f be the function whose output at any point is the distance from that point to the nearest point of X, and let g be the equivalent function for X'. Perturbing each point in X by at most ϵ perturbs the distance from a set by at most ϵ, so $\|f - g\|_\infty \leq \epsilon$. Thus, by Theorem 2.4.2, $d_B(D(f), D(g)) \leq 2\epsilon$. By the way the barcode was defined, each point in $D(f)$ corresponds to an interval in the barcode, whose endpoints are determined by the coordinates of the point. As each point in $D(f)$ is moving each of its coordinates by less than 2ϵ, the movement of each interval's endpoints is also bounded by 2ϵ. Points on the diagonal of $D(f)$, which represented empty intervals, may move off the diagonal but only by 2ϵ in each coordinate, so new intervals may be created, but will be less than 4ϵ in length. □

While the stability theorem above is quite strong, we might conjecture an even stronger version. Are there any added conditions which might guarantee, not only that each bar in the barcode is stable, but also that no new bars are added? If, for instance, our function is the distance in $\mathbb{R}^d$ to a finite point set, then there are ranges in which we know H_0 will not change—for instance, the range from 0 to the minimum pairwise distance between the points. If all of the homology is stable in some range, are we guaranteed that an ϵ-perturbation of the points will not introduce any new bars into the barcode? Alas, the following example shows that we do not have this guarantee—the stability result above is, in some sense, sharp.

Consider three points, $\{(-1, 0), (0, 1), (1, 0)\}$ at the vertices of an isosceles

triangle. This collection of points has stable homology as t ranges between 0 and $\sqrt{2}$. In that range, the homology is just that of three points. For values of r between $\sqrt{2}$ and ∞, the homology is that of a single point. But if we shift the top point upwards by ϵ, and consider $\{(-1,0),(0,1+\epsilon),(1,0)\}$, the situation changes. We still have stability from 0 to $\sqrt{2}+\epsilon'$, but we do NOT have stability from $\sqrt{2}+\epsilon'$ to ∞. At $t=2$, an element of H_1 flashes briefly into existence, hanging around just long enough to disprove our conjecture before vanishing at $t=2+\epsilon$ (see Figure 2.10).

2.4.2 Alternate complexes and ease of computation

As a practical matter, how do we compute persistent homology? Once we have a filtration of simplicial complexes, taking the homology is a relatively straightforward task. Obtaining the filtration from the function we wish to analyze, however, is harder. In the fully general case, we would need to be able to determine the homology of arbitrary sublevel sets of a very general class of function. To avoid this, let us once again restrict our attention to point clouds, and consider the ways a simplicial complex may be generated from a point cloud and a scale parameter.

While the Čech complex described in 2.2.3 earlier is, in some theoretical sense, the only "correct" way to compute persistent homology, it has a major downside: it is extremely computationally expensive to create and store. Fortunately, there are a number of other methods for obtaining simplicial complexes from a point cloud which can approximate the Čech complex. These include the Vietoris-Rips complex [84], the Delaunay complex [16], the Alpha complex [64], the Witness complex [55], and

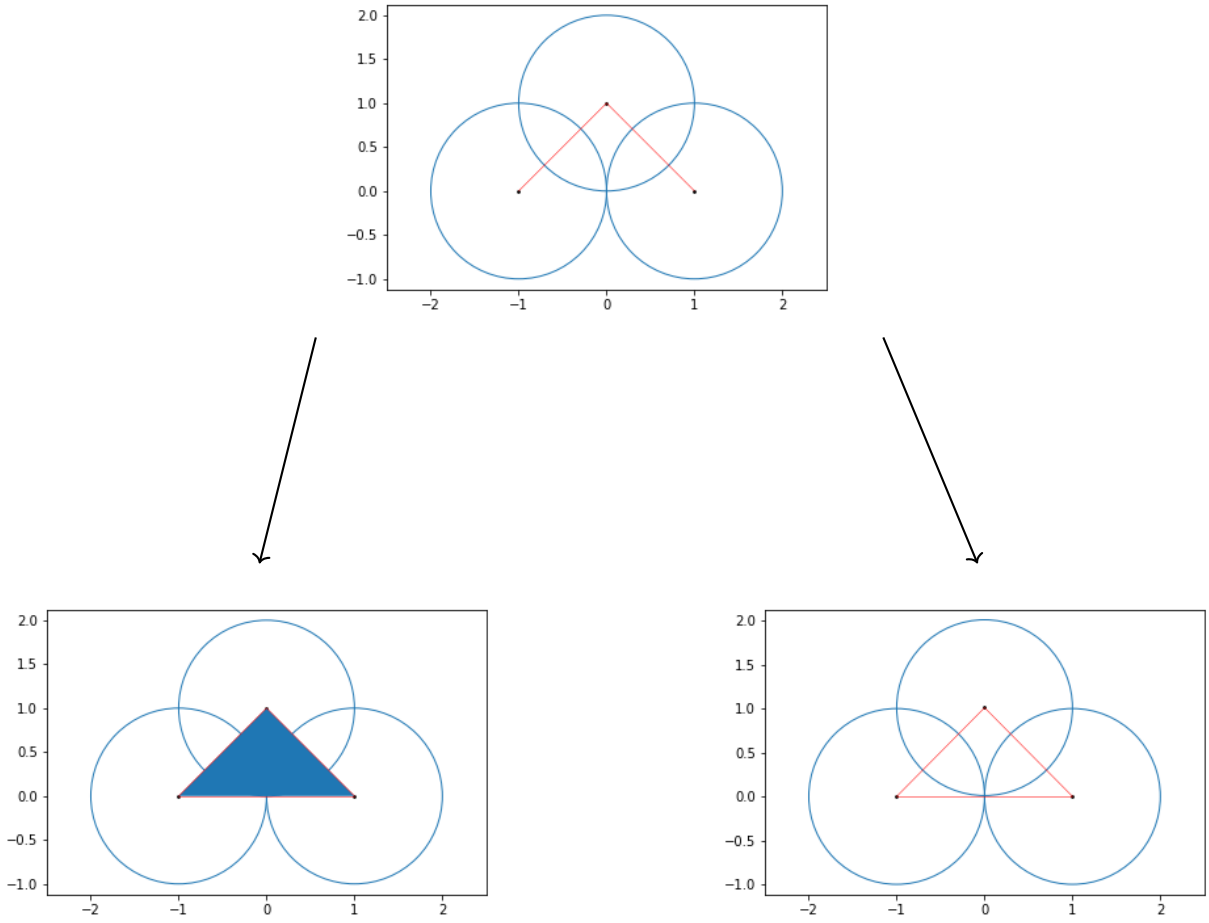


Figure 2.10: A small perturbation of a point cloud can result in a new element of the bar code. In the initial complex, the Čech complex at scale 1 is contractible (top) and will remain so at scale 1.001 (bottom left). However, if the top point is shifted up by ϵ, there will be a scale at which the Čech complex has non-trivial homology (bottom right).

many others. We describe a small sampling of the many possibilities here.

2.4.2.1 Clique complexes and Rips complexes

One of the more computationally difficult parts of computing a Čech complex is checking every subset of X which has cardinality $k+1$ to see if there is a point within r of all of them. Although there is an algorithm to find the radius of the bounding sphere for a finite set of points which is linear in the number of points, the Čech complex requires calculating this for every subset of our point cloud, and then storing the resulting (possibly very many) simplices. Clique complexes provide a way around this issue.

Definition 2.4.4 (Clique complexes)**.** A *clique* complex is a simplicial complex which is maximal for a given set of edges. That is to say, if $\{x_1, \cdots, x_k\}$ is a collection of points which do not form a simplex, then there is some pair $\{x_i, x_j\}$ which does not form an edge.

A set of edges uniquely determines a clique complex. Unsurprisingly, there is a name for the clique complex determined by the edges of a Čech complex.

Definition 2.4.5 (Vietoris-Rips complex)**.** The Vietoris-Rips complex of a point cloud X at scale r is the clique complex of the 1-skeleton of the Čech complex at the radius r.

The Vietoris-Rips complex is much easier to compute and store than the Čech complex. All the information needed to construct it is stored in the matrix of pairwise distances, and no additional computations need to be performed to detect

if a given set of points form a simplex at a given scale. Moreover, the Vietoris-Rips complex is related to the Čech complex by the following theorem of De Silva and Ghrist [56]:

Theorem 2.4.6. *Let X be a point cloud in $\mathbb{R}^d$ and $C_r(X)$ (resp. VR_r) denote the Čech (resp. Vietoris-Rips) complex of the natural good cover of X_r. Then there is a chain of inclusions $VR_{r'}(X) \subset C_r(X) \subset VR_r(X)$ whenever $r/r' \geq \sqrt{2d/(d+1)}$. Moreover, this ratio is the smallest for which the inclusions hold in general.*

So, if we have the persistent homology of the Rips complex of X, any element that persists long enough corresponds to an element of the persistent homology of the Čech complex. Theorem 2.4.6 also guarantees that any element of the persistent homology of the Čech complex with sufficiently high persistence will show up as an element of the Vietoris-Rips complex.

We might have hoped that either every element of the persistent homology of the Vietoris-Rips complex corresponds to an element of the Čech complex, or vice versa, but the following examples show that is not the case.

Let X be the vertices of an equilateral triangle with side length 1. Then the Čech complex of X contains an element of $\mathrm{H}_1(X)$ that begins at $t = 1$ and persists until dying at $t = 2/\sqrt{3}$. But the Vietoris-Rips complex never displays any nontrivial H_1.

Conversely, consider a regular polygon in the plane with $2n + 2$ vertices, with r such that every pair of points is connected if and only if they are not antipodal (i.e., r is just shy of including the center). The Vietoris-Rips complex of this has a

nontrivial element of homology of dimension n, despite the planarity of the figure!

Recall that in the Čech complex case, we constructed a counterexample to the idea that, for a sufficiently small perturbation, no new elements of the barcode would appear (see Figure 2.10). Perhaps we hope that, while we do not have this type of stability for the Čech complex, we may have it for the Rips complex. After all, the above complex was a triangle, so the associated Rips complex will never have non-trivial homology above dimension 0. Alas, this also fails. Consider a set of four points arranged in a square with side length 1. The associated Rips complex has the homology of four points for $0 < t < 1$, then has the homology of S^1 in the range $1 < t < \sqrt{2}$. Past $t = \sqrt{2}$, the homology is once again trivial. If we perturb the top right corner, letting it swing down in a circle around the bottom right corner (maintaining a constant distance of 2 between the two points), then at some point the homology vanishes.

Despite these many deficiencies, the ease-of-use of the Rips complex, together with the theoretical guarantee given by Theorem 2.4.6, makes it the complex of choice for almost all practical applications. Every piece of software to compute persistent homology the author is aware of defaults to using the Rips complex. As a result of this, we will focus our study of stability in Chapter 4 on the homology of the Rips complex derived from a point cloud. This is also the root of our decision to use diameter-like information rather than radius-like.

2.4.2.2 Delaunay complexes

Unlike the Čech or Rips complexes above, the Delaunay complex does not have a scale parameter. It is entirely determined by the point cloud X.

Definition 2.4.7 (Voronoi cells; Delaunay complex)**.** Given a point cloud $X \subset \mathbb{R}^d$, we will decompose the ambient space into regions with one region for each point of X. The region corresponding to x_i consists of all the points of $\mathbb{R}^d$ which are closer to x_i than to any other point of X. These regions are called the *Voronoi cells* of the decomposition. The Delaunay complex of X is given by a 0-simplex for each Voronoi cell, a 1-simplex for each pairwise intersection of Voronoi cells, etc.

See Figure 2.11 for an example of a point cloud and the Voronoi cells of its decomposition.

The Delaunay complex for a point cloud in $\mathbb{R}^d$ will be contractible, making it less than ideal for topological analysis.

2.4.2.3 Alpha complexes

The alpha complex is, in a precise way, a compromise between the Delaunay and Čech complexes.[15] Recall, the Čech complex at scale r is the nerve of the space $\bigcup_{x \in X} B_r(x)$ with respect to the obvious cover. We can obtain an alternate cover to the space by intersecting each $B_r(x)$ with its respective Voronoi cell. This will give us a good cover of our space, so the nerve, called the alpha complex, will be

[15] Alpha complexes were first introduced by Edelsbrunner in the early 1980's [67, 70], and are thus arguably a part of the inspiration for persistent homology in general.

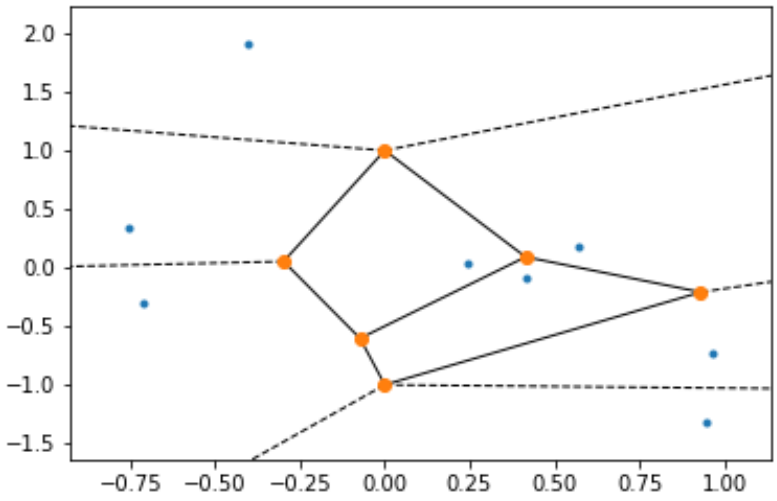


Figure 2.11: A small point cloud (points in blue) and the Voronoi decomposition of $\mathbb{R}^2$ it induces. The corresponding Delaunay complex will have a point for every region (happily, this means a point for every blue point), an edge for every boundary between regions, and a 2-simplex for every triple intersection, shown here as orange points.

homotopy equivalent to the original space, retaining the best feature of the Čech complex. Moreover, unlike the Čech complex, the alpha complex will, in general, have no simplices of dimension higher than the ambient dimension of the space, $\mathbb{R}^d$, which simplifies computations considerably. In contrast, the Čech complex can have simplices of dimension up to $|X| - 1$, which is huge! And worse is the fact that, by Theorem 2.2.4, these simplices of the Čech complex do not add anything to the eventual homology calculation.

In low-dimensional situations, the alpha complex is an excellent choice. As the dimension increases, however, it almost immediately becomes an impractical tool. As an informal explanation, the problem is that the number of hyperplanes bounding each cell goes up swiftly, resulting in an exponent of roughly $\lceil d/2 \rceil$ for

many computationally relevant quantities such as the number of vertices in the Voronoi decomposition, the number of simplices in the Delaunay complex, and many others. But for 2 and 3 dimensions, alpha complexes remain a natural choice.

2.5 Summary

In this Chapter, we have established the basic framework of homology and point clouds; we have discussed the application of homology to measuring the shape of point clouds by persistent homology; we have extended the framework of persistent homology to analyze objects other than point clouds; and we have briefly mentioned some intermediate objects which can replace Čech complexes.

In Chapter 4, we will take advantage of both the stability result stated earlier, and some of the alternate complexes proposed, to find alternate methods of speeding up the computation of persistent homology.

First, however, we shall use Chapter 3 to take a detour into an application of persistent homology to path-planning, suggested by Bhattacharya, Ghrist, and Kumar [20]. While we will present no original theorems in the upcoming Chapter, we believe that some of the insights gained in the course of implementing their algorithm may prove useful to others.

Chapter 3: Applications of persistent homology to path planning

In this Chapter we will discuss the use of unmanned robots with radiological sensors to find radioactive materials in unfamiliar environments. The first several Sections lay out the engineering framework of radiological detection, and discuss some proof-of-concept experiments which have been run. In Section 3.4.3 we discuss the implementation of an algorithm of Bhattacharya, Ghrist, and Kumar which applies concepts from persistent homology to path planning, and some insights from working with this in a real-world context.[1]

3.1 Introduction

The use of unmanned aircraft to assist in radiological surveys, both pre- and post-detonation, remains an emerging technology coincident with the evolution of unmanned aircraft and payloads for sensing and sampling. Some of the past novel work includes development of a tethered nuclear materials sampling payload, a spatially variant deconvolution method for source localization, and collaborative Un-

[1]This Chapter is a lightly edited and expanded version of a paper [90] I coauthored with Wojciech Czaja and Weilin Li at University of Maryland; Kevin Kochersberger, John Peterson, Prashant Kumar, and John Bird at Virginia Tech; and Morgan McLean at the Remote Sensing Lab. The research was supported by the Defense Threat Reduction Agency Basic Research Program. A different perspective on some of the experiments described herein is given by my colleague Weilin Li in [92].

manned Aerial Vehicle (UAV) Unmanned Ground Vehicle (UGV) sensing and localization to safely map source distributions. As threats continue to emerge and the acceptance of unmanned aircraft grows, an expanded portfolio of uses—including event clearing, intelligent search, post-detonation analysis, and tiered search strategies—is being proven.

This Chapter reports on work at Virginia Tech in pre- and post-detonation data collects with UAVs ranging from 10 kg hexacopters to 90 kg single-rotor helicopters. We have shown the integration of UGVs into a measurement system designed to reduce risk to first responders while generating accurate analytics of chemistry and location when mapping radiological distributions. Results are presented for several experiments that define use-cases of interest to first responders.

3.2 Motivation

The use of unmanned systems, both UAVs and UGVs, to assist in the search of nuclear materials has been an outgrowth of applications in disaster response, with significant investment after 9/11. Despite the advantages of autonomous unmanned systems, the transition to full autonomy has been slower than with other Unmanned Aerial System (UAS) applications, such as 3D mapping. This is due in part to a conservative response community that has shown reluctance in the past to embrace the full benefits of autonomy in support of human-machine teaming. More recently, acceptance of perception and autonomy in nuclear materials identification and localization has accelerated, and their useful applications are growing.

Scalable and complementary unmanned systems are a focus of research to integrate small autonomous systems into existing search methods that include large manned aircraft as well as backpack-carried detectors. Some of the mission protocols include initial assessment of a post-detonation environment, localization of sources, 3D data collection, and the expanded use of sensing in the non-visible EM bands.

3.3 Radiological isotope measurement and modeling

After collection of aerial radiological measurements and terrain imagery, there is a challenge in finding the ground-level radioactive distribution which has numerous solution strategies. Kochersberger, et. al [43], showed that a spatially variant point spread function could be employed in deconvolution to map the location of point sources using raw gamma count data. Figure 3.1 shows the result of localizing two closely spaced sources from a 60 m overhead scan using a Yamaha RMAX autonomous helicopter. The stronger source obscures the location of the weaker source in the summed gamma counts shown on the left.

The use of summed gamma counts is a first-order method of localization. Incorporating spectral and visual data has the potential to identify sources with an intensity nearly equal to the background radiation level. Computational harmonic analysis has been applied to support dimension reduction, multiscale representation, and multi-modal data fusion resulting in robust spectral anomaly detection. Sources that would normally be buried in background radiation signatures are extracted with these methods.

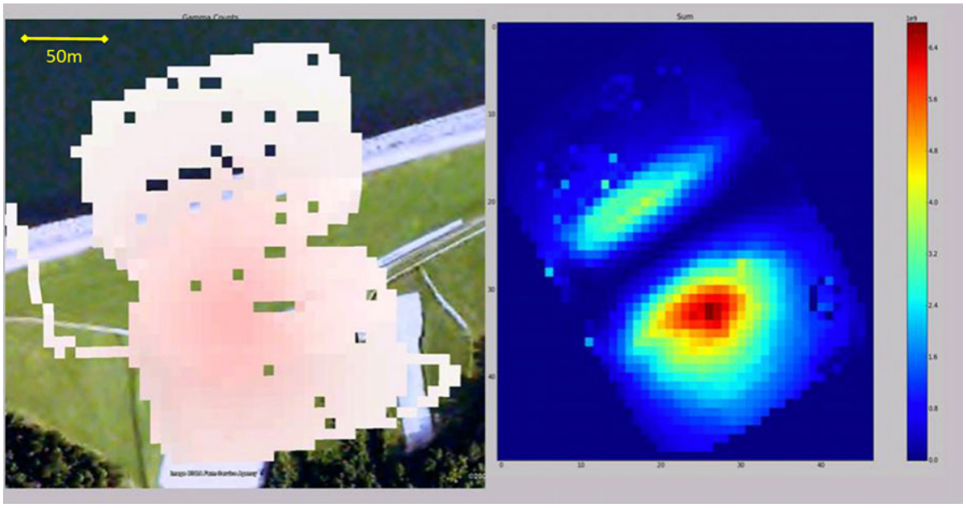


Figure 3.1: Deconvolution method of source localization. The presence of a weaker source is hidden in the raw count data shown on the left, but is revealed when deconvolution is applied, shown on the right.

Laplacian Eigenmaps (LE) and Schrödinger Eigenmaps (SE) have already been successfully used to perform, respectively, unsupervised and supervised clustering of radiological data [53]. An example of this is seen in Figure 3.2, where the application of the Eigenmap nonlinear dimension reduction reveals relatively subtle spectral signatures without the need for a human spectroscopist. Class 5 in the Figure is determined to be significant despite the fact that the gross count levels were the same as background; small peaks in the low energy region are discerned from the unsupervised method indicating the unexpected presence of Cs-137.

Moreover, preliminary results have shown that the Fourier Scattering Transform (FST), a hybrid Fourier and neural network transform that was developed in Czaja and Li [51,52] is effective at locating relatively weak sources. Figure 3.3 shows a plot of predicted radiological anomaly source location using FST.

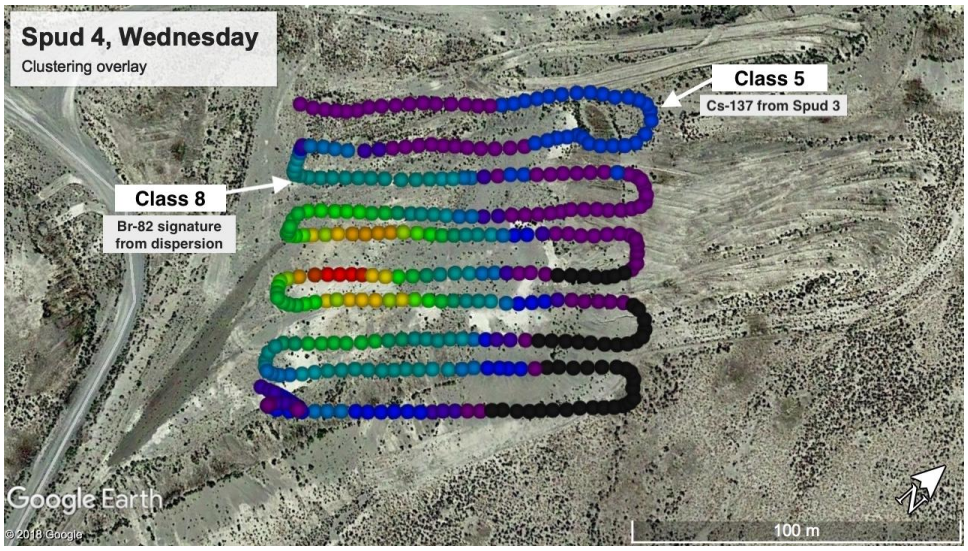


Figure 3.2: An aerial scan conducted by Virginia Tech with a 2" x 2" Nal scintillation-type detector over a dispersion site of Br-82 using conventional explosive.

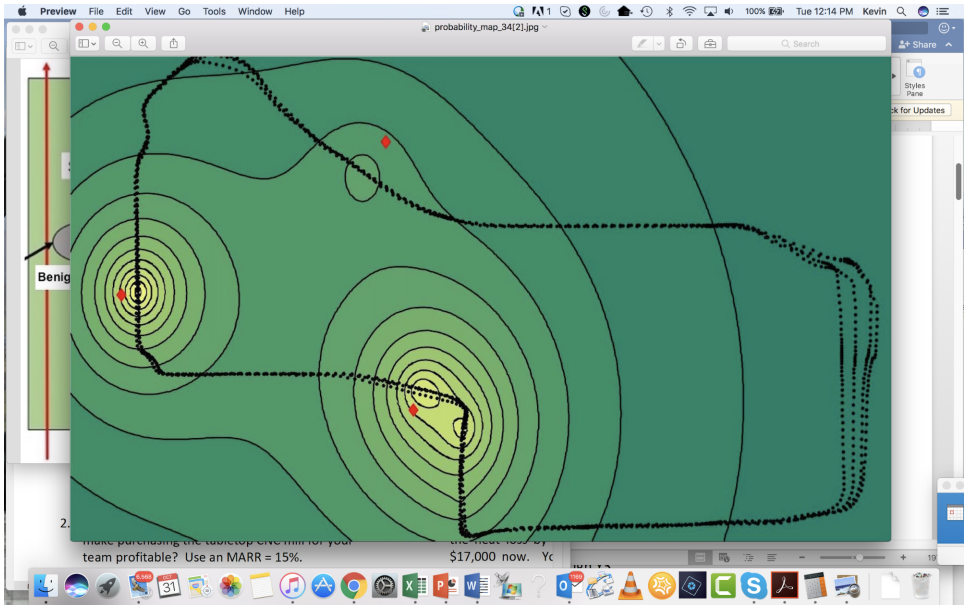


Figure 3.3: Spectral data showing source prediction using FST. The black points are sample points and the red diamonds are true source locations. Half of the data was used as a training set for the FST coefficients while the other half was used for test.

3.4 Multi-agent search methodology

Operating a multi-agent search mission has benefits in complex environments where air and ground systems provide complementary data for analysis. Aerial search can be employed in hard-to-traverse areas, and terrain classification is simplified using nadir aerial imagery. Ground robotic systems can be directed to areas of interest based on classified traversability from the aerial imagery, and they have the capability of long dwell data collection not possible using UAVs. Combined, this data allows the robotic system to answer the question: what does the classified environment and the radiation map tell us about the next place to search? Furthermore, generalizing a search mission to include human agents offers additional flexibility to the autonomous system while improving overall search efficiency.

Our approach to a multi-agent search and localization strategy is subdivided into three main tasks: Data collection, initial radioactive source location/distribution estimation, and final estimation. The following Sections will describe different data collection hardware and approaches for source estimation.

3.4.1 Data collection

We have performed several experiments, listed in Table 3.4.1. Here we give some details of the UAVs which which flew the experiments.

Methods of aerial data collection range from use of the Yamaha RMAX autonomous helicopter carrying a 3" x 9" NaI scintillation-type detector, to a custom-designed hexacopter that carries a 2" x 8" crystal or 2" x 2" crystal plus a Cadmium

Date	Location	Experiment Description
1/18	SRNL	Aerial scan, identification of sources, classification, and secondary ground robotic and aerial long-dwell data collection
4/18	INL	Aerial scan over dispersion site of Br-82. Data is dense enough to create accurate plume maps. Dimension reduction is used to find weak isotopes
7/18	INL	Another aerial survey campaign over dispersal site of Br-82
10/18	NNSS	Scan at crater to identify interesting isotopes. The system was designed for BVLOS flights inside the crater

Table 3.1: This Table lists the experiments run by the UAV/UGV described in this Chapter.

Zinc Telluride (CZT) imager. The RMAX, shown in Figure 3.4, has been equipped with a stereovision system useful in classifying terrain traversability and performing 3D reconstructions. As a 90 kg system capable of lifting 20 kgs of payload, the RMAX has also been used to carry a custom-designed, tether-deployed ground sampling robot which remains tethered throughout its mission for retrieval.

A VT-designed hexacopter shown in Figure 3.5 has been developed as a low-cost aerial platform with a 6 kg payload capacity. Its high payload capacity combined with a Swiftnav Piksi Multi Real Time Kinematic (RTK) Global Position System (GPS) allow it to accurately hold position while carrying the gamma-ray imager.

A smaller hexacopter adapted from a commercially available airframe, shown in Figure 3.6 has also been utilized. An Nvida TX2 computer and Swift Piksi Multi RTK GPS provide computational power and positional accuracy necessary for adaptive sampling and route planning for ground systems.

Figure 3.4: RMAX with Nal detector and stereovision system in-flight.

Figure 3.5: VT-designed hexacopter with CZT and Nal detectors installed (16.8 kg gross weight).

Figure 3.6: The UAV and UGV used in the experiments in Table 3.4.1

.

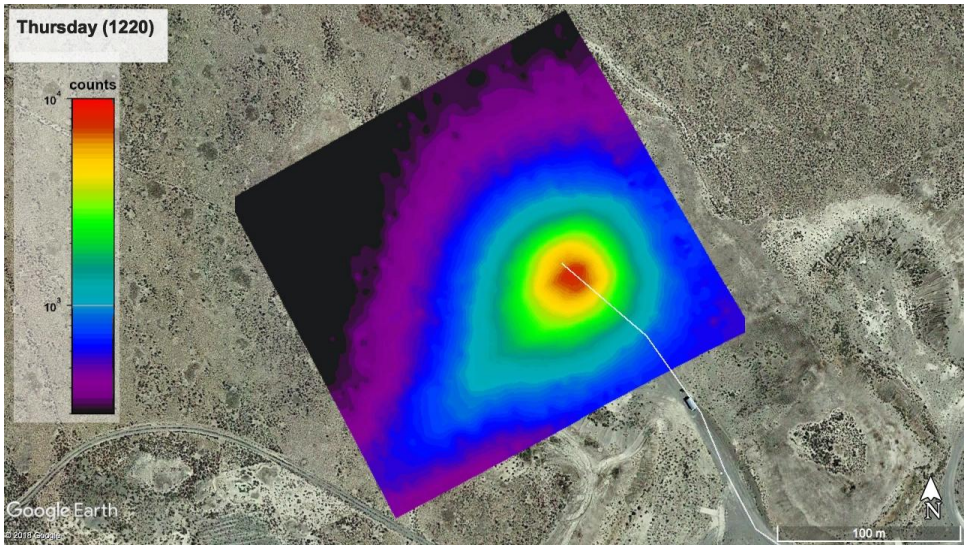


Figure 3.7: Radiation contour generated from the INL flights, July 2018.

An example of the utility of drone measurements is provided in Figure 3.7. The spatial resolution here is much greater than can be obtained by a manned aircraft, since the speed and altitudes of manned aircraft do not favor data resolution. Drones fill a significant need for higher resolution data in a tiered discovery, where initial findings from manned aircraft point to areas of interest that demand further investigation by drones. More accurate source classification and safe perimeter definition follow from the higher quality data products.

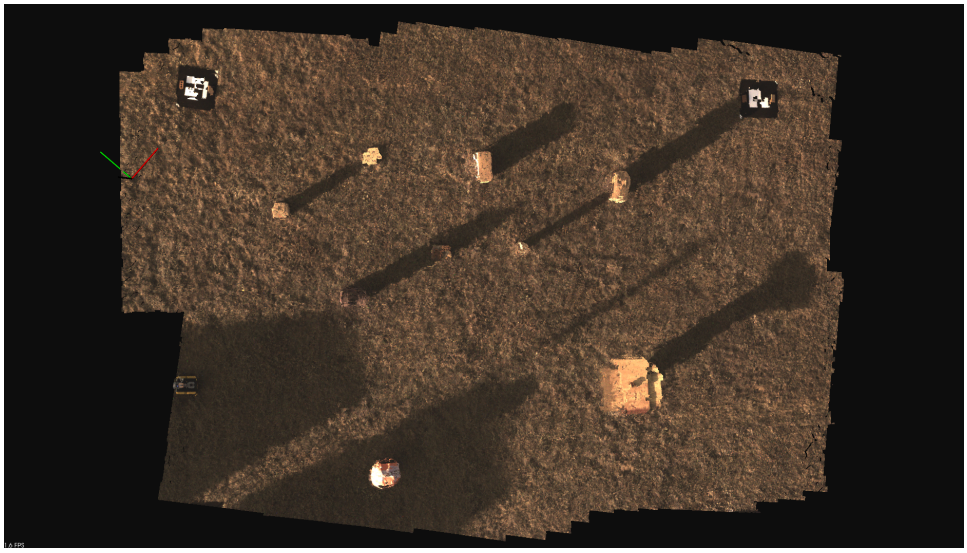


Figure 3.8: A progressive 3D mosaic created during image capture to accelerate ground-based route planning for inspection in time-critical situations.

Radiation measurements from an unmanned aircraft should be complemented by nadir imagery which both provides context to the radiation data and is also used to inform ground robotic and manned searches. Using a custom stereovision system, the Virginia Tech team has developed a data processing architecture to compute disparity maps and find feature point matches to produce 3D mosaics for route planning during an overflight, accelerating the speed at which ground-based inspections can occur shown in Figure 3.8. Christie [43] showed the benefits of reasoning about 3D and 2D information to obtain accurate classes on the ground which reduce mission failure with low-risk route plans in a detailed search.

3.4.2 Online radioactive source location estimation

A challenge in most data collection scenarios is to extract meaningful source information when background radiation levels can be at the same order of magnitude

as the source. By leveraging an analytical model between the counts observed in a detector volume and the activity of sources of radiation, environments may be efficiently explored to reduce misclassification risk and hypothesis uncertainty.

This analytical model was derived for a spherical detector volume with a mono-energetic point and validated in Geant4 [3], a particle transport simulator. If the relative position of a measurement and a source is known, then there is a linear relationship between the unknown activity of the source and the counts recorded in the measurement. A set of hypotheses on the position of sources within the environment can be constructed. Then the strength as well as an uncertainty of each source in each hypothesis may be estimated through Maximum Likelihood Estimation. The likelihood of each hypothesis itself may be computed to identify the most likely hypothesis out of the set.

With this model, it is possible to estimate the expected reduction in uncertainty that would occur in each hypothesis if the robot were to make additional observations. For each candidate in a set of trajectories, this improvement may be characterized by metrics such as mutual information and the reduction in misclassification risk, allowing the system to select the most informative trajectory. Misclassification risk is the risk that given a user defined threshold on source activity, a source which is currently estimated to have a strength above that threshold is actually weaker than that threshold and vice versa. This risk may be estimated on a per source basis by examining the estimated strength and uncertainty in strength of each source in each hypothesis. Mutual Information, given by the difference in entropy of the distribution, measures the reduction in uncertainty in the estimate

of the strength. Maximizing Mutual Information maximizes the reduction in uncertainty. By optimizing over these rewards, the system is able to select the most informative trajectory through the environment.

This method enables observations from multiple independent robots with separate radiation detectors to be combined to form a single estimate of the position and strength of a source in the environment. The planning method described above allows the paths of these robots to be coordinated to efficiently explore the environment by considering their joint effects on the distributions.

Figure 3.9 shows an example of a trajectory navigated by the UGV when aerially collected radiation measurements were used to initialize the model for a good initial guess on the location of the point source.

3.4.3 Terrain based source inspection

We approach the problem of source estimation as a two-step process where the second step leverages data from the first round to improve the allocation of air and ground sampling resources. Reconstructing the 3D environment from nadir imagery and semantically classifying the visual 2D data allows ground vehicles to efficiently navigate the environment. Figure 3.10 shows an experiment conducted at Kentland Farm, near Virginia Tech, where the first aerial scan provides a source location estimate. The UGV then navigates to the estimated source location to collect longer dwell data using the estimated terrain classes to minimize cost.

Operational tempo is maintained in the overall search mission by generating

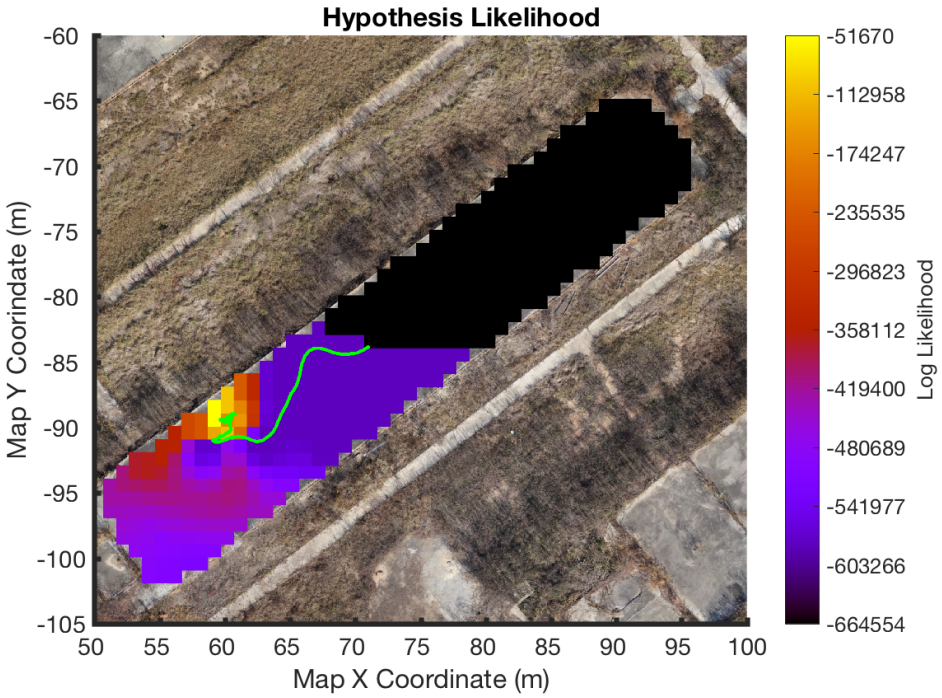


Figure 3.9: Trajectory taken by UGV to localized point source of radiation. This trajectory was informed by aerially collected radiation measurements. Color indicates the log likelihood of hypotheses on the position of the point source. Warmer indicates more likely.

Figure 3.10: In this example, aerial data (left) is collected over two weak sources (pink), and a single source (red) is identified. Starting 100m away, a ground robot takes the minimum cost path (right) to visit the source based on a power consumption heuristic and distance.

3D terrain reconstructions while the initial survey flight is in progress. Aerial data will provide the bulk of the planning information, but ground vehicles will augment this with higher resolution data for obstacle detection and the capacity for on-line learning of terrain costs to reduce the dependence on comprehensive training data sets.

Using aerial data generated by the UAV, the LE spectral clustering provides an initial probability distribution map for the distribution of radioactive sources. The UGV may use this probability distribution map and generate a list of candidate locations for radioactive sources.

Once the UGV has a candidate location to inspect and a semantically classified map, each given to it by the UAV, it must select a path. If the traversability status of all the terrain is fully understood, the UGV can pick a path minimizing time, distance, energy-cost, or some other criterion. If the traversability status is uncertain, however, whether through uncertainty in the environmental semantic classification (is the black streak asphalt or mud?) or through uncertainty in the robot's own capabilities, (will it get stuck in the mud?) we may be faced with the unenviable task of trading off a safe path (one with a high degree of certainty in traversability) with a path that is otherwise desirable (shorter, or generating more Bayesian information).

One solution for a problem of this type was suggested by Bhattacharya, Ghrist, and Kumar [20]. They give an algorithm that takes an image where each pixel has been assigned a probability of traversability, along with a desired start and end point, and returns a path from the start point to the end point that is intended to

be a desirable path in the face of uncertainty. Their algorithm is:

1. Pick a finite set of probabilities at which to make calculations. If unsure which probabilities to pick, a reasonable choice is equally spaced numbers from 0 to 1, including both endpoints.

2. At each probability picked above, we create a derived image, one where each pixel is either 1 (all pixels at or above the probability threshold are counted as traversable) or 0 (all pixels below the probability threshold are counted as non-traversable).

3. For each of these images, calculate all $\mathbb{Z}/2$ homology classes of paths from the start pixel to the end pixel. If there are n obstacles that may be navigated around, there will be 2^n classes of paths (roughly speaking, a path may pick "go clockwise" or "go counterclockwise" to avoid each obstacle), so it may be desirable to ignore all obstacles below a certain size for computational efficiency.

4. If $0 < p < q < 1$, then every path in the image with threshold q is a legitimate path inside the image with threshold p—at the extreme, a path with threshold 1 only goes through pixels which are guaranteed to be traversable. Calculate the "persistence" of a path based on this correspondence.

A point the author found to be of practical importance in implementing the first step of this algorithm: in practice, we are not handed a "probability of traversal," but rather an image of an area which has been semantically classified together

with some information about our UGV's capabilities. Even if our semantic classification and assessment of the UGV's capacity is correct enough that we can assign a "traversability score" to every pixel and be guaranteed that the pixels with a higher score are more likely to be traversable, there is a large amount of freedom in how to map these scores onto probabilities from 0 to 1. Even the most straightforward methods, such as linear and log-linear mapping, often produce vastly disparate results.

The last step in this algorithm is more subtle than it appears, and so is deserving of more explanation. Just as in Chapter 2, we have a map that can be thought of as a time progression map (including the paths with more certain traversability into the regions with a lower traversability threshold). Further paralleling Chapter 2, this map induces a map on the homological objects of interest. These objects were traditional elements of homology in Chapter 2 and are paths whose homology classes we are interested in here. Still following Chapter 2's example, any element not in the image of a stage of this map is one we can consider to be "born" at that time, and any element in the image with a unique preimage is "persisting". Unlike the persistent homology groups described in Chapter 2, homology classes of paths between two fixed points do not have a natural group structure, or distinguished 0 element, so there is no question of an element dying by "being mapped to 0". The only way for a class to die is for it to merge with another class, and in that case, which class dies, and which survives? In Chapter 2, we resolved this by resorting to the classification of finitely generated modules over a PID, but, as the present case lacks even a group structure, we certainly can't make use of that. Bhattacharya,

Ghrist, and Kumar suggest looking at the set of preimages which mapped to the homology class of some path γ, and pick the one with the lowest Hausdorff distance to γ—consider that as being part of the same persistence class, and declare all other preimages dead.

Another option, original to this author, is to parameterize the paths, whether through distance or expected travel time of the robot and take an integral over the parameterizations of the distance between γ and each of its preimages. We use this as the distance, then proceed as in Bhattacharya, Ghrist, and Kumar's original idea.

This algorithm produces a path that is desirable in that it is minimally sensitive to our uncertainty in the traversability of the terrain. If there is terrain that cannot be traversed, this path is "close to" one that is more likely traversable.

Once the UGV reaches a target location, it collects higher quality data for the supervised FST algorithm to classify. The FST computes oscillatory features of the spectral data and satisfies several desirable properties for classification: it is energy preserving, it is stable with respect to additive noise, and it contracts sufficiently small translations and diffeomorphisms.

After classification, the spectral information corresponding to the scouted location is incorporated into the global dataset and the probability distribution is recomputed using the supervised spectral clustering algorithm of SE. The path for the UGV is recomputed according to this new distribution. This process continues until all suspected locations are checked.

When an environment is large, complex, and includes locations inaccessible to ground vehicles, the combined deployment of both UAVs and UGVs allows the

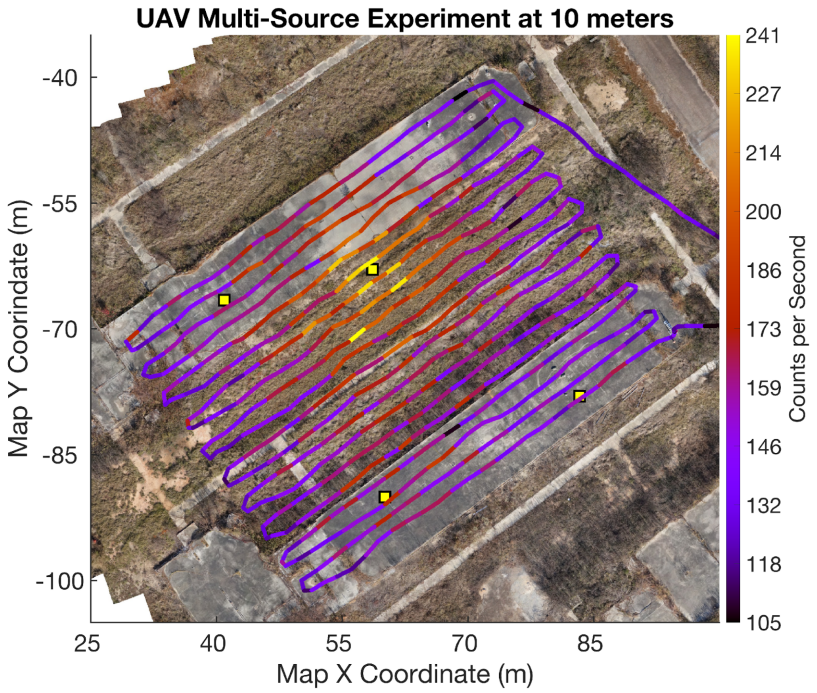


Figure 3.11: Initial exhaustive scan result. The yellow points indicate where sources have been placed. The colored track indicates the path of the UAV and the measured counts per second.

system to localize sources. Multi-agent task allocation ensures that the environment is efficiently and completely explored while considering the mobility constraints of the vehicles and the effectiveness of the sensors equipped to each vehicle.

A demonstration of a search and classification mission with variable terrain traversability was performed at a National Labs in January, 2018. In this experiment, three sources were located on concrete pads, considered traversable for the UGV, and one source was placed in a grassy area, considered less traversable for the UGV. An initial scan of the area provided source contours that would be used to inform a secondary search using either the UGV or UAV, depending on the terrain type. Figure 3.11 shows the initial scan result.

LE dimension reduction was performed on the exhaustive search data to estimate locations for secondary exploration. Additionally, the aerial imagery was classified to determine where the UGV can explore, and a route plan was generated for both the UAV and UGV, shown in Figure 3.12. The air and ground vehicles are tasked to dwell at each exploration point to obtain higher quality data.

For contrast, we include the alternate, high-persistence paths generated by our implementation of Bhattacharya, Ghrist, and Kumar's path-planning algorithm, in Figure 3.13. Some of the most notable differences between these paths and those the UGV actually took, shown in Figure 3.12 are due to an eccentricity of the DOSL library created by Bhattacharya[2] [19], and used in our work—it has a preference for straight lines in one of the 8 cardinal directions, as it is locally going pixel-by-pixel. This results in, e.g., the parallelograms made by both the red and the blue pathways.

3.5 Discussion of topological path planning

Other than the differences created by the above feature of DOSL, the paths generated by our path planning software are quite similar to those generated by the standard methods. Some thoughts on the contrasting advantages of each:

1. The largest advantage of the path planning algorithm is that, as an algorithm, it does not require human input after the initial set up. We caution, however, that the initial set up is not trivial, and there are many choices to be made that human thought is important for. For example, the choice of what size

[2]The DOSL library itself makes use of OpenCV [23].

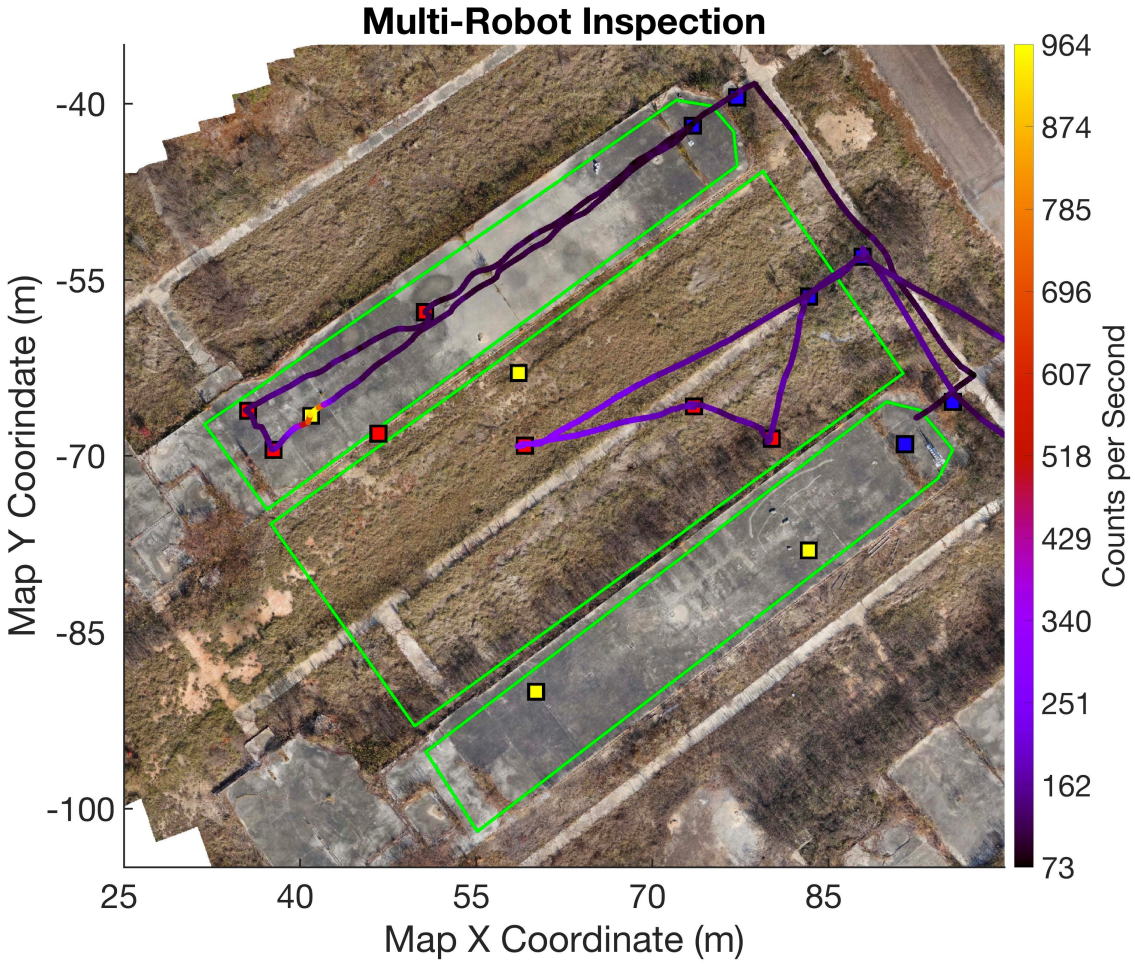


Figure 3.12: The top and bottom concrete pads and the center grass area are identified as traversable and less-traversable regions respectively for the ground robot (UGV). Yellow points are the actual radiation sources while the red points are the estimated location. In this image, the purple line is the path the robot took, using standard navigational methods.

Figure 3.13: These are alternate paths the UGV could have taken, generated by our implementation of Bhattacharya, Ghrist, and Kumar's path-planning algorithm. We color the concrete areas white, representing the semantic identification of the concrete pads as certainly traversable.

obstacles to exclude, or how many layers to calculate the paths within.

2. The largest disadvantage to the algorithm is the running time. Generating Figure 3.13 took more than four hours of computer time. Downsampling the image, calculating the paths at fewer layers, and choosing to ignore all obstacles below a certain threshold can all speed up the calculations dramatically, but all run the risk of worse outcomes.

3. The final paths generated by the topological algorithm and a human operator were fairly similar.

Because of this, for terrain which can be semantically classified as having a very high likelihood of traversibility, we recommend against this topological method–the necessity of calculating two to the power of (number of obstacles) paths in each of possibly many layers is a high computational burden for not much gain. In highly uncertain environments, however, more investigation is warranted.

Bhattacharya, Ghrist, and Kumar do not provide a quantity of practical interest which this path maximizes or minimizes. It is difficult to do so—the most naive formulations of such a criteria have counterexamples. The counterexample shown in Figure 3.14 actually shows that if each pixel's chance of being traversable is independent, no algorithm that only takes the homological classes of the paths into account can succeed in being minimally sensitive to the traversal probabilities.

If we add the hypothesis that the traversability of the pixels is strongly correlated, in the sense that "if pixel p, with chance of traversability $P_T(p)$, is traversable, then so is every pixel q such that $P_T(q) \geq P_T(p)$," the counterexample mentioned

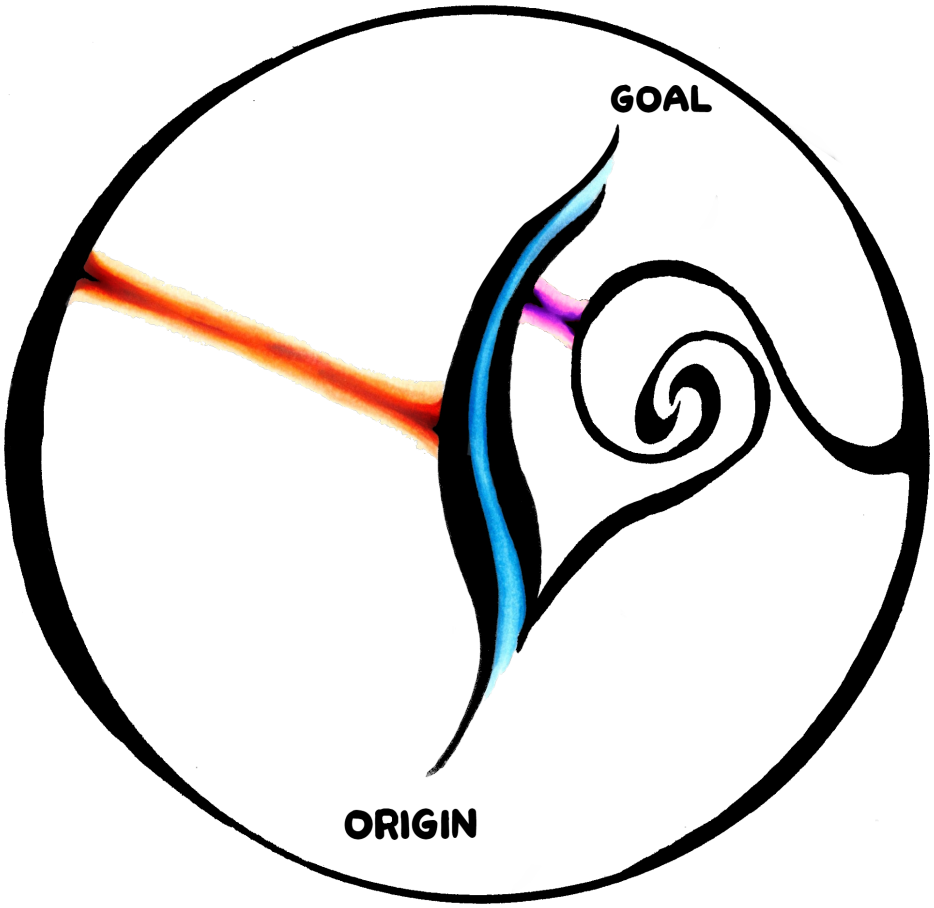


Figure 3.14: Black represents non-traversable terrain, red represents terrain with chance of traversability p, blue represents terrain with chance of traversability ϵ higher than p, and pink represents terrain with an intermediate chance of traversability. While the central path through the blue region is both shortest and has the most persistent homological class, as a practical matter it only takes a single pixel of blue being non-traversable to necessitate a very large divergence from that path. Then if the second most persistent path is the backup choice, on the eastern side, then again a small amount of bad luck can necessitate a great deal of backtracking. The western path, while marginally less persistent, will only require a single traversable pixel of red to be found.

above is no longer a problem. While this sounds highly restrictive, it is, in practice, perhaps more plausible than we might fear—if, for instance, the terrain includes areas that we can semantically classify as "very muddy," "somewhat muddy," and "less muddy," then we may intuitively expect that a robot's ability to traverse areas labeled "somewhat muddy" would imply an ability to traverse "less muddy" areas, without guaranteeing its ability to navigate "very muddy" areas.

Or, for an example less directly relevant to the current circumstance, consider a configuration space. It may be that a system's ability to enter into some types of states, represented by a region of the configuration space, is uncertain, but that the uncertainty is entirely caused by boundary conditions—after the system successfully achieves one state in the uncertain region, its ability to transform into other states in the same region is assured. Think for instance of a robotic arm with many joints, with the position of the arm and its joints represented as a single point in the space, in an environment containing obstacles we may or may not be able to fit the arm between. Once the arm is in a new region, it can move freely.

There are some tantalizing parallels between this and simulations of proteins, where we are not able to move freely through the state space, but rather are constrained by physical chemistry. In this case, the "uncertainty" of being able to enter a region is replaced by a high energy requirement to transition between otherwise stable states. A multistage path to changing the folding structure of a protein, or understanding the natural folding pathways, could conceivably take advantage of some of the path-planning we have discussed in this Chapter. See, e.g., [5] for a recent example of a successful attempt to understand a path through such a configuration

space.[3]

3.6 Conclusion

This Chapter has showcased a use of persistent homology, in path planning for UGV's after preliminary UAV flights in response to radiological incidents. UAV's are able to fly at low altitudes enabling them to make effective radiation measurements despite their limited payload capacities. Methods such as LE and SE have been used to automatically detect anomalous sources in raw gamma-ray spectra. Aerial vehicles are also suitable to carry scene mapping sensors in the form of either Lidar or stereo vision systems. This combination of scene information and radiation measurements not only improves the accuracy of radiation models, but also enables terrain informed navigation for UGVs to investigate candidate source locations. Operational tempo may be improved by utilizing online algorithms for terrain reconstruction and source localization.

As we closed with a discussion on the use of topological path planning, one of the major disadvantages was the length of computation time required. In the next Chapter, we will lay a foundation for a technique to speed up the calculations of persistent homology.

[3] In this case, the path represents a pair of proteins undergoing a coupled folding-binding event.

Chapter 4: Results on stability of persistent homology and dimension reduction

Several of the steps involved in computing persistent homology are computationally intensive, sometimes prohibitively so. In this Chapter, we will discuss some of the theory of computational speeds of these steps, while in the next we will give some experimental results on two popular software implementations of these.

The fastest available software still requires nearly half an hour on a very fast computer to calculate even just through the second homology of a point cloud with only 2000 points in 3 dimensions. It is thus desirable to try to find shortcuts, by which we can sacrifice a small amount of accuracy for a significant gain in speed. In this Chapter we discuss the theoretical foundations one such technique.

4.1 Pipeline of persistent homology

Let us once again make use of the computational pipeline we have described, focusing this time on the speeds of computation.

1. We may preprocess a point cloud.

2. We obtain a parameterized family of simplicial complexes from this point

cloud, possibly using Čech, Rips, or Alpha complexes, and interpret the parameterization as a filtration.

3. We may reduce the filtered simplicial complex.

4. By taking the homology, we obtain a family of homology groups. By reinterpreting the filtration, we can consider it a parameterized family.

5. We can then display this information in a persistence diagram or a barcode.

4.1.1 Preprocessing

The point cloud may be preprocessed through either sparsification [41, 94] or dimension reduction. Sparsification is computationally very inexpensive, and is discussed in more detail elsewhere. Dimension reduction can also be performed inexpensively—a random projection matrix can be calculated beforehand.

4.1.2 Obtaining a filtered simplicial complex from a point cloud

The naive implementation of the Čech complex involves calculating an enclosing radius for every n-tuple of points and thus scales with the number of subsets of n, 2^n. This is, to put it mildly, computationally infeasible. In practice, therefore, it is almost always preferable to use a complex which is easier to calculate. The mathematical community has settled on the Rips complex, which, as discussed in Chapter 2, is related to the Čech complex and thus has a theoretical accuracy guarantee and, being a clique complex, requires little computation beyond the pairwise distances of the points. As this pairwise distance computation itself scales both

with the square of the number of points and the ambient dimension of the points, dimension reduction is a natural candidate for speeding computation. A previous attempt to cirumvent the "dimensionality curse" was the invention of witness complexes [55]. While these have desireable convergence properties for zero and first dimensional homology, there exist cases where they do not give correct results for higher dimensional homology groups [108].

4.1.3 Reduction of filtered simplicial complex

A filtered simplicial complex can be greatly reduced using discrete Morse theory [98]. This is a well developed area with a great deal of its own literature to support it, and we have nothing further to add to that illustrious tale of mathematical victories—which has recently been added to by Henselman and Ghrist [85] through their use of matroids.

4.1.4 Obtaining a parameterized family of homology groups

The simplest way of obtaining a parameterized family of homology groups is through column reduction of the boundary matrix. This bears a great deal of resemblence to classical matrix row reduction algorithms, and so shares speed estimations with them.[1] Many refinements to this algorithm have been found. For instance, refinements that greatly improve speed include taking advantage of duality to perform easier cohomological calculations rather than homological ones [58, 59], not bothering to reduce columns in the matrix which will not contribute to the final

[1] In particular, for an $n \times n$ matrix over a finite field it is $O(n^3)$ [4].

answer [42], and others. The author has been unable to locate full analyses of the time complexity of the algorithm in its fully-refined splendor; indeed, Ulrich Bauer, who wrote the code Ripser, which is the fastest implementation available, stated in 2017 that, to his knowledge, "there is no asymptotic analysis of the algorithm" [14].

4.1.5 Displaying information in a persistence diagram or barcode

From a parameterized family of homology groups, there is negligible computation time involved in displaying the information. We include this step mostly for completeness, and also as an excuse to point out some of the interesting work done in persistence landscapes [26] and elaborated on with persistence images [1], which makes the information of the persistent homology of an object a much more fertile foundation for the application of machine learning techniques.

4.2 Stability of persistent homology

Recall Theorem 2.4.2, on page 39, which says, roughly speaking, that anything we do to the point cloud that doesn't interfere much with pairwise distances will not interfere much with the persistent homology. We restate it here for convenience:

Theorem 4.2.1 (Cohen-Steiner, Edelsbrunner, Harer [45])**.** *Let X be a triangulable space with continuous tame functions $f, g : X \to \mathbb{R}$. Then the bottleneck distance between the persistence diagrams $D_B(f, g)$ is at most twice the l^∞ distance between f and g:*

$$d_B(D(f), D(g)) \leq 2\|f - g\|_\infty.$$

While they do not state and prove this theorem for Rips complexes, only for the Čech complex, we note (for use in a few pages) their proof goes through with no complications in the Rips complex case.

Note that $\mathbb{R}^n$ is triangulable, and the function which gives distance to the nearest point of a point cloud is both continuous and tame. To ensure tameness, recall that we defined a point cloud to necessarily be not only discrete, but finite.

In this Section, we will prove a closely related theorem, a highly general stability result for the persistent homology of the Rips complex associated to a point cloud. Our proof will only apply to the "naive" view of persistent homology, where the object that has persistent homology is a point cloud, and only to the Rips complex. As an immediate application, we will pair our Theorem 4.2.11 with the Johnson-Lindenstrauss Lemma (Theorem 4.3.1) and obtain a result on preservation of persistent homology through dimension reduction.

The results we show here were inspired by Baraniuk and Wakin's paper on random projection of smooth manifolds [12]. To the best of the authors knowledge, this paper was the first to suggest the application of the Johnson-Lindenstrauss Lemma (Theorem 4.3.1) to persistent homology. After finding our result, we discovered that in so doing, we have overlapped with a result of Don Sheehy [120], who showed that the persistent homology of a point cloud and its distance function are interleaved.[2] His result was strengthened just last year by Arya et al [7], who

[2]We borrow and slightly modify some terminology of his that we have not seen elsewhere for Definition 4.3.2.

gave a result applicable to any linear transformation which preserves distances to within a multiplicative factor of $(1 \pm \epsilon)$. Their results are both stronger than ours in that they consider alternate versions of "distance to a point cloud", where we only discuss Euclidean distance. On the other hand, our result is more general in that it depends on neither linearity of the function, nor any specific form of approximate distance preservation.

The structure of the proof we give here follows Cohen-Steiner, et al's proof of Theorem 2.4.2 very closely. To aid comparison of the two, we borrow much of their language and many of their notational conventions.

4.2.1 Notation

For technical reasons which will become clear, we will not work with point clouds—instead, we will phrase many of our theorems as concerning *weighted graphs*.

Definition 4.2.2 (Weighted graph)**.** A *weighted graph* is a set of points together with a positive number for every pair of points, which we shall think of and refer to as the edge length.

While the edge length acts much like a distance, it is not guaranteed to satisfy the triangle inequality. Any point cloud can be interpreted as a weighted graph, with distance given by the ambient metric of $\mathbb{R}^n$. We will use either $d(x_1, x_2)$ or $\|x_1 - x_2\|$ to refer to the length of the edge between x_1 and x_2—the former when we wish to emphasize the lack of triangle inequality, the latter when we believe the most common applications will be in the case of a weighted graph derived from a

point cloud.

A weighted graph naturally gives rise to a filtered Rips complex, which in turn has persistent homology that we can consider the persistence diagram of. Just as with point clouds, if X is a weighted graph, we use X_i for the subgraph with all edges of length less than or equal to $2i$. (The factor of two is due to the conversion from radii to diameters).

We use $R(X_i)$ to denote the Rips complex associated with X_i. We use $\mathrm{H}_k^i(X)$ to denote $\mathrm{H}_k(R(X_{i/2}))$, the homology of the Rips complex with diameter i with coefficients in a field. Suppressing the k, let β^i denote the dimension of $\mathrm{H}_k^i(X)$.

For $a \leq b \leq c$, let $\mathrm{H}_k^{a\to b}$ be the image of $\mathrm{H}_k^a(X)$ in $\mathrm{H}_k^b(X)$ induced by the inclusion $R(X_{a/2}) \hookrightarrow R(X_{b/2})$. Note that if the interval $[a, b]$ contains none of the pairwise distances between points of X, this map is an isomorphism.

Consider the sequence $\mathrm{H}_k^a(X) \to \mathrm{H}_k^b(X) \to \mathrm{H}_k^c(X)$ induced by inclusions of the Rips complexes. Note that this sequence induces natural maps from $\mathrm{H}_k^{a\to b}$ to $\mathrm{H}_k^{a\to c}$ (a surjection) and from $\mathrm{H}_k^{a\to c}$ to $\mathrm{H}_k^{b\to c}$ (an inclusion). We will use $\beta^{a\to b}$ for the dimension of $\mathrm{H}_k^{a\to b}$.

From here on, we fix a homology dimension k, and suppress it in the notation.

Let $(a_i)_{i=1\ldots n}$ be the set of edge lengths of X, ordered from smallest to largest, and let (b_i) be an interleaved sequence; ie, $b_{i-1} < a_i < b_i$. For any two subscripts, we will refer to the quantity $\mu^{i\to j} := \beta^{b_{i-1}\to b_j} - \beta^{b_i\to b_j} + \beta^{b_i\to b_{j-1}} - \beta^{b_{i-1}\to b_{j-1}}$ as the *multiplicity*.

Suppose we have four real numbers, none of them in (a_i), and ordered $w < x < y < z$. There is a natural surjection, $\mathrm{H}^{x\to y}(X) \twoheadrightarrow \mathrm{H}^{x\to z}(X)$. With apologies for

overloading our notation, and keeping in mind that we are suppressing the subscript k indicating homological dimension, we will write $\mathrm{H}_x^{y,z}$ for the kernel of this surjection; that is, $\mathrm{H}_x^{y,z}$ is elements of H_x which die between times y and z. There is an inclusion $\mathrm{H}_w^{y,z} \hookrightarrow \mathrm{H}_x^{y,z}$. This is because any element of H_w which dies between times y, z must still be alive at time x. We will thus use $\mathrm{H}_{w,x}^{y,z}$ to represent the quotient $\mathrm{H}_x^{y,z}/\mathrm{H}_w^{y,z}$.

We will write $D(X)$ for the persistence diagram associated with the Rips complex of X, filtered by edge length in the obvious way. This is the multiset of points in $\mathbb{R}^2$, where (a_i, a_j) has multiplicity $\mu^{i\to j}$, together with all points on the diagonal $\Delta := (x, x)$ counted with infinite multiplicity. We will refer to the sum of the multiplicities of the off-diagonal points in any region of the persistence diagram as the *size* of that portion of the persistence diagram. The size of the entire diagram is thus $\#(D(X) - \Delta) := \sum_{i<j} \mu^{i\to j}$. The dimension of $\mathrm{H}_{w,x}^{y,z}(X)$ is exactly the size of the portion of $D(X)$ in a box $\#(D(X) \cap [w, x] \times [y, z])$.

We will refer to the quadrant of $\mathbb{R}^2$ above and to the left of (x, y) by $Q_x^y := (-\infty, x] \times [y, \infty)$.

Lemma 4.2.3 ($k-$triangle lemma of Cohen-Steiner, Edelsbrunner, Harer [45])**.** *Suppose we have a weighted graph X and two values $x < y$, with neither x nor y being pairwise distances between points of X. Then $\#(D(X) \cap Q_x^y) = \beta^{x\to y}$.*[3]

[3]As stated originally, the theorem is more general; it concerns tame functions on any space, while what we do here could be interpreted as piecewise linear functions on complete graphs.

4.2.2 Quadrant lemma and box lemma

Throughout this Section, suppose we have a bijection between weighted graphs, $\phi : X \to X''$ which approximately preserves edge lengths in the sense that there are *bounding functions* for ϕ: continuous, monotonic, invertible functions $L(d), U(d) : \mathbb{R}^+ \to \mathbb{R}^+$ with $L(d) \leq d \leq U(d)$ with, for any $x_1, x_2 \in X$, $L(\|x_1 - x_2\|) \leq \|\phi(x_1) - \phi(x_2)\| \leq U(\|x_1 - x_2\|)$. We will identify the points of X with their images in X''. Equivalently, we may instead think of X and X'' as two sets of edge lengths assigned to a single set of vertices.

Note that any statements we make will have mirror statements with the roles of X and X'' reversed, U replaced by L^{-1}, and L replaced by U^{-1}. We will refer to this as the "mirror principle".

Proposition 4.2.4. *Given two reals, $b < c$, we have the inclusions*

$$R(X''_{L(b)/2}) \hookrightarrow R(X_{b/2}) \hookrightarrow R(X_{c/2}) \hookrightarrow R(X''_{U(c)/2}).$$

Proof. In order for a set of points $\delta \subset X$ to not form a simplex in $R(X_{b/2})$, there must be at least one pair of points, $x_1, x_2 \in \delta$ with $d(x_1, x_2) > b$. By definition of L, $d(\phi(x_1), \phi(x_2)) > L(d(x_1, x_2)) > L(b)$, so the corresponding set of points do not form a simplex in $R(X''_{L(b)/2})$, establishing the first inclusion. The second is trivial. The third inclusion follows from the applying the mirror principle to the first inclusion. □

These inclusions of Rips complexes induce maps on homology groups:

$$\mathrm{H}^{L(b)}(X'') \to \mathrm{H}^{b}(X) \to \mathrm{H}^{c}(X) \to \mathrm{H}^{U(c)}(X'') \tag{4.1}$$

Induced by similar inclusions, we also have

$$\mathrm{H}^{b}(X) \to \mathrm{H}^{U(b)}(X'') \to \mathrm{H}^{U(c)}(X''). \tag{4.2}$$

As all the relevant maps on the level of Rips complexes are inclusions, the composition of maps in (4.2) is equivalent to the composition of the latter two maps in (4.1).

We also have the inclusions

$$\mathrm{H}^{L(b)\to U(c)}(X'') \subset \mathrm{Im}(\mathrm{H}^{b}(X) \to \mathrm{H}^{c}(X) \to \mathrm{H}^{U(c)}(X'')) \subset \mathrm{H}^{U(b)\to U(c)}(X''), \tag{4.3}$$

where the first inclusion comes from (4.1), and the second from (4.2).

Lemma 4.2.5 (Quadrant lemma)**.**

$$\#(D(X'') \cap Q^{U(c)}_{L(b)}) \leq \#(D(X) \cap Q^{c}_{b})$$

Proof. This is exactly the statement that the dimension of the image of the middle map of (4.1) is greater than or equal to the dimension of the image of the composition of all three maps. □

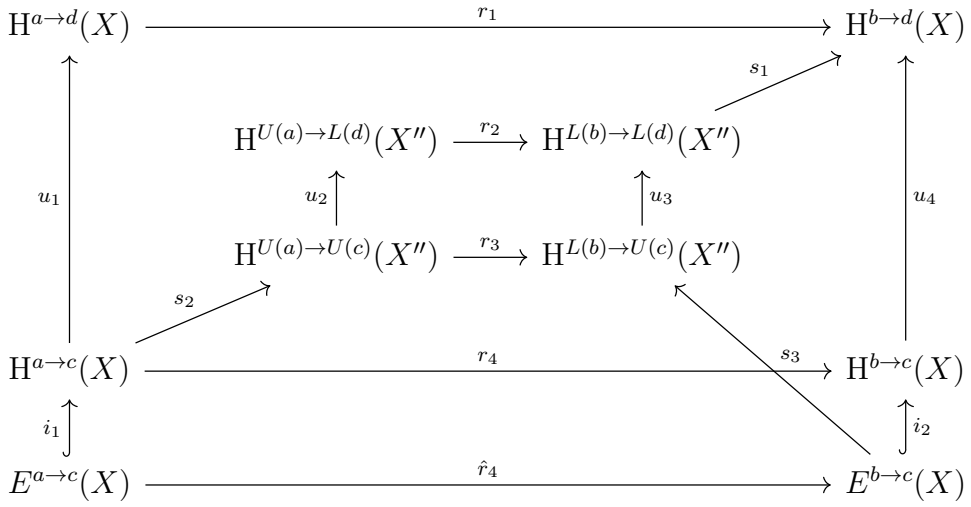
Lemma 4.2.6 (Box lemma)**.** $\#(D(X'') \cap [U(a), L(b)] \times [U(c), L(d)]) \leq \#(D(X) \cap$

$[a,b] \times [c,d]$).

Proof. Note that we may assume order is preserved; $U(a) < L(b) < U(c) < L(d)$, otherwise there is nothing to prove, as the smaller box becomes the empty set.

It suffices to find a surjection from a subspace of $\mathrm{H}^{c,d}_{a,b}(X)$ onto $\mathrm{H}^{U(c),L(d)}_{U(a),L(b)}(X'')$, as the dimensions of these vector spaces are exactly the quantities in the inequality.

Consider the following commutative diagram:



The vertical maps labeled u_i are all natural maps as discussed in the previous Section, as are the horizontal maps labeled r_i (the r_i maps are, in fact, inclusions, while the u_i are surjections).

We know there is a map from $\mathrm{H}^c(X)$ to $\mathrm{H}^{U(c)}(X'')$. We define $E^{b\to c}$ to be the preimage of the kernel of u_3 under this map, intersected with the subgroup $\mathrm{H}^{b\to c}(X)$. By construction, we therefore have a map, which we call s_3, from $E^{b\to c}$ to $\mathrm{H}^{L(b)\to U(c)}(X'')$, with $\operatorname{Im} s_3 \subset \ker u_3$. The only requirements for being in the domain

of s_3 are mapping to the kernel of u_3 and being in $H^{b->c}(X)$. By the first inclusion in (4.3), everything in the domain of u_3 comes from something in $H^{b->c}(X)$ so in fact we have $\operatorname{Im} s_3 = \ker u_3$. We define $E^{a\to c}$ by intersecting this further with $\mathrm{H}^{a\to c}(X)$. The vertical i maps are thus inclusions. The map s_2 comes from the inclusion in (3). s_1 comes from the inclusion obtained by the "mirror principle" mentioned earlier.

As all the maps are natural, the diagram commutes. Thus $u_4 \circ i_2 = s_1 \circ u_3 \circ s_3$. By construction, $u_3 \circ s_3$ is zero, so $u_4 \circ i_2$ is as well. We have a subspace inclusion $E^{b\to c}/E^{a\to c} \subset \mathrm{H}^{c,d}_{a,b}(X)$.

We will now show that $\dim \mathrm{H}^{U(c),L(d)}_{U(a),L(b)}(X'') \leq \dim E^{b\to c}/E^{a\to c}$, which will complete the proof. Recall that $\mathrm{H}^{U(c),L(d)}_{U(a),L(b)}(X'') = \ker u_3/\ker u_2$. By construction of the E spaces, and the fact that the r_i maps are inclusions of subspaces, we have $E^{b\to c}/E^{a\to c} = \ker u_4/\ker u_1$. By construction, s_3 is a surjection of $\ker u_4$ onto $\ker u_3$. If we can show that $s_2(\ker u_1)$ is a subspace of $\ker u_2$, we will be done. For any $h \in \ker u_1$, we have $u_3 \circ s_3 \circ \hat{r}_4(h) = 0$, because $s_3(E^{b\to c})$ lies entirely inside the kernel of u_3. By commutativity of the diagram, $r_2 \circ u_2 \circ s_2(h)$ is also 0. As r_2 is an inclusion map, $u_2 \circ s_2(h) = 0$, so we are done. □

Corollary 4.2.7. *If there is a point of the persistence diagram of X at (a, b) with multiplicity μ, then the total multiplicity of the points of the persistence diagram of X'' within the rectangle $[L(a), U(a)] \times [L(b), U(b)]$ is at least μ.*

Proof. Immediate application of box lemma. □

By our mirror principle, we also have:

Corollary 4.2.8. *If there is a point of the persistence diagram of X'' at (a', b') with multiplicity μ, then the total multiplicity of the points of the persistence diagram of X within the rectangle $[U^{-1}(a'), L^{-1}(a')] \times [U^{-1}(b'), L^{-1}(b')]$ is at least μ.*

At this point, we have found a map from the points of the persistence diagram of X to those of the persistence diagram of X'' which doesn't move points "very far". If we can strengthen this map to be a bijection, we will be done. We start with an easy case; if the bijection of the weighted graph doesn't change edge lengths much, then maybe every resulting rectangle only has the minimum number of points necessary for a bijection. In order to formalize this, we make the following:

Definition 4.2.9 (Resolution of a persistence diagram)**.** Given a weighted graph X, the *k-dimensional resolution of the persistence diagram of X* is the minimum distance between two points of the persistence diagram for the k-dimensional homology, at least one of which is off the diagonal. Denoting this by $\delta_{X,k}$, we have

$$\delta_{X,k} := \min\{\|p_1 - p_2\|_\infty \mid p_1 \in D_k(X), p_2 \in D_k(X) - \Delta\}.$$

Taking the minimum over all k, we obtain the *resolution of the persistence diagram of X*:

$$\delta_X := \min \delta_{X,k}.$$

Another way of considering resolution: if every point of $D(X)$ has a square with side length δ_X centered on it, then no two squares will intersect, unless they are both centered on the diagonal.

We have the following:

Lemma 4.2.10 (Easy bijection lemma of Cohen-Steiner, Edelsbrunner, Harer [45])**.** *Suppose $\phi : X \to X''$ is a bijection of weighted graphs which does not change the pairwise distances by more than the resolution of X; ie, $L(d) = d-\epsilon$ and $U(d) = d+\epsilon$ are bounding functions for ϕ, with $\epsilon < \delta_X$.*[4] *Then there is a bijection $\psi : D(X) \to D(X'')$ with $\|p - \psi(p)\|_\infty \leq \epsilon$ for all $p \in D(X)$.*

Proof. From our Corollaries 4.2.7 and 4.2.8, if μ is the multiplicity of a point of $D(X)$ located at (a, b), we have

$$\mu \leq \#(D(X'') \cap [a-\epsilon, a+\epsilon] \times [b-\epsilon, b+\epsilon]) \leq \#(D(X) \cap [a-2\epsilon, a+2\epsilon] \times [b-2\epsilon, b+2\epsilon]).$$

As the resolution of $D(X)$ is larger than ϵ, the point at (a, b) is the only point of $D(X)$ within this rectangle, so we have a (local) bijection. Iterating this over every off-diagonal point of $D(X)$, the only way for a point of $D(X'')$ to remain unmatched is if it is within ϵ of the diagonal. Matching all such points with their nearest point on the diagonal, we have thus constructed a bijection which moves no point further than ϵ, and so are done. □

Earlier, we mentioned we could take the point of view of X and X'' as two different assignations of edge lengths to the same underlying point set. We now take that viewpoint, and put an entire family of edge lengths on this point set.

[4] In fact, these are not properly bounding functions as we have defined them, because they are not invertible functions $\mathbb{R}^+ \to \mathbb{R}^+$; $d + \epsilon$ is not onto, thus not invertible, and $d - \epsilon$ has a codomain which is slightly too large. As our point clouds are finite sets, we take any bounding function which has the correct values at all pairwise distances. A bijection will never cause the distance between two points to become non-positive, so this creates no issues.

Letting c be the maximum change in edge length, we denote the weighted graph with accompanying function by X^t, with $t \in [0, c]$, where the edge length between x_1 and x_2 in X^t is given by

$$\text{length} = \begin{cases} \min\{(\|x_1 - x_2\| + t), \|\phi(x_1) - \phi(x_2)\|\}, \|\phi(x_1) - \phi(x_2)\| > \|x_1 - x_2\| \\ \max\{(\|x_1 - x_2\| - t), \|\phi(x_1) - \phi(x_2)\|\}, \|\phi(x_1) - \phi(x_2)\| < \|x_1 - x_2\|. \end{cases}$$

The X^ts form a continuous family of spaces transitioning from $X^0 = X$ to $X^c = X''$. We will divide this into intervals which are small enough for us to use Lemma 4.2.10 on each interval, and concatenate the bijections.

We are now in a position to prove

Theorem 4.2.11 (Generalized stability theorem)**.** *Let X be a point cloud, and let $\phi : X \to \mathbb{R}^n$ be a map which approximately preserves distances, in the sense that there are continuous, monotonic, invertible functions $L(d), U(d) : \mathbb{R}^+ \to \mathbb{R}^+$ with $L(d) \leq d \leq U(d)$ such that for any $x_1, x_2 \in X$ we have $L(\|x_1 - x_2\|) \leq \|\phi(x_1) - \phi(x_2)\| \leq U(\|x_1 - x_2\|)$. Then there exists a bijection ψ between the points of the persistence diagrams of the Rips complexes associated to X and $\phi(X)$ which satisfies the following condition: if (a, b) is a point of the persistence diagram of X, and $\psi(a, b) = (a', b')$ a point of the persistence diagram of $\phi(X)$, then $L(a) \leq a' \leq U(a)$ and $L(b) \leq b' \leq U(b)$.*

Existence of a bijection. Note that the resolution of the persistence diagrams associated to the X^ts are always positive numbers, δ_{X^t}. The set of open intervals $J_t = (t - \frac{\delta_{X^t}}{4}, t + \frac{\delta_{X^t}}{4})$ is thus an open cover of the closed interval $[0, c]$. By compact-

ness, it has a finite minimal subcover. Let $t_1 < t_2 < \cdots < t_n$ be the midpoints of the intervals in some minimal subcover. By minimality, any two consecutive intervals will have non-empty intersection, so we have

$$t_{i+1} - t_i \leq (\delta_{X^{t_i}} + \delta_{X^{t_{i+1}}})/4 \leq \max\{\delta_{X^{t_i}}, \delta_{X^{t_{i+1}}}\}/2.$$

We thus have a family of bijections, which we shall label $\phi_i : X_i^t \to X_{i+1}^t$. These each have bounding functions $U_i(d) = d + (t_{i+1} - t_i)$ and $L(d) = d - (t_{i+1} - t_i)$. By construction, this is less than the resolution of the relevant persistence diagram, so by Lemma 4.2.10 there is a bijection $\psi_i : D(X^{t_i}) \to D(X^{t_{i+1}})$ with $\|x - \psi_i(x)\|_\infty \leq t_{i+1} - t_i$.[5] Chaining all these bijections together, we obtain a bijection between the points of $D(X)$ and those of $D(X')$. □

Analysis of distance traveled. How far does this bijection move a point of $D(X)$ which begins at (a, b)? To check this, we decompose the bijections into two steps: let X' be Rips complex where the edge length $d(x_1, x_2)$ is given by the minimum of their edge length in X and their edge length in X'', and let X'^t be the continuously varying family of weighted graphs between X and X', where the edge length between x_1, x_2 in X'^t is given by

$$\text{length} = \begin{cases} \|x_1 - x_2\|, \|\phi(x_1) - \phi(x_2)\| > \|x_1 - x_2\| \\ \\ \max\{(\|x_1 - x_2\| - t), \|\phi(x_1) - \phi(x_2)\|\}, \|\phi(x_1) - \phi(x_2)\| < \|x_1 - x_2\|. \end{cases}$$

[5] Because we are taking the max of the two δ's, we don't know if we have a bijection from $D(X_t) \to D(X_{t+1})$ or vice-versa. But it's a bijection, so this is fine.

We thus have $X'^0 = X$ and $X'^1 = X'$. With the same set of t_i's as before, we label these contracting bijections $\phi'_i : X'^{t_i} \to X'^{t_{i+1}}$. What are the bounding functions for these ϕ_i? The edge lengths are never increasing, so we may take $U_i(d) = d$. For $L_i(d)$, our global bound has improved by $t_{j+1} - t_j$ after each step, or t_i total. Obviously, no lower bounding function can be above the identity, so we have

$$L_i(d) = \begin{cases} d, & L(d) + t_i > d \\ \max\{L(d) + t_i, d - (t_{i+1}, t_i)\} & \text{else.} \end{cases}$$

By Corollary 4.2.7, each of the bijections may only move the point(s) at (a, b) left and down. The movement is guaranteed to stop at or before the time when $L(d) + t_i = d$, so the composition of the bijections results in a point beginning at (a, b) ending within the rectangle $[L(a), a] \times [L(b), b]$.

The same argument applies to expanding X' to X'', moving points up and to the right, where they end within the rectangle $[L(a), U(a)] \times [L(b), U(b)]$. □

This theorem resembles, but is not identical to, the stability theorem of Cohen-Steiner, Edelsbrunner, and Harer (Theorem 4.2.1). Their theorem allows us to take persistent homology of many kinds of functions, not merely functions which give distance to a point cloud, and our Theorem 4.2.11 here is restricted to that case. In that case, our theorem gives a result which mirrors theirs if we let $L(t) = t - \epsilon$, $U(t) = t + \epsilon$. As our theorem concerns Rips complexes, and theirs concerns Čech complexes, this mirroring is not a direct overlap.

While there may be some concern that the Čech complex is more an object

of interest than the Rips complex, in practical applications involving point clouds, almost all calculations are done using the Rips complex or approximations or sparsifications of it. Indeed, Otter et al.'s paper [107] lists seven computational topology software packages which compute persistent homology from a point cloud. Six of the seven have the Rips complex as as option—for the most state-of-the-art, it is the only option—and no other complex has more than three software packages. This theorem is thus more directly applicable than one which concerns the Čech complex itself.

Also note that if $U(t)$ becomes very large after some point—eg, if we have

$$U(t) = \begin{cases} t(1+\epsilon), & x \leq c, \\ \infty, & x > c, \end{cases}$$

then, although we have no guarantees about persistent homology classes which come into existence after c, any that began existing before c will still exist, and will not have shrunk "too much". Similarly, if we have bounds which are relatively tight after c, but very loose beforehand (perhaps we have both a source of noise we wish to model as a perturbation of up to a constant magnitude, and a distortion that could affect the pairwise distances, a distortion we are modeling as being bounded by a constant multiple), we may have better guarantees about any persistent homology classes which die after c than those which die early.

4.3 Using stability for linear dimension reduction

In addition to the theoretical interest, this result will allow us to speed up the computations of persistent homology. For a given point cloud, it isn't obvious how perturbing the points will lead to swifter computations.[6] However, reducing the ambient dimension of the point cloud can frequently speed up computations. If the number of points in X is higher than the ambient dimension (which it almost always is), we don't expect to embed X into a lower dimension perfectly isometrically. However, with high probability we can still do so approximately, by the following:

Theorem 4.3.1 (Johnson-Lindenstrauss Lemma [54, 87])**.** *Let X be a point cloud in $\mathbb{R}^D$, and fix $0 < \epsilon < 1$, and*

$$d \geq \frac{4}{\epsilon^2/2 - \epsilon^3/3} \ln(\#X).$$

There exists a linear map $\phi : \mathbb{R}^D \to \mathbb{R}^d$ which approximately preserves distances, in the sense that for every $x, y \in X$ we have

$$(1-\epsilon)\|x-y\|^2 \leq \|\phi(x) - \phi(y)\|^2 \leq (1+\epsilon)\|x-y\|^2.$$

Note that by taking square roots, we obtain the inner inequalities of the fol-

[6] Indeed, to the best of the author's knowledge, in general it won't.

lowing

$$(1-\epsilon)\|x-y\| \leq \sqrt{1-\epsilon}\|x-y\| \leq \|\phi(x)-\phi(y)\| \leq \sqrt{1+\epsilon}\|x-y\| \leq (1+\epsilon)\|x-y\|,$$

where the outer inequalities are true because ϵ is less than 1, and squaring decreases positive numbers less than 1 and increases those greater than 1.

Definition 4.3.2 (ϵ-JL projection for X)**.** An *ϵ-JL projection of a point cloud X* is a linear function $\phi : \mathbb{R}^D \to \mathbb{R}^d$ such that any two points x_1, x_2 of X satisfy

$$(1-\epsilon)d(x_1,x_2) \leq d(\phi(x_1),\phi(x_2)) \leq (1+\epsilon)d(x_1,x_2).$$

While there exist versions of Theorem 4.3.1 that include information on the likelihood of a projection onto a random subspace preserving distances, we will not make use of that. We will instead take an ϵ-JL projection as given.

Pairing the two theorems seems natural—we have one which states that dimension reduction doesn't shift distances very much, and another which states that if the distance function doesn't change much then neither does the persistent homology.

Corollary 4.3.3. *Let X be a point cloud in $\mathbb{R}^D$, and let ϕ be an ϵ-JL projection. If h is an element of the persistence diagram of the Rips complex associated to X with birth time t_b and end time t_e, then there is an element of $D(\phi(X))$ with birth time in $[t_b(1-\epsilon), t_b(1+\epsilon)]$ and end time in $[t_e(1-\epsilon), t_e(1+\epsilon)]$.*

Proof. Follows from Theorem 4.2.11 using $L(t) = t(1-\epsilon)$ and $U(t) = t(1+\epsilon)$. □

Corollary 4.3.3 is of interest for two reasons. From one point of view, the fact that random projection has a high probability of preserving persistent homology serves as validation of random projection as a dimension reduction scheme. In a much more practical way, this gives us a method by which we might hope to reduce the computation times of persistent homology. Chapter 5 takes the second perspective, and we will evaluate its usefulness there.

4.4 Other dimension reduction methods

The above property of approximately preserving distances is a desirable one in any dimension reduction method, so we may expect many dimension reduction methods to generally preserve persistent homology, not just the reduction of "project onto a random subspace". As an example, we consider the following dimension reduction method, which we will refer to as Grammian reduction. Essentially equivalent to principal component analysis, or PCA, it is a prototypical example of a dimension reduction method, and the forefather of basically all spectral methods of dimension reduction [77, 117].

1. Given a point cloud X with $|X| = n$, we construct the Gram matrix, G, whose $(i,j)^{\text{th}}$ entry is the inner product $< x_i, x_j >$. G is, of course, $n \times n$.

2. As the Gram matrix is symmetric, the spectral theorem tells us it factors as $G = QDQ^T$, where Q is an orthogonal matrix and D is diagonal. Again, Q

and D are $n \times n$. Without loss of generality, we order the eigenvalues of D and the columns of Q so that $i < j \implies D_{i,i} \leq D_{j,j}$.

3. We can truncate D into a smaller $k \times k$ matrix, $\hat{D}$, which contains just the largest k eigenvalues, and similarly truncate Q into an $n \times k$ matrix $\hat{Q}$.

4. Observe that if we let $\hat{X} = \hat{D}^{1/2}\hat{Q}^T$, then the columns of $\hat{X}$ form a set of n points inside $\mathbb{R}^k$, and their Gram matrix $\hat{G} = \hat{X}^T\hat{X} \approx G$, so the columns of $\hat{X}$ are a family of points whose pairwise distances is approximately equal to the pairwise distances between elements of X.

Theorem 4.4.1. *Suppose X is a point cloud with n points, and Gram matrix $G = QDQ^T$, where Q is orthogonal and D is diagonal, with values $D_{i,i} = \lambda_i$ and $\hat{X}$ is a point cloud with Gram matrix $\hat{G} = Q\hat{D}Q^T$, and $\hat{D}_{i,i} := \lambda_i$ for $i \leq k$, and $\hat{D}_{i,i} := 0$ for $i > k$. If h is an element of the persistence diagram of the Rips complex associated with X with birth time t_b and death time t_d, the persistence diagram for the Rips complex of $\hat{X}$ has an element with birth time t_α and death time t_ω , which satisfy*

$$t_b - 2\sqrt{\sum_{i=k+1}^{n} \lambda_i} \leq t_\alpha \leq t_b + 2\sqrt{\sum_{i=k+1}^{n} \lambda_i} \text{ and } t_d - 2\sqrt{\sum_{i=k+1}^{n} \lambda_i} \leq t_\omega \leq t_d + 2\sqrt{\sum_{i=k+1}^{n} \lambda_i}.$$

Proof. The original point cloud is isometric to the columns of $D^{1/2}Q^T$, so we can view this as a perturbation moving each point by a succession of orthogonal moves. As Q is orthogonal, each movement is by at most the eigenvalue being eliminated. Changing any two eigenvalues results in two movements orthogonal to each other,

so the total magnitude of the movement is, by the Pythagorean Theorem, bounded by the squareroot of the squares of the values of $D^{1/2}$ other words, by the squareroot of the sum of the eigenvalues. Thus the distances move by at most twice that, and so, by Theorem 4.2.11, this Grammian reduction approximately preserves persistent homology. □

While the entire theory of dimension reduction techniques is far too large to go into here, we wish to emphasize that any theoretical guarantee of distance preservation on *any scale* translates directly via Theorem 4.2.11 to a theorem on the preservation of persistent homology. ISOMAPS, Locally Linear Embeddings, Laplacian Eigenmaps, Schrödinger eigenmaps, and diffusion mapping are all techniques which have either local distance convergence guarantees or strong reasons to expect such to be true, thus we expect there to be a version of Theorem 4.4.1 for each of them. New dimension reduction methods are proposed frequently; see Njeunje [105] for an example. We hope this proof serves as a template for future proofs for many dimension reduction techniques.

4.5 Directions for further research

Theorem 4.2.11 as we have proven it applies only to Rips complexes. If I is a subset of points in X, then let d_I be the radius of the smallest ball enclosing all points in I. If, at each of the d_I times, we have approximate pairwise distance preservation, in the sense of Theorem 4.2.11, then a Čech complex version of our result should hold.

If the $U(t)$ and $L(t)$ functions are replaced by a single probability distribution, it seems clear that a probabilistic verstion of Theorem 4.2.11 should still obtain. Moreover, it seems intuitively clear that an element of persistent homology with many representatives should be more stable, and while the deterministic version of our result cannot take advantage of a many-representative situation, such a probabalistic version should be able to. As we will see in the next Chapter, the extent to which persistent homology is preserved under random projection is higher than the theory might lead us to expect.

Chapter 5: Experimental results on computation times of persistent homology

In the previous Chapter, we showed that persistent homology can be preserved through reduction of the ambient dimension. A natural question to now explore is how does the speed of calculating the persistent homology of a point cloud scale with ambient dimension in practice? Is Corollary 4.3.3 of practical significance, or merely theoretical interest?

While there has been a great deal of work in speeding up the computation of persistent homology (see our Section 1.2 for some examples), the author is not aware of many tests of the practical effect of these in the literature. We thus take this opportunity to fill in the gap by running experiments intended to establish how time to calculate persistent homology scales with several other variables of interest and make some conjectures based on the results we obtain. We hope these serve to guide future research.

5.1 Datasets and software used for benchmarks

We use datasets from [107], which have been put forward as a standard collection of benchmarks for persistent homology speed computations and since been used

Dataset denotation	Filename on github	# Points	Dim
Celegans	celegans_weighted_undirected_reindexed_for _matlab.txt_maxdist_2.6429_SP_distmat.txt _point_cloud.txt	297	202
Dragon1000	dragon_vrip.ply.txt_1000_.txt	1000	3
Dragon2000	dragon_vrip.ply.txt_2000_.txt	2000	3
Fractal	fractal_9_5_2_linear_edge_list.txt _1866.1116_point_cloud.txt	512	257
HIV1	HIV1_2011.all.nt.concat.fa_hdm.txt _point_cloud.txt	1088	673
House	house104_edge_list.txt_0.72344_point_cloud.txt	445	261
Human	human_gene2_sampled_reindexed_for_matlab.txt _maxdist_91.9097_SP_distmat.txt_point_cloud.txt	1397	688
Klein	klein_bottle_pointcloud_new_400.txt	400	3
Network	network379_edge_list.txt_38.3873_point_cloud.txt	379	300
Random	random_point_cloud_50_16_.txt	50	16
Senate	senate104_edge_list.txt_0.68902_point_cloud.txt	103	60
Vicsek	Vicsek__particles_300_distance_1_noise _0.1_v0_0.03 _box_25_timestep_1500_of_3000.txt	300	3

Table 5.1: List of datasets taken from Nina Otter's github page [106] and used for experiments in this chapter.

for that in many places (e.g. Morozov and Nigmetov [100] or just last year in Pérez et al. [110]). We use two software packages, Dionysus 2 [99] and Ripser [15, 123].[1]

5.1.1 Datasets

For ease of replicability of results, we give in Table 5.1 the correspondance between the longer file names used by Otter et al. in [107] and graciously made available at [106], and the shorter names which we shall use to refer to the datasets. We also include images of their first two persistent homology diagrams in Figure 5.1. While the datasets appear relatively unstructured, there are still things to note.

[1]The original Ripser package was written by Ulrich Bauer in C++. The version we use here is a python adaptation, written by Christopher Tralie, Nathaniel Saul, and Rann Bar-On.

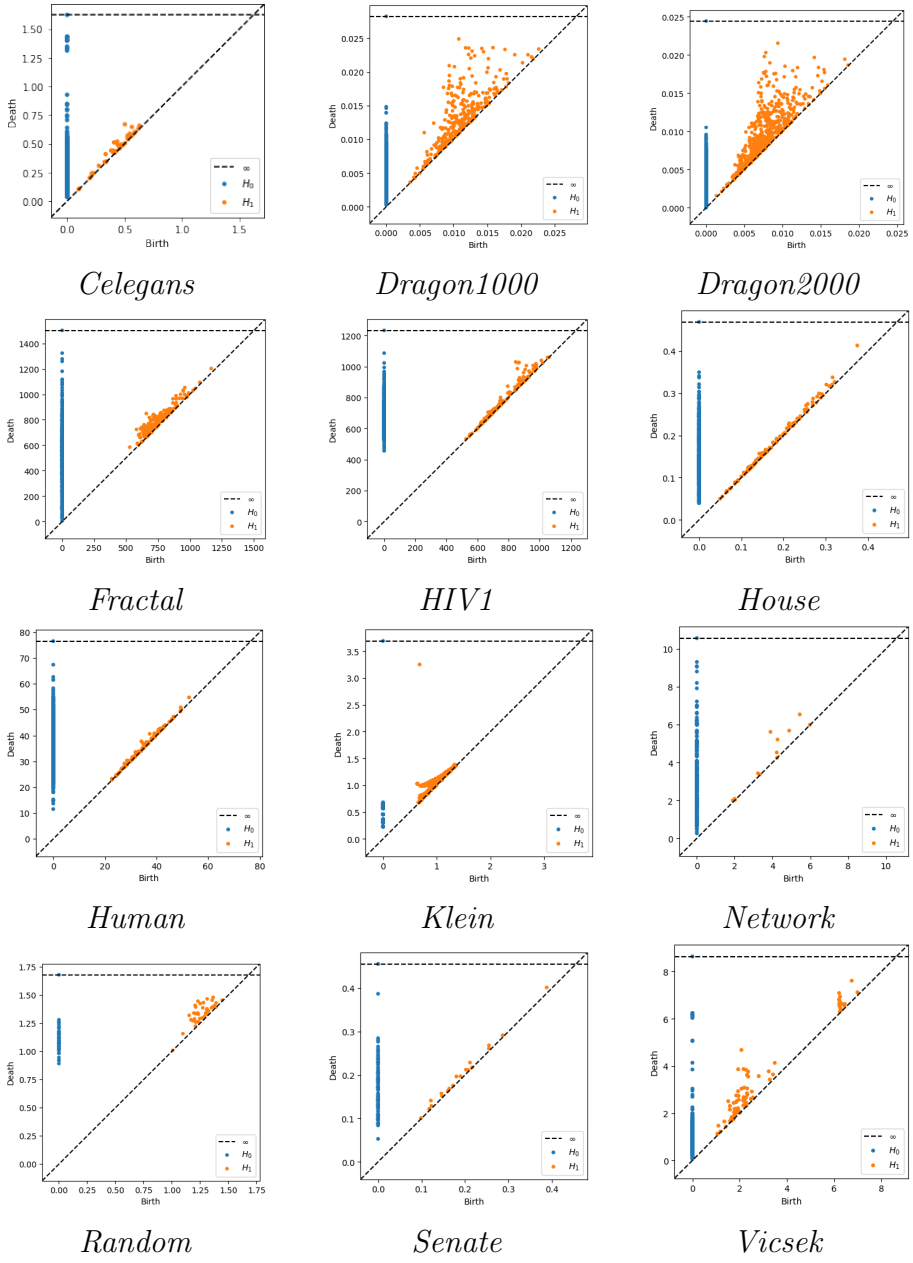


Figure 5.1: Persistence diagrams for the datasets in H_0, H_1. Note the expected extremely high persistence element of H_1 in **Klein**, as the Klein bottle has non-trivial 1st homology. Also note the high isolated element of H_0 in **Senate**—this could indicate a senator or small group of senators with very distinctive voting behaviour, or (more likely) the Democratic/Republican split.

1. **Celegans** is a neural net derived from the nervous system of the roundworm. Each point is a neuron, and the distances between the points are given by the inverse of the weight between the neuronal connections.

2. **Dragon1000** and **Dragon2000** consist of points on the surface of a statue of a small dragon.

3. **Fractal** is a synthetic self-similar network. Due to the self-similarity, note that the H_0's are very evenly distributed

4. Each point of **HIV1** is a genomic sequence sampled from HIV, with distances given by the Hamming distance. As Hamming distances tend to be large, there are no low-persistence points of H_0—the points are fairly isolated from each other.

5. **House** and **Senate** represent the voting behaviour of the 104^{th} US Congress. Nodes are more or less similar depending on how often they voted together. Note the high isolated element of H_0 in **Senate**—this could indicate a senator or small group of senators with very distinctive voting behaviour, or (more likely) the Democratic/Republican split. The lack of a similar point in **House** is due to the higher number of "moderate" representatives.

6. Each point of **Human** is a human gene, with edge lengths given by correlation level of expression of corresponding genes.

7. **Klein** is a synthetic data set sampled from a Klein bottle immersed into $\mathbb{R}^3$. Hence, the presence of a very highly persistent element of H_1.

8. Each node of **Network** is a scientist publishing papers on networks. Distances are given by inverses of number of collaborations in publication history.

9. **Random** is a synthetic dataset consisting of random points drawn independantly from a uniform distribution on a unit hypercube. The relatively tight clustering of the H_0 points is due to the infamous "curse of dimensionality"—distances between random points become more similar as the ambient dimension increases.

10. **Vicsek** is a synthetic dataset modeling points contained in a box.

We might intuitively expect that, given a list of genes, if the expression of each is well-correlated with the expression of the adjacent genes on the list, the expressions of any two genes on the list should be well-correlated. Similarly, we may expect a similar pattern with voting records. We see this expectation fulfilled in the relative flatness of the H_1 diagrams for the **Human**, **Senate**, and **House** datasets. We have no such expectation for points on the surface of a statue, and this is reflected in the cloud of off-diagonal points in the two **Dragon** datasets.

These datasets represent a wide cross section of some of the uses of persistent homology. They contain both real-world and synthetic data, both naturally geometric and non-geometric data. Many of them have been used in published works using persistent homology to study other fields (See [107] for more details). They thus represent a good cross-section for us to study the practical aspects of computing persistent homology. In the remainder of this chapter, we will measure the speed of performing persistent homology calculations with these datasets.

5.1.2 Software

We benchmark against two sets of software to avoid any software-package specific effects. We use Ripser, as it is the fastest available software, and Dionysus as it is one of the easiest to use. Dionysus represents a relatively straightforward implementation of the standard algorithms, and was written/is maintained by Morozov, who is responsible for many of the original results in the field of computational topology. Ripser takes advantage of many adaptations proposed earlier.[2] The author has been unable to locate in the literature a detailed theoretical breakdown of how much each of these speeds up the computations, although Bauer has some experimental results in [15]. As used within this chapter, each takes in a list of lists of floating point numbers, together with a maximum homology dimension, and returns the persistent homology of the point cloud for that dimension and all lower.

5.2 Experiments for a baseline of persistent homology computation speed

We expect the calculation speed of persistent homology to scale with several factors. Most obviously, the speed of the calculation should depend on the number of points in the point cloud, and whether we calculate just H[0], or H[0] and H[1], and so forth. Moreover, we might expect it to depend on the intrinsic dimension of the point cloud–the smallest $\mathbb{R}^d$ which the point cloud isometrically embeds into–as

[2]The Ripser paper itself [15] has a long section detailing influences and giving a geneology of the major improvements, which we will not replicate here.

well as the ambient dimension.[3] In this section, we run experiments attempting to isolate the practical effect of each of these.

5.2.1 Scaling with number of points

A fully naive algorithm calculating $\mathrm{H}_0, \mathrm{H}_1, \mathrm{H}_2$ for a point cloud with n points would calculate the birth times of all 3-simplices, which requires roughly $\binom{n}{4}$, which is $O(n^4)$ simplices, followed by matrix reduction on the square matrix, which is (# of simplices) $\times$ (# of simplices), and as mentioned earlier, the simplest matrix reduction algorithm on an $n \times n$ matrix is $O(n^3)$. The two programs we are using here are far from naive, so we expect them to do better than the implied $O(n^{12})$ bound, and they do.

Table 5.2 shows the results of an experiment to check the scaling of Dionysus computation times as number of points increases.

We plot the best fit quartic in Figure 5.2. Interestingly, the residual of the best-fit quartic for the final point is very small, suggesting that the best-fit quartic for the initial nine points was successfully predicting the tenth point—a good sign for our curve.

We run the same experiment with Ripser (Table 5.3 and Figure 5.3) and obtain substantially the same result. While Ripser is much, much faster, it doesn't scale differently. This is perhaps unsurprising; several of Ripser's improvements may be expected to reduce the number of simplices for which calculations need to be

[3]Indeed, if the speed doesn't depend on ambient dimension, our result of the previous chapter will not help us in speeding it up.

# of points	mean	min	max	std dev
5	0.00005	0.00004	0.00013	0.00002
8	0.00023	0.00022	0.00027	0.00001
11	0.001	0.0009	0.0011	0.0001
17	0.009	0.007	0.015	0.001
25	0.048	0.043	0.055	0.003
38	0.355	0.308	0.504	0.049
57	2.366	2.08	2.888	0.2
85	17.437	15.44	21.061	1.269
128	118.477	107.598	139.04	7.827
192	849.037	754.377	961.577	51.48
288	6250.355	-	-	-

Table 5.2: Dionysus computation of subsamples of Dragon2000 dataset. Each row except the last of this table is a set of 20 runs of Dionysus to calculate H_0, H_1, H_2 for a random subsample of the dataset **Dragon2000**. The number of points in the subsample is increased by 50% each row. The last row represents only a single run, due to prohibitively high run time. For the set of 20 times, the mean, min, max, and standard deviation are all given. All times are in seconds.

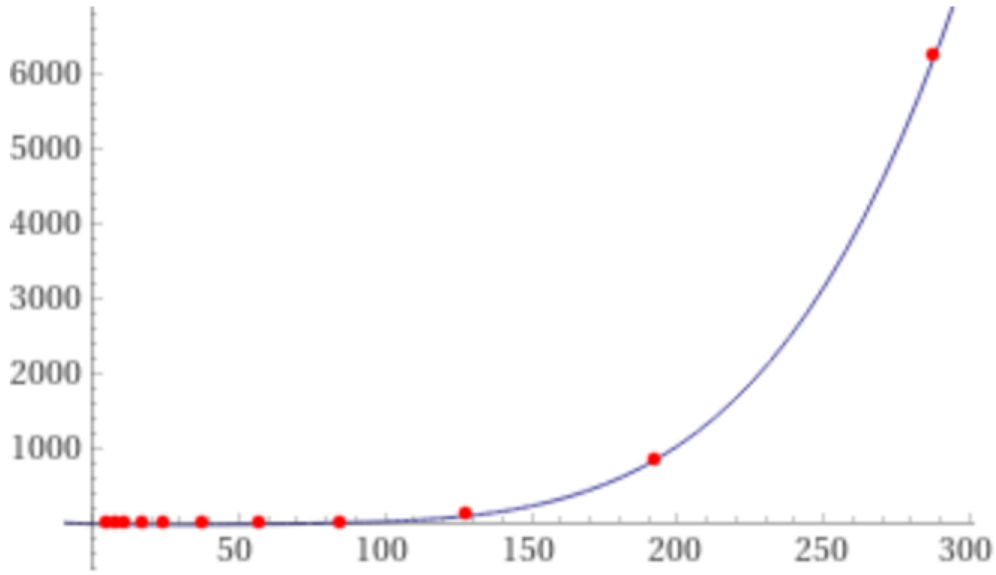


Figure 5.2: A graph of the points of Table 5.2, with the best-fit quartic line. Both the polynomial and the image were created with Wolfram Alpha. The r^2 value is .999997.

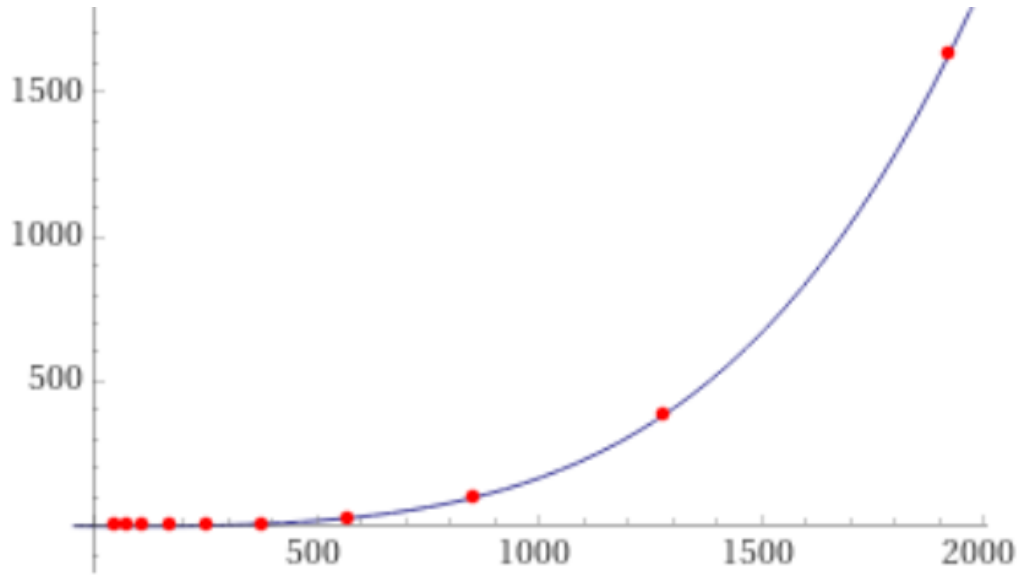


Figure 5.3: A graph of the points of Table 5.3, with the best-fit quartic line. Both the polynomial and the image were created with Wolfram Alpha. The r^2 value is greater than .999999.

performed by a constant factor but this does not necessarily improve scaling.

We conjecture that the implementation of persistent homology calculation for H_d and below of a point cloud of n points in Dionysus is $O(n^{d+2})$, and the same for Ripser.

5.2.2 Scaling with number of homology groups computed

Most applications of persistent homology in the literature that the author is familiar with involve at most computations of H_0 and H_1. This is understandable—the "meaning" of non-trivial elements of the higher order homology groups of a point cloud quickly becomes unclear. It is fortunate for us that these are rarely useful in applications, as the expectation is that these higher groups would grow to be unweildy to compute very swiftly, an expectation we will see confirmed here.

# of points	mean	min	max	std dev
50	0.026	0.025	0.03	0.001
75	0.084	0.080	0.097	0.003
112	0.181	0.173	0.199	0.005
169	0.602	0.569	0.645	0.018
253	2.191	2.02	2.272	0.068
380	7.937	7.272	8.397	0.339
570	27.947	24.783	29.468	1.226
854	98.891	91.065	110.853	5.226
1281	382.773	348.107	411.482	17.295
1922	1624.179	1437.346	1734.625	81.949

Table 5.3: Ripser computation of subsamples of Dragon2000 dataset. Each row of this table is a set of 20 runs of Ripser to calculate H_0, H_1, H_2 for a random subsample of the dataset **Dragon2000**. The number of points in the subsample is increased by 50% each row. For the set of 20 times, the mean, min, max, and standard deviation are all given. All times are in seconds.

Moreover, although testing this is beyond the scope of this dissertation, we expect each additional higher homology group to make the scaling times with number of points worse.

In Table 5.4, we show the time for Dionysus to compute the first several homology groups of **Random**.[4] Computational limitations prevent us from calculating a high sample from H_6, or indeed calculating any groups beyond that. With this small a number of points, accurate curve fitting is not feasible, but it seems that log of the number of seconds is increasing at a nearly constant rate. Table 5.5 gives the same information for Ripser, with very similar results. As noted before, for the naive algorithm to calculate H_k of n points requires calculations for $\binom{n}{k}$ simplices, which (for $k << n$) grows roughly as $O(n^k)$. We thus observe that both theory and observation is consistent with exponential growth, and feel some relief that these

[4] **Random** was picked due to low enough dimensionality to be tractable, but high enough to have nontrivial higher homology.

Highest H_i computed	mean	min	max	std dev
0	0.019	0.018	0.03	0.003
1	0.29	0.287	0.308	0.004
2	3.672	3.649	3.699	0.012
3	36.161	35.999	36.553	0.149
4	285.756	284.89	287.456	0.516
5	1998.12	1978.995	2005.671	5.689
6	15041.805	-	-	-

Table 5.4: Dionysus computation of **Random** dataset. Each row of this table except the last is a set of 20 runs of Dionysus to calculate H_0 through H_i for the dataset **Random**. The maximum homology dimension calculated increases each row. The last row represents only a single run, due to prohibitively high run time. For the set of 20 times, the mean, min, max, and standard deviation are all given. All times are in seconds.

Highest H_i computed	mean	min	max	std dev
0	0.001	0.001	0.003	0
1	0.003	0.003	0.005	0
2	0.028	0.027	0.032	0.001
3	0.171	0.152	0.189	0.015
4	1.522	1.497	1.609	0.027
5	12.731	12.598	13.077	0.111
6	85.465	84.58	86.007	0.366

Table 5.5: Ripser computation of **Random** dataset. Each row of this table is a set of 20 runs of Ripser to calculate H_0 through H_i for the dataset **Random**. The maximum homology dimension calculated increases each row. For the set of 20 times, the mean, min, max, and standard deviation are all given. All times are in seconds.

intrinsic dimension	mean	min	max	std dev
3	3.773	3.76	3.809	0.012
53	3.786	3.766	3.872	0.025
103	3.774	3.765	3.804	0.012
153	3.764	3.757	3.796	0.011
203	3.762	3.74	3.818	0.016
253	3.756	3.749	3.789	0.011
303	3.76	3.749	3.794	0.015
353	3.81	3.748	4.552	0.171
403	3.754	3.732	3.795	0.012
453	3.769	3.742	4.02	0.06

Table 5.6: Ripser computation of the **Vicsek** dataset with dimensional noise. Ripser computation of the **Vicsek** dataset with dimensional noise. Each row of this table is a set of 20 runs of Ripser to calculate H_0, H_1, H_2 for the dataset **Vicsek**. First, the points are randomly rotated into $\mathbb{R}^{\text{Intrinsic dimension}}$. Then, a small amount of noise is added to them (The noise is normally distributed with mean 0 and standard deviation 5% of the diameter of the point set). Then the points are randomly rotated into 1000 dimensions, and their persistent homology is calculated. For the set of 20 times, the mean, min, max, and standard deviation are all given. All times are in seconds.

higher homology groups are not desired for most applications.

5.2.3 Scaling with intrinsic dimension of the dataset

We don't expect the calculation of persistent homology to depend much, if at all, on the intrinsic dimension of the dataset. In Table 5.6, we see that indeed, it does not—adding higher dimensional noise to the dataset has almost no effect on the times to calculate persistent homology. Even going from 3 dimensions to 453 dimensions has almost no effect on the average time to calculate the persistent homology through H_2.

We do not perform this calculation with Dionysus because, as we will see in the next Section, Dionysus scales quite badly with ambient dimension.

5.2.4 Scaling with ambient dimension

The extent to which the speed of persistent homology calculations depends upon various parameters will, of course, change depending not only on what algorithms are used to calculate it, but on the exact methods the software uses to implement those calculations. Nowhere is that more apparent than in the effect of the ambient dimension upon the speed of calculation. Ripser takes the input point cloud, and immediately converts it to a matrix of pairwise distances; from then on, a "point" is merely a reference to a certain column of a matrix. Dionysus, on the other hand, continues to view a point in $\mathbb{R}^d$ as a list of d coordinates, so any manipulations of it require manipulating all the coordinates. Thus, a reduction in the ambient dimension affects the computation time of Dionysus in almost direct proportion. Ripser, in contrast, is affected through the first step, which is computationally already very fast, and further effects are primarily through changes which affect the actual simplices and matrix reduction itself.

Table 5.7 shows the computation time for Dionysus to calculate the through H_1 of the **Random** dataset. The scaling is roughly linear in the number of ambient dimensions. Ripser, on the other hand, is almost totally insensitive to the number of ambient dimensions, as we will see in the next section.

5.3 Calculation of persistent homology via random projections

Corollary 4.3.3 in Chapter 4 tells us that persistent homology can be approximately preserved through random projections. In this Section, we test the increase

Ambient dimension	mean	min	max	std dev
1024	11.142	11.111	11.194	0.02
512	5.62	5.616	5.676	0.013
256	2.865	2.864	2.868	0.001
128	1.491	1.49	1.505	0.003
64	0.759	0.756	0.765	0.002
32	0.42	0.415	0.422	0.002
16	0.242	0.24	0.245	0.002

Table 5.7: Dionysus computation of the **Random** dataset isometrically embedded in larger dimensions. Each row of this table is a set of 20 runs of Dionysus to calculate H_0, H_1 for the dataset **Random**, after the points are randomly rotated into $\mathbb{R}^{\text{ambient dimension}}$. Notice that halving the ambient dimension results in roughly halving the computation time. For the set of 20 times, the mean, min, max, and standard deviation are all given. All times are in seconds.

in speed this gives us with the fastest currently available software. Tables 5.8, 5.9, 5.10, 5.11, 5.12, 5.13 show the time it takes to calculate each dataset when it has been random rotated into 1000 dimensions, and compares it to the time to calculate after being randomly projected back down.

As the comparative importance of the time to calculate the matrix of pairwise distances changes dramatically depending on how many homology groups we are calculating, we include the results with just H_0, H_1 with H_0, and all three of the lowest groups through H_2.

In the H_0 alone case, the conversion from point cloud to distance matrix is responsible for a higher portion of the run time. Thus, the advantages of random projection are more pronounced. The run times for every one of our benchmark datasets is decreased, on median by roughly 8%. One third of the datasets showed speed increases ranging from nearly 25% to over 50%!

In the H_1 case, our picture is still very promising. All but one dataset showed

Dataset	mean	min	max	std dev
Celegans (pre)	0.025	0.024	0.026	<.001
Celegans (post)	0.023	0.022	0.024	<.001
Dragon1000 (pre)	0.222	0.208	0.227	0.003
Dragon1000 (post)	0.217	0.189	0.221	0.007
Dragon2000 (pre)	0.895	0.889	0.919	0.007
Dragon2000 (post)	0.875	0.857	0.894	0.014
Fractal (pre)	0.075	0.075	0.075	<.001
Fractal (post)	0.047	0.047	0.048	<.001
HIV1 (pre)	0.257	0.255	0.257	0.001
HIV1 (post)	0.226	0.226	0.227	<.001
House (pre)	0.057	0.056	0.057	<.001
House (post)	0.043	0.042	0.043	<.001
Human (pre)	0.417	0.413	0.419	0.002
Human (post)	0.405	0.383	0.414	0.012
Klein (pre)	0.045	0.045	0.045	<.001
Klein (post)	0.042	0.042	0.043	<.001
Network (pre)	0.041	0.041	0.042	<.001
Network (post)	0.038	0.036	0.039	0.001
Random (pre)	0.002	0.002	0.002	<.001
Random (post)	0.001	0.001	0.001	<.001
Senate (pre)	0.005	0.005	0.005	<.001
Senate (post)	0.003	0.003	0.003	<.001
Vicsek (pre)	0.026	0.025	0.027	<.001
Vicsek (post)	0.024	0.023	0.026	0.001

Table 5.8: Ripser computation of many datasets, first randomly rotated into 1000 dimensions, then projected down to 10. Each row of this table is a set of 20 runs of Ripser to calculate H_0. For each set of 20 times, the mean, min, max, and standard deviation are all given. All times are in seconds.

Dataset	mean	min	max	std dev
Celegans difference	0.002	0.002	0.003	<.001
% difference	8.478	7.9	11.482	0.707
Dragon1000 difference	0.005	-0.011	0.034	0.008
% difference	2.369	-5.266	15.193	3.541
Dragon2000 difference	0.02	-0.001	0.037	0.012
% difference	2.284	-0.151	4.041	1.301
Fractal difference	0.028	0.027	0.028	<.001
% difference	36.943	36.558	37.397	0.226
HIV1 difference	0.03	0.029	0.031	0.001
% difference	11.833	11.351	12.238	0.248
House difference	0.014	0.013	0.015	<.001
% difference	24.383	23.78	25.537	0.407
Human difference	0.012	0.003	0.032	0.011
% difference	2.893	0.786	7.775	2.639
Klein difference	0.003	0.003	0.003	<.001
% difference	6.673	5.597	7.556	0.531
Network difference	0.003	0.003	0.005	0.001
% difference	8.46	6.996	12.403	1.552
Random difference	0.001	0.001	0.001	<.001
% difference	53.377	51.346	55.234	1.425
Senate difference	0.002	0.002	0.002	<.001
% difference	41.845	40.434	43.082	0.83
Vicsek difference	0.002	0.001	0.003	0.001
% difference	6.357	3.261	10.508	2.59

Table 5.9: Ripser computation of many datasets, first randomly rotated into 1000 dimensions, then projected down to 10. Each row of this table is the differences between a set of 20 runs of Ripser to calculate H_0 pre- and post-projection. For each set of 20 differences, the mean, min, max, and standard deviation are all given. All times are in seconds.

Dataset	mean	min	max	std dev
Celegans (pre)	0.079	0.078	0.082	0.001
Celegans (post)	0.078	0.075	0.082	0.001
Dragon1000 (pre)	0.675	0.673	0.677	0.001
Dragon1000 (post)	0.668	0.61	0.719	0.027
Dragon2000 (pre)	3.541	3.422	3.696	0.068
Dragon2000 (post)	3.367	3.028	3.884	0.208
Fractal (pre)	0.185	0.183	0.188	0.001
Fractal (post)	0.165	0.145	0.187	0.014
HIV1 (pre)	1.353	1.344	1.379	0.008
HIV1 (post)	0.999	0.898	1.087	0.049
House (pre)	0.138	0.136	0.14	0.001
House (post)	0.12	0.106	0.143	0.013
Human (pre)	1.038	1.031	1.043	0.003
Human (post)	1.43	1.374	1.497	0.03
Klein(pre)	0.284	0.281	0.286	0.001
Klein (post)	0.213	0.142	0.329	0.052
Network (pre)	0.154	0.153	0.155	<.001
Network (post)	0.077	0.075	0.08	0.001
Random (pre)	0.004	0.004	0.004	<.001
Random (post)	0.003	0.002	0.003	<.001
Senate (pre)	0.011	0.011	0.011	<.001
Senate (post)	0.009	0.009	0.009	<.001
Vicsek (pre)	0.087	0.087	0.088	<.001
Vicsek (post)	0.049	0.046	0.053	0.002

Table 5.10: Ripser computation of many datasets, first randomly rotated into 1000 dimensions, then projected down to 10. Each row of this table is a set of 20 runs of Ripser to calculate H_1. For each set of 20 times, the mean, min, max, and standard deviation are all given. All times are in seconds.

Dataset	mean	min	max	std dev
Celegans difference	0.0004	-0.001	0.004	0.001
% difference	0.514	-1.42	4.939	1.543
Dragon1000 difference	0.008	-0.042	0.066	0.027
% difference	1.124	-6.246	9.763	3.993
Dragon2000 difference	0.173	-0.237	0.515	0.211
% difference	4.868	-6.489	14.457	5.914
Fractal difference	0.019	-0.001	0.038	0.013
% difference	10.5	-0.456	20.863	7.124
HIV1 difference	0.354	0.263	0.455	0.048
% difference	26.179	19.498	33.619	3.582
House difference	0.018	-0.005	0.031	0.012
% difference	13.158	-3.263	22.713	8.858
Human difference	-0.392	-0.454	-0.335	0.03
% difference	-37.826	-43.496	-32.19	2.922
Kleinbottle difference	0.071	-0.045	0.142	0.052
% difference	25.076	-15.872	49.918	18.374
Network difference	0.077	0.073	0.079	0.001
% difference	50.131	47.921	51.125	0.809
Random difference	0.001	0.001	0.002	<.001
% difference	36.444	31.911	43.05	3.01
Senate difference	0.002	0.001	0.002	<.001
% difference	16.909	11.244	20.696	2.541
Vicsek difference	0.038	0.034	0.041	0.002
% difference	43.78	39.217	46.789	1.966

Table 5.11: Ripser computation of many datasets, first randomly rotated into 1000 dimensions, then projected down to 10. Each row of this table is the differences between a set of 20 runs of Ripser to calculate H_1 pre- and post-projection. For each set of 20 differences, the mean, min, max, and standard deviation are all given. All times are in seconds.

Dataset	mean	min	max	std dev
Celegans (pre)	3.534	3.495	3.986	0.104
Celegans (post)	3.527	3.314	3.718	0.106
Dragon1000 (pre)	166.588	165.612	169.937	0.831
Dragon1000 (post)	162.747	154.296	171.653	4.366
Dragon2000 (pre)	1716.02	1696.383	1739.451	10.616
Dragon2000 (post)	1751.798	1692.084	1833.825	38.766
Fractal (pre)	19.115	19.015	19.291	0.075
Fractal (post)	19.997	17.429	21.938	1.09
HIV1 (pre)	273.694	272.051	275.807	0.998
HIV1 (post)	247.278	226.706	266.279	10.406
House (pre)	13.73	13.645	13.844	0.045
House (post)	12.942	11.957	13.942	0.544
Human (pre)	504.035	501.405	505.167	0.809
Human (post)	554.189	504.662	581.235	17.456
Klein (pre)	14.852	14.737	14.986	0.05
Klein (post)	14.7	13.119	16.288	0.885
Network (pre)	12.281	12.193	12.349	0.046
Network (post)	8.132	7.48	8.406	0.231
Random (pre)	0.028	0.028	0.031	0.001
Random (post)	0.026	0.024	0.027	0.001
Senate (pre)	0.146	0.144	0.148	0.001
Senate (post)	0.111	0.106	0.129	0.005
Vicsek(pre)	3.729	3.709	3.793	0.02
Vicsek (post)	3.687	3.474	3.882	0.109

Table 5.12: Ripser computation of many datasets, first randomly rotated into 1000 dimensions, then projected down to 10. Each row of this table is a set of 20 runs of Ripser to calculate H_2. For each set of 20 times, the mean, min, max, and standard deviation are all given. All times are in seconds.

Dataset	mean	min	max	std dev
Celegans difference	0.007	-0.142	0.268	0.112
% difference	0.153	-4.041	6.722	3.088
Dragon1000 difference	3.84	-1.716	11.316	3.998
% difference	2.31	-1.01	6.833	2.403
Dragon2000 difference	-35.778	-117.059	21.915	38.649
% difference	-2.087	-6.86	1.279	2.253
Fractal difference	-0.882	-2.766	1.595	1.078
% difference	-4.613	-14.426	8.383	5.637
HIV1 difference	26.416	7.674	46.131	10.485
% difference	9.65	2.801	16.908	3.829
House difference	0.788	-0.205	1.728	0.55
% difference	5.737	-1.495	12.625	4.001
Human difference	-50.154	-77.709	-0.14	17.605
% difference	-9.952	-15.433	-0.028	3.494
Klein difference	0.153	-1.448	1.742	0.884
% difference	1.029	-9.756	11.724	5.96
Network difference	4.15	3.87	4.814	0.233
% difference	33.789	31.526	39.157	1.886
Random difference	0.003	0.0004	0.005	0.001
% difference	9.477	1.512	15.898	3.76
Senate difference	0.035	0.015	0.039	0.005
% difference	23.756	10.558	26.991	3.453
Vicsek difference	0.041	-0.164	0.296	0.12
% difference	1.1	-4.419	7.806	3.204

Table 5.13: Ripser computation of many datasets, first randomly rotated into 1000 dimensions, then projected down to 10. Each row of this table is the differences between a set of 20 runs of Ripser to calculate H_2 pre- and post-projection. For each set of 20 differences, the mean, min, max, and standard deviation are all given. All times are in seconds.

an increase, and the median speedup surprisingly *increased*, nearly doubling to roughly 15%! Nearly half the datasets had speed increases of over 25%, and again the greatest increase was over 50%.

Random projection results in less of a speed up in the H_2 case. While the median case is still faster post-projection, it is only by about 1.7%, and, while two of the datasets showed were sped up by significantly more than 10%, it is only those two. Additionally, for three datasets, the computations were actually slowed by the lower ambient dimension.

As we have previously discussed, applied computational topology rarely makes use of the homology groups above H_1 in practice, so random projection may be useful in cases where a sacrifice of accuracy in favor of speed is desirable.

5.3.1 Reasons to expect computational slowdown or speedup

Ideally, we would be able to characterize what datasets are good candidates for significant speedup or slowdown through random projection. In particular, the datasets **Human** and **HIV1** are similar on many metrics–similar number of points, similar ambient dimension, both modeling genomic data, albeit with different models.

We can't even claim that the computational speedup for H_0 is correlated with that for H_2—the datasets **Fractal** and **Random** each have very significant speedups for H_0, but **Fractal** experiences the second-worst slow-down for H_2, while **Random** is one of the four most positively effected.

5.4 Heuristics for better preservation than guaranteed

We close with some discussion of the accuracy of this method of estimating persistent homology. Ramamurthy, Varshney, and Thiagarajan [113] have given some empirical evidence that persistent betti numbers can be approximately reconstructed with high probability after random projection. And even in cases that the Johnson Lindenstrauss Lemma (Theorem 4.3.1) does not give any guarantees, it is not uncommon for significant features of homology to be well-preserved, as in Figure 5.4.

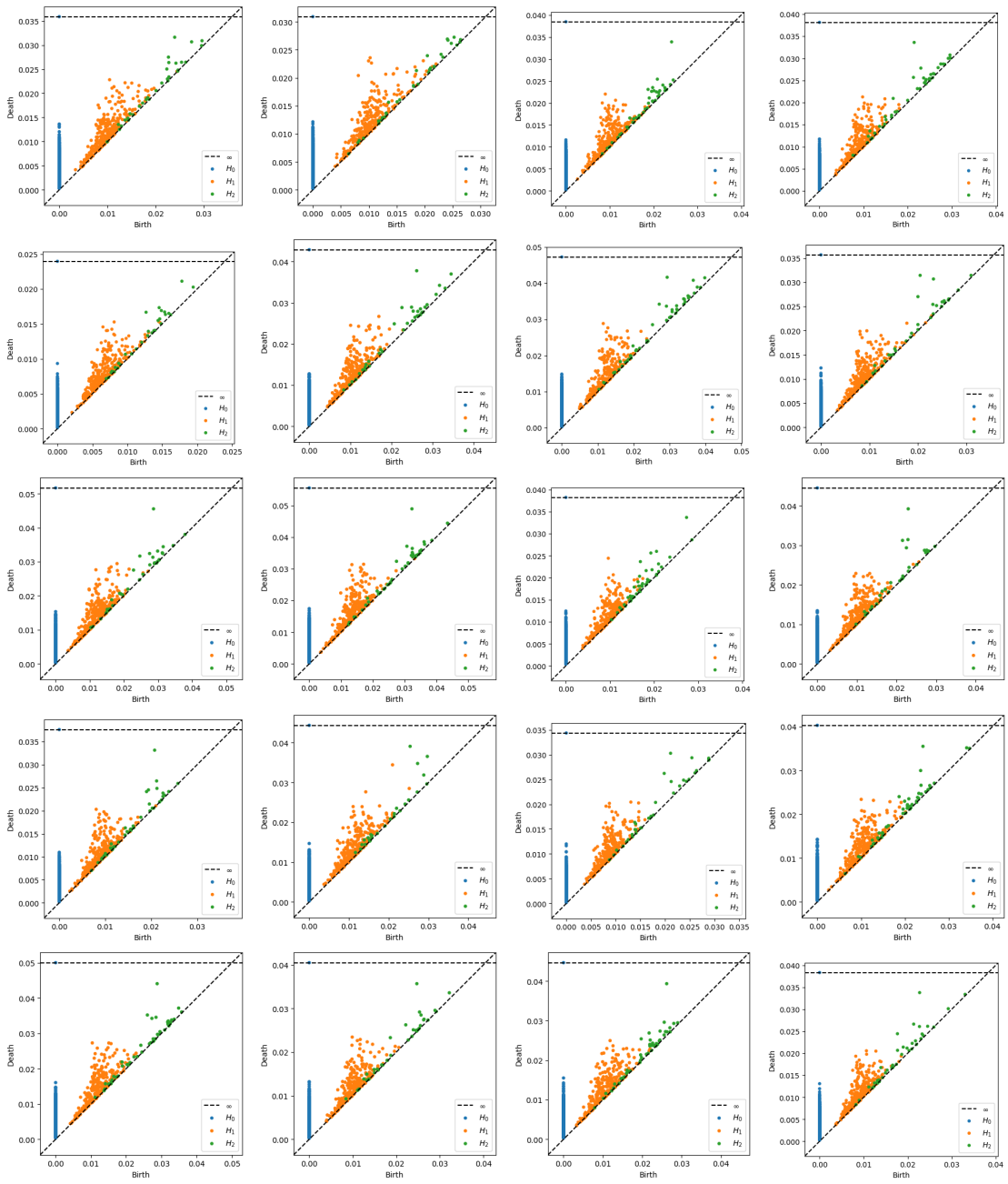


Figure 5.4: Persistence diagrams for the Dragon1000 dataset, randomly rotated in 1000 dimensions, and projected down to 10. Notice how, in most of the diagrams, a green dot representing an element of H_2 has high persistence. On the other hand, notice the difference in scales–between images 5 and 10, the vertical scale nearly doubles! To avoid unconscious bias effects, Dragon1000 was the first dataset the author performed this with, and these are the results of the first 20 attempts.

Bibliography

[1] H. Adams, T. Emerson, M. Kirby, R. Neville, C. Peterson, P. Shipman, S. Chepushtanova, E. Hanson, F. Motta, and L. Ziegelmeier. Persistence images: A stable vector representation of persistent homology. *Journal of Machine Learning Research*, 18, 2017.

[2] R. J. Adler, O. Bobrowski, M. S. Borman, E. Subag, S. Weinberger, et al. Persistent homology for random fields and complexes. In *Borrowing strength: theory powering applications–a Festschrift for Lawrence D. Brown*, pages 124–143. Institute of Mathematical Statistics, 2010.

[3] S. Agostinelli, J. Allison, K. a. Amako, J. Apostolakis, H. Araujo, P. Arce, M. Asai, D. Axen, S. Banerjee, G. . Barrand, et al. GEANT4—a simulation toolkit. *Nuclear instruments and methods in physics research section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506(3):250–303, 2003.

[4] H. Alt. *Computational discrete mathematics: advanced lectures*, volume 2122. Springer, 2003.

[5] A. Antoszewski, C.-J. Feng, B. P. Vani, E. H. Thiede, L. Hong, J. Weare, A. Tokmakoff, and A. R. Dinner. Insulin dissociates by diverse mechanisms of coupled unfolding and unbinding. *The Journal of Physical Chemistry B*, 124(27):5571–5587, 2020.

[6] M. A. Armstrong. *Basic topology*. Springer Science & Business Media, 2013.

[7] S. Arya, J.-D. Boissonnat, K. Dutta, and M. Lotz. Dimensionality reduction for k-distance applied to persistent homology. *Journal of Applied and Computational Topology*, 5(4):671–691, 2021.

[8] D. Attali and A. Lieutier. Geometry-driven collapses for converting a Čech complex into a triangulation of a nicely triangulable shape. *Discrete & Computational Geometry*, 54(4):798–825, 2015.

[9] D. Attali, A. Lieutier, and D. Salinas. Vietoris–Rips complexes also provide topologically correct reconstructions of sampled shapes. *Computational Geometry*, 46(4):448–465, 2013.

[10] W. Bae, J. Yoo, and J. Chul Ye. Beyond deep residual learning for image restoration: Persistent homology-guided manifold simplification. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 145–153, 2017.

[11] R. Balan, M. Begué, C. Clark, and K. Okoudjou. Optimization methods for frame conditioning and application to graph Laplacian scaling. In *Frames and Other Bases in Abstract and Function Spaces*, pages 27–45. Springer, 2017.

[12] R. G. Baraniuk and M. B. Wakin. Random projections of smooth manifolds. *Foundations of computational mathematics*, 9(1):51–77, 2009.

[13] S. Basu and L. Parida. Spectral sequences, exact couples and persistent homology of filtrations. *Expositiones Mathematicae*, 35(1):119–132, 2017.

[14] U. Bauer. Ripser: Efficient computation of Vietoris–Rips persistence barcodes.

[15] U. Bauer. Ripser: efficient computation of Vietoris-Rips persistence barcodes. *Journal of Applied and Computational Topology*, 2021.

[16] U. Bauer and H. Edelsbrunner. The Morse theory of Čech and Delaunay complexes. *Transactions of the American Mathematical Society*, 369(5):3741–3762, 2017.

[17] P. Bendich, H. Edelsbrunner, D. Morozov, and A. Patel. The robustness of level sets. In *European Symposium on Algorithms*, pages 1–10. Springer, 2010.

[18] E. Betti. Sopra gli spazi di un numero qualunque di dimensioni. *Annali di Matematica Pura ed Applicata (1867-1897)*, 4(1):140–158, 1870.

[19] S. Bhattacharya. Discrete optimal search library (dosl): A template-based C++ library for discrete optimal search, 2017. Available at https://github.com/subh83/DOSL.

[20] S. Bhattacharya, R. Ghrist, and V. Kumar. Persistent homology for path planning in uncertain environments. *IEEE Transactions on Robotics*, 31(3):578–590, 2015.

[21] K. Borsuk. On the imbedding of systems of compacta in simplicial complexes. *Fundamenta Mathematicae*, 35(1):217–234, 1948.

[22] A. K. Bousfield and D. M. Kan. *Homotopy limits, completions and localizations*, volume 304. Springer Science & Business Media, 1972.

[23] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[24] G. E. Bredon. *Topology and geometry*, volume 139. Springer Science & Business Media, 2013.

[25] P. Bubenik, V. De Silva, and J. Scott. Metrics for generalized persistence modules. *Foundations of Computational Mathematics*, 15(6):1501–1531, 2015.

[26] P. Bubenik et al. Statistical topological data analysis using persistence landscapes. *J. Mach. Learn. Res.*, 16(1):77–102, 2015.

[27] P. Bubenik and P. T. Kim. A statistical approach to persistent homology. *Homology, homotopy and Applications*, 9(2):337–362, 2007.

[28] P. Bubenik and J. A. Scott. Categorification of persistent homology. *Discrete & Computational Geometry*, 51(3):600–627, 2014.

[29] M. Buchet, F. Chazal, S. Y. Oudot, and D. R. Sheehy. Efficient and robust persistent homology for measures. *Computational Geometry*, 58:70–96, 2016.

[30] F. Cagliari, M. Ferri, and P. Pozzi. Size functions from a categorical viewpoint. *Acta Applicandae Mathematica*, 67(3):225–235, 2001.

[31] E. Candes, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 52(2):489–509, 2006.

[32] E. Candes and T. Tao. Near optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans. Inform. Theory*, 52(12):5406–5425, 2006.

[33] E. J. Candes, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.

[34] G. Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.

[35] G. Carlsson and V. De Silva. Zigzag persistence. *Foundations of computational mathematics*, 10(4):367–405, 2010.

[36] G. Carlsson, V. De Silva, and D. Morozov. Zigzag persistent homology and real-valued functions. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, pages 247–256. ACM, 2009.

[37] G. Carlsson, T. Ishkhanov, V. De Silva, and A. Zomorodian. On the local behavior of spaces of natural images. *International journal of computer vision*, 76(1):1–12, 2008.

[38] G. Carlsson and A. Zomorodian. The theory of multidimensional persistence. *Discrete & Computational Geometry*, 42(1):71–93, 2009.

[39] E. W. Chambers, B. T. Fasy, and L. Ziegelmeier. *Research in Computational Topology*. Springer, 2018.

[40] F. Chazal, D. Cohen-Steiner, M. Glisse, L. J. Guibas, and S. Y. Oudot. Proximity of persistence modules and their diagrams. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, pages 237–246, 2009.

[41] F. Chazal, B. Fasy, F. Lecci, B. Michel, A. Rinaldo, and L. Wasserman. Subsampling methods for persistent homology. In *International Conference on Machine Learning*, pages 2143–2151. PMLR, 2015.

[42] C. Chen and M. Kerber. Persistent homology computation with a twist. In *Proceedings 27th European workshop on computational geometry*, volume 11, pages 197–200, 2011.

[43] G. Christie, A. Shoemaker, K. Kochersberger, P. Tokekar, L. McLean, and A. Leonessa. Radiation search operations using scene understanding with autonomous uav and ugv. *Journal of Field Robotics*, 34(8):1450–1468, 2017.

[44] A. Cloninger. *Exploiting data-dependent structure for improving sensor acquisition and integration*. PhD thesis, University of Maryland, College Park, 2014.

[45] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. In *Proceedings of the twenty-first annual symposium on Computational geometry*, pages 263–271. ACM, 2005.

[46] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Extending persistence using Poincaré and Lefschetz duality. *Foundations of Computational Mathematics*, 9(1):79–103, 2009.

[47] D. Cohen-Steiner, H. Edelsbrunner, J. Harer, and D. Morozov. Persistent homology for kernels, images, and cokernels. In *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*, pages 1011–1020. SIAM, 2009.

[48] W. Crawley-Boevey. Decomposition of pointwise finite-dimensional persistence modules. *Journal of Algebra and its Applications*, 14(05):1550066, 2015.

[49] F. H. Croom. *Basic concepts of algebraic topology*. Springer Science & Business Media, 2012.

[50] J. Curry, R. Ghrist, and V. Nanda. Discrete morse theory for computing cellular sheaf cohomology. *Foundations of Computational Mathematics*, 16(4):875–897, 2016.

[51] W. Czaja and W. Li. Analysis of time-frequency scattering transforms. *Applied and Computational Harmonic Analysis*, 47(1):149–171, 2019.

[52] W. Czaja and W. Li. Rotationally invariant time–frequency scattering transforms. *Journal of Fourier Analysis and Applications*, 26(1):1–23, 2020.

[53] W. Czaja, B. Manning, L. McLean, and J. M. Murphy. Fusion of aerial gamma-ray survey and remote sensing data for a deeper understanding of radionuclide fate after radiological incidents: examples from the Fukushima Dai-Ichi response. *Journal of Radioanalytical and Nuclear Chemistry*, 307(3):2397–2401, 2016.

[54] S. Dasgupta and A. Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

[55] V. De Silva and G. E. Carlsson. Topological estimation using witness complexes. *SPBG*, 4:157–166, 2004.

[56] V. De Silva and R. Ghrist. Coverage in sensor networks via persistent homology. *Algebraic & Geometric Topology*, 7(1):339–358, 2007.

[57] V. De Silva, R. Ghrist, and A. Muhammad. Blind swarms for coverage in 2-d. In *Robotics: Science and Systems*, pages 335–342, 2005.

[58] V. De Silva, D. Morozov, and M. Vejdemo-Johansson. Dualities in persistent (co) homology. *Inverse Problems*, 27(12):124003, 2011.

[59] V. De Silva, D. Morozov, and M. Vejdemo-Johansson. Persistent cohomology and circular coordinates. *Discrete & Computational Geometry*, 45(4):737–759, 2011.

[60] T. K. Dey, F. Mémoli, and Y. Wang. Multiscale mapper: Topological summarization via codomain covers. In *Proceedings of the twenty-seventh annual acm-siam symposium on discrete algorithms*, pages 997–1013. SIAM, 2016.

[61] A. Dick and H. Weyl. *Emmy Noether, 1882-1935*. Springer, 1981.

[62] T. Doster. *Harmonic analysis inspired data fusion for applications in remote sensing*. PhD thesis, University of Maryland, College Park, 2014.

[63] D. S. Dummit and R. M. Foote. *Abstract algebra*. Wiley Hoboken, 3 edition, 2004.

[64] H. Edelsbrunner. Alpha shapes-a survey. *Tessellations in the Sciences*, 27:1–25, 2010.

[65] H. Edelsbrunner and J. Harer. Persistent homology-a survey. *Contemporary mathematics*, 453:257–282, 2008.

[66] H. Edelsbrunner and J. Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.

[67] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Transactions on information theory*, 29(4):551–559, 1983.

[68] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 454–463. IEEE, 2000.

[69] H. Edelsbrunner and D. Morozov. Persistent homology: theory and practice. Technical report, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), 2012.

[70] H. Edelsbrunner and E. P. Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics (TOG)*, 13(1):43–72, 1994.

[71] S. Eilenberg and J. A. Zilber. Semi-simplicial complexes and singular homology. *Annals of Mathematics*, pages 499–513, 1950.

[72] J. F. Espinoza, R. Hernández-Amador, H. A. Hernández-Hernández, and B. Ramonetti-Valencia. A numerical approach for the filtered generalized Čech complex. *Algorithms*, 13(1):11, 2020.

[73] L. Euler. Elementa doctrinae solidorum. *Novi commentarii academiae scientiarum Petropolitanae*, pages 109–140, 1758.

[74] B. T. Fasy, F. Lecci, A. Rinaldo, L. Wasserman, S. Balakrishnan, and A. Singh. Confidence sets for persistence diagrams. *The Annals of Statistics*, 42(6):2301–2339, 2014.

[75] A. T. Fomenko. *Visual geometry and topology*. Springer Science & Business Media, 2012.

[76] P. Frosini. A distance for similarity classes of submanifolds of a Euclidean space. *Bulletin of the Australian Mathematical Society*, 42(3):407–415, 1990.

[77] K. Fukumizu, F. R. Bach, and M. I. Jordan. Kernel dimension reduction in regression. *The Annals of Statistics*, 37(4):1871–1905, 2009.

[78] R. Ghrist. Barcodes: the persistent topology of data. *Bulletin of the American Mathematical Society*, 45(1):61–75, 2008.

[79] R. W. Ghrist. *Elementary applied topology*, volume 1. Createspace Seattle, WA, 2014.

[80] D. Govc. On the definition of the homological critical value. *Journal of Homotopy and Related Structures*, 11(1):143–151, 2016.

[81] A. Hafftka. *Tensor completion for multidimensional inverse problems with applications to magnetic resonance relaxometry*. PhD thesis, University of Maryland, College Park, 2016.

[82] A. Halevy. *Extensions of Laplacian eigenmaps for manifold learning*. PhD thesis, University of Maryland, College Park, 2011.

[83] A. Hatcher. *Algebraic topology*. Cambridge university press, 2005.

[84] J.-C. Hausmann et al. On the Vietoris-Rips complexes and a cohomology theory for metric spaces. *Annals of Mathematics Studies*, 138:175–188, 1995.

[85] G. Henselman and R. Ghrist. Matroid filtrations and computational persistent homology. *arXiv:1606.00199*, 2016.

[86] K. P. Heuston. *The subfornical organ and water deprivation in pigeons*. Harvard University, 1994.

[87] W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics*, 26:189–206, 1984.

[88] T. Kaczynski, K. M. Mischaikow, and M. Mrozek. *Computational homology*, volume 157. Springer, 2004.

[89] M. Kerber and R. Sharathkumar. Approximate Čech complex in low and high dimensions. In *International Symposium on Algorithms and Computation*, pages 666–676. Springer, 2013.

[90] K. Kochersberger, J. Peterson, P. Kumar, J. Bird, M. McLean, W. Czaja, W. Li, and N. Monson. Unmanned aircraft applications in radiological surveys. In *2018 IEEE International Symposium on Technologies for Homeland Security (HST)*, pages 1–5. IEEE, 2018.

[91] M. P. Lesnick. *Multidimensional interleavings and applications to topological inference*. Stanford University, 2012.

[92] W. Li. *Topics in Harmonic Analysis, Sparse Representations, and Data Analysis*. PhD thesis, University of Maryland, College Park, 2018.

[93] N. O. Malott, A. M. Sens, and P. A. Wilsey. Topology preserving data reduction for computing persistent homology. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 2681–2690. IEEE, 2020.

[94] S. Mandal. *Applications of Persistent Homology and Cycles*. PhD thesis, The Ohio State University, 2020.

[95] W. S. Massey. *A basic course in algebraic topology*, volume 127. Springer, 2019.

[96] J. P. May. *A concise course in algebraic topology*. University of Chicago press, 1999.

[97] J. P. May. *The geometry of iterated loop spaces*, volume 271. Springer, 2006.

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

[99] D. Morozov. Dionysus. *Software available at http://www. mrzv. org/software/dionysus*, 2012.

[100] D. Morozov and A. Nigmetov. Towards lockfree persistent homology. In *Proceedings of the 32nd ACM Symposium on Parallelism in Algorithms and Architectures*, pages 555–557, 2020.

[101] J. M. Murphy. *Anisotropic harmonic analysis and integration of remotely sensed data*. PhD thesis, University of Maryland, College Park, 2015.

[102] M. Nicolau, A. J. Levine, and G. Carlsson. Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival. *Proceedings of the National Academy of Sciences*, 108(17):7265–7270, 2011.

[103] P. Niyogi, S. Smale, and S. Weinberger. Finding the homology of submanifolds with high confidence from random samples. *Discrete & Computational Geometry*, 39(1-3):419–441, 2008.

[104] P. Niyogi, S. Smale, and S. Weinberger. A topological view of unsupervised learning from noisy data. *SIAM Journal on Computing*, 40(3):646–663, 2011.

[105] F. O. N. Njeunje. *Computational methods in machine learning: transport model, Haar wavelet, DNA classification, and MRI*. PhD thesis, University of Maryland, College Park, 2018.

[106] N. Otter. Ph-roadmap, 2017.

[107] N. Otter, M. A. Porter, U. Tillmann, P. Grindrod, and H. A. Harrington. A roadmap for the computation of persistent homology. *EPJ Data Science*, 6(1):17, 2017.

[108] S. Y. Oudot. On the topology of the restricted Delaunay triangulation and witness complex in higher dimensions. *arXiv:0803.1296*, 2008.

[109] S. Y. Oudot. *Persistence theory: from quiver representations to data analysis*, volume 209. American Mathematical Society Providence, 2015.

[110] J. B. Pérez, S. Hauke, U. Lupo, M. Caorsi, and A. Dassatti. giotto-ph: A python library for high-performance computation of persistent homology of Vietoris-Rips filtrations. *arXiv:2107.05412*, 2021.

[111] H. Poincaré. *Analysis situs*. Gauthier-Villars, 1895.

[112] D. Quillen. Higher algebraic k-theory: I. In *Higher K-theories*, pages 85–147. Springer, 1973.

[113] K. N. Ramamurthy, K. R. Varshney, and J. J. Thiagarajan. Computing persistent homology under random projection. In *2014 IEEE Workshop on Statistical Signal Processing (SSP)*, pages 105–108. IEEE, 2014.

[114] B. Riemann. *Theorie der Abel'schen functionen*. Georg Reimer, 1857.

[115] V. Robins. Towards computing homology from finite approximations. In *Topology proceedings*, volume 24, pages 503–532, 1999.

[116] M. Robinson. *Topological signal processing*, volume 81. Springer, 2014.

[117] L. K. Saul, K. Q. Weinberger, F. Sha, J. Ham, and D. D. Lee. Spectral methods for dimensionality reduction. *Semi-supervised learning*, 3, 2006.

[118] G. Segal. Classifying spaces and spectral sequences. *Publications Mathématiques de l'IHÉS*, 34:105–112, 1968.

[119] D. R. Sheehy. Linear-size approximations to the Vietoris–Rips filtration. *Discrete & Computational Geometry*, 49(4):778–796, 2013.

[120] D. R. Sheehy. The persistent homology of distance functions under random projection. In *Proceedings of the thirtieth annual symposium on Computational geometry*, pages 328–334, 2014.

[121] G. Singh, F. Memoli, T. Ishkhanov, G. Sapiro, G. Carlsson, and D. L. Ringach. Topological analysis of population activity in visual cortex. *Journal of vision*, 8(8):11–11, 2008.

[122] E. H. Spanier. *Algebraic topology*. Springer Science & Business Media, 1989.

[123] C. Tralie, N. Saul, and R. Bar-On. Ripser.py: A lean persistent homology library for python. *The Journal of Open Source Software*, 3(29):925, Sep 2018.

[124] B. Walker. *Topological Structure of Spatially-Distributed Network Coded Information*. PhD thesis, University of Maryland, College Park, 2014.

[125] S. Weinberger. What is... persistent homology? *Notices of the AMS*, 58(1):36–39, 2011.

[126] J. H. Whitehead. Combinatorial homotopy. i. *Bulletin of the American Mathematical Society*, 55(3):213–245, 1949.

[127] H. Whitney. Differentiable manifolds. *Annals of Mathematics*, pages 645–680, 1936.

[128] A. Zomorodian and G. Carlsson. Computing persistent homology. *Discrete & Computational Geometry*, 33(2):249–274, 2005.

[129] A. J. Zomorodian. *Topology for computing*, volume 16. Cambridge university press, 2005.