SLATER, JOSHUA W., M.A. Sampling from the Space of Persistence Diagrams. (2024)
Directed by Dr. Thomas Weighill. 55 pp.

The efficacy of using persistent homology as a tool to understand "the shape of data" has been demonstrated in a variety of different machine learning problem domains. Like many other unsupervised techniques within machine learning, the quintessential persistent homology pipeline is one-directional; data goes in, we use persistent homology to compute information about topological invariant that are present with that data, and a succinct summarization of this information, a persistence diagram, comes out. In this work, we investigate the opposite direction of this pipeline. Using Random Walk Metropolis (RWM), we explore spaces of grayscale images and weighted graphs whose persistence diagrams approximates a given target persistence diagram, presenting sampling schemes that make this process tractable. Following an overview of relevant terminology and results, we show that our methods may be used to generate images and weighted graphs whose underlying persistence diagrams closely approximate a given target.

SAMPLING FROM THE SPACE OF PERSISTENCE DIAGRAMS

by

Joshua W. Slater

A Thesis Submitted to
the Faculty of The Graduate School at
The University of North Carolina at Greensboro
in Partial Fulfillment
of the Requirements for the Degree
Master of Arts

Greensboro
2024

Approved by

_____
Committee Chair

APPROVAL PAGE

This thesis written by Joshua W. Slater has been approved by the following committee of the Faculty of The Graduate School at The University of North Carolina at Greensboro.

Committee Chair _____
Thomas Weighill

Committee Members _____
Michael Hull

_____
Clifford Smyth

_____
Jianping Sun

_____
Date of Acceptance by Committee

_____
Date of Final Oral Examination

ACKNOWLEDGMENTS

I am deeply grateful of Dr. Weighill for his thoughtful supervision of this work, and more broadly, for his tireless support in overseeing my journey in mathematics. In addition, I would like to thank Dr. Hull, Dr. Smyth and Dr. Weighill for their support throughout my time at UNCG.

Table of Contents

List of Figures

Chapter 1: Introduction and Related Works

There are a number of reasons one might want might want to study the persistent homology pipeline in reverse, that is, to study instances or distributions of data which share a common a common persistence diagram. This idea has been tangentially explored through a variety of works and mathematical and statistical lenses over the past decade, often with practical implications in mind. Persistent homology may capture structures within data that, say, enhance the classification capabilities of a model, but instances of data (and thus of corresponding persistence diagrams) may be difficult to come by. For example, [16] details an instance in which persistent homology proved valuable in differentiating between a variety of crystalline structures, but collecting training data from which to build a model was infeasible.

At the risk of grave oversimplification, perspectives on how to explore relationships between persistence diagrams and the data that generates them fall into two camps. The first are those grounded in gradient-based optimization, sometimes referred to as topological optimization; these are intended to generate discrete instances of data whose underlying persistence diagrams approximate a given target persistence diagram. The second are those which primarily focus on sampling from a relevant space of persistence diagrams itself, with the goal being to generate some distribution of persistence diagrams from which samples that approximate a given target collection of persistence diagrams may be drawn. In some respects, the end goal of these differing perspectives is highly complimentary; one attempts to utilize some broad distribution of data as a way to generating new persistence diagrams, the other largely forgoes the data from which persistence diagrams are based, instead focusing on the space of persistence diagrams itself to generate new persistence diagrams.

The broad idea of this work is to study sufficiently restricted spaces of data, notably grayscale images and finite weighted graphs, which share a common persistence diagram using Random Walk Metropolis methods. Although we have drawn inspiration from some of the many sampling based works, namely, in that we rely on methods

from MCMC to explore the space of data which shares a common persistence diagram, the overarching goals for this work align more with optimization based methods. That is, rather than sample persistence diagrams in accordance with a model with provable statistical guarantees, we are more interested in finding instances of data (the more radically different from the data which generated the given target persistence diagram, and from one another, the better) that generate a given persistence diagram, or at least suitably approximate it.

Related Optimization-Based Work

Many optimization methods rely on some flavor of gradient descent to solve problems of the form $\min_{x \in X} \mathcal{L}(x)$, where $\mathcal{L} : X \to \mathbb{R}$ is some loss function, and thus inherently related to the thing one is trying to minimize. If one wishes to generate data that approximates a given persistence diagram, the many stability results that persistence diagrams enjoy with respect to the various p-Wasserstein distances motivates their candidacy for good choices of $\mathcal{L}$; given a target persistence diagram $\mathrm{PD}(y)$, previous work has sought to define $\mathcal{L}(x) = W_p(\mathrm{PD}(x), \mathrm{PD}(y))$, or some variation thereof. [1]

Of course, certain problems must first be overcome if we are to use the W_p distances as loss function, chief among which is that it is not immediately obvious how to calculate gradients from these W_p. In [11], it was shown that, under the assumption that points of a finite point cloud are distributed in general position, stability of bottleneck distance W_∞ implies the differentiability of the persistence map, i.e the map that sends a point cloud to its persistence diagram. As a result, they demonstrate that given a starting point cloud P and target persistence diagram $\mathcal{D}$, one can apply Newton-Raphson to iteratively transform P into a point cloud P' with $\mathrm{PD}(P') = \mathcal{D}$, and the authors provided some convergence guarantees. Broadly, their method relies on the correspondence between birth/death values and birth/death *simplices*. If you know, for instance, that the appearance of an edge annihilates a connected component, and thus adds a point to the persistence diagram, increasing or decreasing the distance between the two endpoints of the edge changes when the edge appears in the filtration, and thus when the components are merged, and thus the location of the corresponding point in the persistence diagram.

Building off the general method established in [11],[18] proposes methods that greatly reduces convergence time of gradient decent. Moreover, their methods extend back-propogation to provide gradient information to entire cycles, rather than pairs of simplices as previously.

[1] In practice, persistent homology based topological loss as is often used as regularization term; for instance, see [2], [17]

Related Sampling Based Methods

Previous sampling-based works have largely been focused on generating samples of persistence diagrams themselves, rather than sampling from some distribution of data which generates persistence diagrams. For instance, [1] proposes the use of Markov Chain Monte Carlo (MCMC) to sample from a model that intends to replicate a given persistence diagram in a statistically reliable way.

In a similar vein, the authors in [20] propose a framework for generating additional persistence diagrams, which they consider as pairwise interacting point processes, from a given collection for the purposes of augmenting available topological training data for use other machine learning algorithms. Whereas the proposed model of [1] primarily considers interactions between points within a given persistence diagram, the approach of [20] is twofold: they consider considering pairwise interactions between points to capture local behavior of a given persistence diagram, while simultaneously accounting for the spatial arrangement of persistence diagram points within a given persistence diagram, that is, their relative proximity to the diagonal, assigning greater importance to points with longer lifetimes.

[16] more closely aligns with the general premise of this work, in that they treat point clouds as a prior in a broader Bayesian framework which they propose for the purpose of preforming statistical inference on persistence diagrams, treated as Poisson point processes. Nevertheless, the authors make efforts to avoid working with point clouds directly, choosing instead to use persistence diagrams themselves as a proxy for point cloud data, under the perspective that the topological features present within a point cloud, those features which are encapsulated in persistence diagrams, are more informative than the point cloud data itself.

Other Related Works

Neither a sampling nor optimization based work, [7] explores the space of functions $f : [0, 1] \to \mathbb{R}$ which share an underlying persistence diagram. This is perhaps the work closest in spirit to ours, however, the scope of their work is restricted to function on the unit interval rather than images or weighted graphs, and the work does not rely on sampling methods. Instead, they rely on a notion of functional equivalence, "graph equivalence", to restrict the space of functions defined on the unit interval which share a common persistence diagram to a workable (enumerable) size, employing extensive use the elder rule and the elder rule.

Chapter 2: Background

2.1 Introduction to Persistent Homology

From a pragmatic point of view, persistent homology captures structural characteristics of a given topological space as a parameter of choice varies. The general idea is this: we gradually reconstruct our space in a discrete fashion, piece by piece, in accordance with our choice of parameter, and record the emergence or disappearance of topologically significant structural features, such as connected components, holes, voids and higher dimensional analogues along the way. A fixed parameter value provides a snapshot of the construction process; by amalgamating these snapshots, we not only gain insight into the types of structural features present within our space, but also gain the ability to differentiate between features that *persist*; those that are present in the snapshot at a variety of parameter values, and those which are not.

Naturally, persistent homology, along with other methods within the discipline of topological data analysis, is a useful in problems where the structural features it captures, such as holes, play a critical role. Before proceeding in more detail, we shall give an informal introduction to some of the concepts that make persistent homology mathematically rigorous.

Working with topological spaces is, in general, difficult from a computational perspective. For instance, it is difficult to rigorously codify the representation of the torus $\mathbb{T}^2$ as the unit square with opposite sides identified, or as the product $\mathbb{S}^1 \times \mathbb{S}^1$ within even the generous confines allowed by modern computers. It is therefore critical that we have a system of representing spaces, potentially topologically feature-rich spaces, in a way that is computationally feasible. To that end, vertices, edges and triangles are easy to work with from a computational perspective, and, while it is difficult to model the torus as a quotient space computationally, it is relatively simple to model the torus via triangulation. And crucially, from a topological perspective the triangulated torus is indistinguishable from the real thing. In order to make analysis of complex, potentially high dimensional spaces, or pragmatically, complex, high-dimensional data feasible, we will both extend and refine the familiar idea of triangulation to more

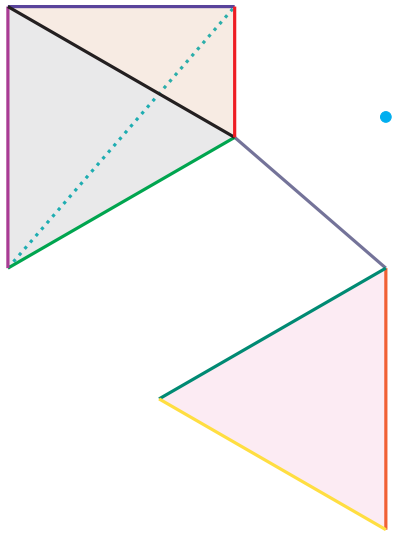
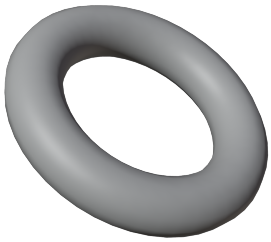
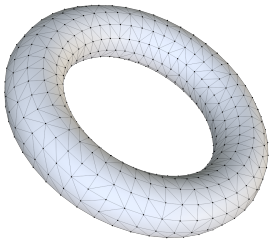


Figure 2.1



(a) A smooth torus.



(b) A triangulated torus.

Figure 2.2. Two representations of the torus

general contexts through *simplicies* (sing. *simplex*) and *simplicial complexes*;

Definition 2.1. *A k-simplex*

$$\sigma = [x_0, x_1, \ldots, x_k]$$

is the convex hull of an ordered set of $k+1$ points $x_0, \ldots, x_k \in \mathbb{R}^m$ in general position, where $m \geq k$. If σ is a k-simplex, then the convex hull of any non-empty subset of $\{x_0, \ldots, x_k\}$ defines a face of σ. The ordering $x_0, x_1, \ldots, x_k$ determines an orientation of σ.

Considering specific examples helps shed light on this definition; a 0-simplex is identified with a point, which, perhaps intuitively, has no faces. A 1-simplex is an edge; it

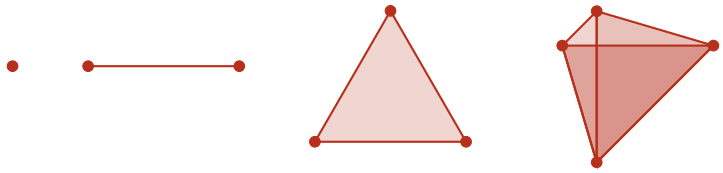
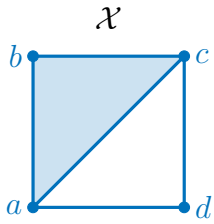


Figure 2.3. (Left to Right) - The model 0,1,2 and 3-simplices.



0-simplices: $\{a, b, c, d\}$

1-simplices: $\{ab, bc, cd, ad, ac\}$

2-simplices: $\{abc\}$

Figure 2.4. Examples of simplices and faces:

has two faces, namely, the two endpoints, which are themselves 0-simplices. And a two simplex, a filled triangle, has three 1-simplex faces, and 3-zero simplex faces (see Figure 2.4).

Simplices suffice as building blocks for any reasonable space. But what does it mean to "build" with these simplices? In three dimensions, we should expect our constructions to parallel the triangulation of the torus 2.2b. That example highlights some important properties that we should like to incorporate in our own constructions, namely, the construction is a collection of vertices, (straight) edges, and filled triangles, and that adjacent triangles intersect at an edge.

Definition 2.2. *A simplicial complex $\mathcal{X}$ is a collection of simplices* [1] *such that*

1. *For any simplex $\sigma \in \mathcal{X}$, all faces of σ are also contained in $\mathcal{X}$*
2. *For any two simplices σ, τ with nonempty intersection, $\sigma \cap \tau$ is a face of both σ and τ.*

Figure 2.5 shows examples of spaces that *fail* to be a simplicial complex. This work is primarily focused on weighted graphs, which carry a natural simplicial complex structure; we identify nodes with 0-simplices and edges with 1-simplices. Regardless, having defined what it means to be a simplicial complex, it now makes sense to consider maps between these structures:

[1] We assume when we write $\sigma = [v_0, \ldots, v_k]$ that the ordering of the vertices of σ $v_0, \ldots, v_k$ is the correct orientation of σ in $\mathcal{X}$

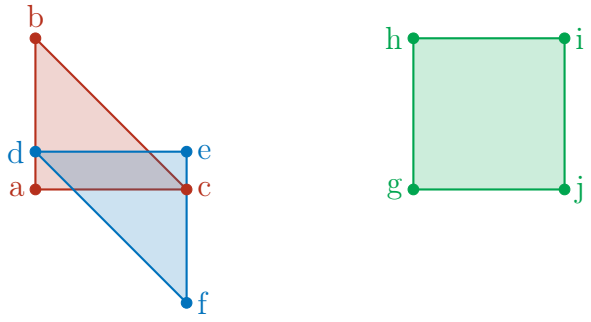


Figure 2.5. (Left) - This space fails condition (2), since the intersection of $[abc]$ and $[def]$ is not a face of either. (Right) - This space fails condition (1), since the filled square is not a simplex

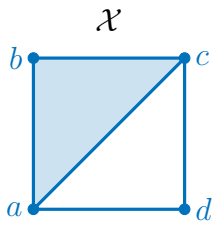
Definition 2.3. *A simplicial map $f : \mathcal{X} \to \mathcal{Y}$ is a map from the vertices (0-simplices) of $\mathcal{X}$ to the vertices of $\mathcal{Y}$, with the property that if $\sigma = [v_1, \ldots, v_n] \in \mathcal{X}$, the set $\{f(v_1), \ldots f(v_n)\}$ spans a simplex in $\mathcal{Y}$.*

Perhaps the most clear and relevant example of simplicial maps those induced by inclusion; a **subcomplex** $\mathcal{S}$ of a simplicial complex $\mathcal{X}$ is a collection of simplices in $\mathcal{X}$ that form a simplicial complex in their own right. By definition then, the inclusion $i : \mathcal{S} \to \mathcal{X}$ is a simplicial map, and maps of this type will play an important role when we define filtrations in a later section.

2.2 Chain Complexes and Homology

Having defined the types of structures with which we will work, simplicial complexes, we shall move on to further develop the framework which allows us to concord structural features present within them. While this work is focused mainly on detecting connected components, which are admittedly, plainly seen in the weighted graphs we are interested in, the ideas of this section generalize to detecting holes, voids, and higher dimensional analogues. The big idea is this: we can group simplices within a given simplical complex by dimension, assign to each of these collections a (free) vector space structure, and study relationships between these vector spaces vis-à-vis the behavior of linear maps that obey a particular composition condition. The most fundamental constructions in this section are *chain groups*:

Definition 2.4 (Chain Groups)**.** *Let $\mathcal{X}$ be a simplicial complex, and let $\mathbb{F}$ be a field. For $k \in \mathbb{N}$, the k^{th}-chain group, denoted $C_k(\mathcal{X})$, is the free $\mathbb{F}$-vector space over the set of k-simplicies in $\mathcal{X}$. By convention, we take $C_k = \{0\}$ for $k < 0$.*



0-simplices: $\{a, b, c, d\}$

1-simplices: $\{ab, bc, cd, ad, ac\}$

2-simplices: $\{abc\}$

$\mathbf{C_0}(\mathcal{X})$: $\{0, a, b, c, d, a+b, a+c, a+d, \ldots, a+b+c+d\}$

$\mathbf{C_1}(\mathcal{X})$: $\{ab, bc, cd, ad, ac, ab+bc, ab+ac, \ldots, ab+bc+cd+ad+ac\}$

$\mathbf{C_2}(\mathcal{X})$: $\{0, abc\}$

Figure 2.6. Example simplicial complex and corresponding chain groups

By assigning a simplicial complex this additional structure, in particular a vector space structure, we gain access to many of the tools, techniques and jargon of linear algebra. And naturally, having constructed a collection of vector spaces, we shall now move to discuss the maps between these spaces. Moreover, simplicial maps induce maps on chain groups in the following way - given a simplicial map $f : \mathcal{X} \to \mathcal{Y}$, we can define a map $\hat{f}_k : C_k(\mathcal{X}) \to C_k(\mathcal{Y})$ via

$$\hat{f}_k(\sigma) = \begin{cases} 0 & f(\sigma) \text{ is not a } k\text{-simplex} \\ f(\sigma) & \text{else} \end{cases}$$

and extending linearly [2]. On the other hand, restricting to a particular simplicial complex $\mathcal{X}$, we can also construct maps $C_k(\mathcal{X}) \to C_{k-1}(\mathcal{X})$. These maps, intuitively named *boundary maps*, are the star of the homology show:

Definition 2.5 (Boundary Maps)**.** *For $k \geq 1$, the kth boundary map $\partial_k : C_k(\mathcal{X}) \to C_{k-1}(\mathcal{X})$ is defined by*

$$\sigma = [v_0, v_1, \ldots, v_k] \mapsto \sum_{i=0}^{k} (-1)^i [v_0, \ldots, \hat{v}_i, \ldots, v_n]$$

where $[v_0, \ldots, \hat{v}_i, \ldots, v_k]$ is the $(k-1)$-face of σ spanned by $\{v_0, \ldots, v_k\} - \{v_i\}$, and extending linearly. Under the convention that $C_{-1}(\mathcal{X})$ is trivial, ∂_0 is define to be the zero map.

[2] If f is not injective on the vertices of σ, then $f(\sigma) = 0$

Taking $\mathbb{F} = \mathbb{F}_2$, the previous expression thus reduces from an alternating sum of $(k-1)$-faces to

$$f(\sigma) = \sum_{i=0}^{k} [v_0, \ldots, \hat{v}_i, \ldots, v_n]$$

Given that these ∂_k are linear maps, we can start to ask questions about kernels and images: A k-chain $x \in C_k(\mathcal{X})$ is called a *cycle* if $\partial_k(x) = 0$, and/or a *boundary* if $x = \partial_{k+1}(y)$ for some $y \in C_{k+1}(\mathcal{X})$.

This definition of boundary maps leads to an important property, quintessential to boundary maps both here, and in other contexts. Observe, for any k-simplex σ, we have

$$\begin{aligned}
\partial_{k-1}\partial_k(\sigma) &= \partial_{k-1}\left(\sum_{i=0}^{k}(-1)^i [v_0, \ldots, \hat{v}_i, \ldots, v_k]\right) \\
&= \sum_{j<i}(-1)^i(-1)^j [v_0, \ldots, \hat{v}_j, \ldots, \hat{v}_i, \ldots, v_k] \\
&+ \sum_{j>i}(-1)^i(-1)^{j-1} [v_0, \ldots, \hat{v}_i, \ldots, \hat{v}_j, \ldots v_k] \\
&= 0
\end{aligned}$$

since any term from the first summation will appear in the second with opposite sign. In other words, every element in the image of ∂_k is jointly in the kernel of ∂_{k-1}, $\mathrm{Im}(\partial_k), i.e \subseteq \mathrm{Ker}(\partial_{k-1})$. What do elements in the kernel of ∂_k, cycles, look like? Roughly, they correspond to loops, or *cycles*, in the simplicial complex. Continuing with our running example, one can easily see that $AB + BC + CA$ is an element in the kernel of ∂_1, so too is $AD + DC + CA$, or $AB + BC + CD + DA$.

Meanwhile, elements in the image of ∂_k, our boundaries, are formal combinations of faces of $(k+1)$-simplices, but the insight that composing boundary maps gives us the zero transformation gives us more; those chain group elements that in the image of a boundary map constitute the *boundary* of a filled in "hole". And the fact that $\partial_{k+1}\partial_k = 0$ allows us to quotient out elements that are jointly a boundary and a cycle. Hopefully, this informal discussion gives credence to the idea that homology is, roughly speaking, a way of detecting holes that are indeed holes, in that they are holes that are not "filled in".

2.3 k-th Simplicial Homology Group and Betti Numbers

Definition 2.6. *The k-th simplicial homology group of $\mathcal{X}$, denoted $H_k(\mathcal{X})$, is given by*

$$H_k(\mathcal{X}) = \mathrm{Ker}(\partial_k)/\mathrm{Im}(\partial_{k+1})$$

We refer to $\dim(H_k(\mathcal{X}))$ *is called the k-th Betti number of $\mathcal{X}$.*

We have seen that a simplicial map $f : \mathcal{X} \to \mathcal{Y}$ induces a map on chain groups, $\hat{f} : C_k(\mathcal{X}) \to C_k(\mathcal{Y})$. Simultaneously, the $C_k(\mathcal{X})$ are linked via boundary maps, so too are the $C_k(Y)$. In fact, there is an important link between simplicial maps, boundary maps, and maps on homology:

Lemma 2.7. *Let $f : \mathcal{X} \to \mathcal{Y}$ be a simplicial map, with induced map on chain groups $\hat{f}_k : C_k(\mathcal{X}) \to C_k(\mathcal{Y})$. Then*

1. *The following diagram commutes:*

$$\begin{array}{ccccccc}
\ldots & \xrightarrow{\partial_{k+2}} & C_{k+1}(\mathcal{X}) & \xrightarrow{\partial_{k+1}} & C_k(\mathcal{X}) & \xrightarrow{\partial_k} & \ldots \\
 & & \downarrow{\scriptstyle \hat{f}_{k+1}} & & \downarrow{\scriptstyle \hat{f}_k} & & \\
\ldots & \xrightarrow{\partial_{k+2}} & C_{k+1}(\mathcal{Y}) & \xrightarrow{\partial_{k+1}} & C_k(\mathcal{Y}) & \xrightarrow{\partial_k} & \ldots
\end{array}$$

2. *$\hat{f}_k : C_k(\mathcal{X}) \to C_k(\mathcal{Y})$ preserves cycles and boundaries. In other words, $\hat{f}_k$ induces a map $f_* : H_k(\mathcal{X}) \to H_k(\mathcal{Y})$.*

Proof. By linearity, it suffices to show equality $\partial_k \hat{f}_{k+1} = \hat{f}_k \partial_k$ on individual simplices. Let $\sigma = [v_0, \ldots, v_{k+1}] \in C_{k+1}(\mathcal{X})$. If $f(\sigma)$ is a $(k+1)$-simplex in Y, we have that f is injective on the vertices of σ and thus

$$\begin{aligned}
\partial_k\left(\hat{f}_{k+1}(\sigma)\right) &= \sum_{i=0}^{k} (-1)^i \left[f(v_0), \ldots \widehat{f(v_i)}, \ldots f(v_k)\right] \\
&= \sum_{i=0}^{k} (-1)^i \, \hat{f}_k([v_0, \ldots, \hat{v}_i, \ldots v_k]) \\
&= \hat{f}_k\left(\partial_k(\sigma)\right)
\end{aligned}$$

On the other hand, if $f(\sigma)$ is not a $(k+1)$-simplex, then f is not injective on vertices. If f maps the $k+1$ vertices of σ to $n < k$ vertices in $\mathcal{Y}$, then the image of every face

of σ under $\hat{f}_k$ is zero, and otherwise, f takes the $k+1$ vertices of σ to k vertices in $\mathcal{Y}$, so order the vertices of σ such that $f(v_0) = f(v_1)$. Observe now that

$$\begin{aligned} f([v_0, v_2, \ldots, v_k]) &= [f(v_0), f(v_2), \ldots, f(v_k)] \\ &= [f(v_1), f(v_2), \ldots, f(v_k)] \\ &= f([v_1, v_2, \ldots, v_k]) \end{aligned}$$

Other k-faces of σ, contain the vertices v_0 and v_1; $\hat{f}_k$ maps such faces to zero, since their image cannot be a k-simplex in $\mathcal{Y}$. Thus

$$\begin{aligned} \partial_k\left(\hat{f}_{k+1}(\sigma)\right) &= \sum_{i=0}^{k} (-1)^i \left[f(v_0), \ldots \hat{f(v_i)}, \ldots f(v_k)\right] \\ &= [f(v_0), f(v_2), \ldots, f(v_k)] - [f(v_1), f(v_2), \ldots, f(v_k)] = 0 \\ \hat{f}_k\left(\partial_k(\sigma)\right) &= \sum_{i=1}^{k} \hat{f}_k([v_0, v_1, \ldots, \hat{v}_i, \ldots v_k]) \\ &= [f(v_0), f(v_2), \ldots, f(v_k)] - [f(v_1), f(v_2), \ldots, f(v_k)] = 0 \end{aligned}$$

and thus $\partial_k \hat{f}_{k+1} = \hat{f}_k \partial_k$. Moreover, $\hat{f}_k$ preserves boundaries and cycles, for if $\tau = \sum \alpha_i \sigma_i$ is a k-cycle, then $\partial_k(\tau) = 0$, and thus

$$(\partial_k \hat{f}_k)(\sigma) = \hat{f}_{k-1}\left(\partial_k(\sigma)\right) = \hat{f}_{k-1}(0) = 0$$

and if $\eta = \sum \alpha_j \sigma_j$ is a k-boundary, say $\eta = \partial_{k+1}(\tau)$, then

$$\hat{f}_k(\eta) = (\hat{f}_k \partial_{k+1})(\tau) = \partial_{k+1}(\hat{f}_{k+1}(\tau))$$

and so $\hat{f}_k(\eta)$ is a k-boundary as well. Thus $\hat{f}_k$ induces a well-defined map on homology $f_{*k} : H_k(\mathcal{X} \to H_k(\mathcal{Y})$,

$$H_k(\mathcal{X}) \ni [\sigma] \mapsto [\hat{f}_k(\sigma)] \in H_k(\mathcal{Y})$$

□

2.4 Filtrations

At this point, we have developed the mathematical foundation for the "snapshot" portion of persistent homology: given a simplicial complex $\mathcal{X}$, calculating, say, $H_0(\mathcal{X})$, allows us to both count and identify representative vertices for connected components in $\mathcal{X}$.

Certainly, the ability of homology to identify connected components, holes, and

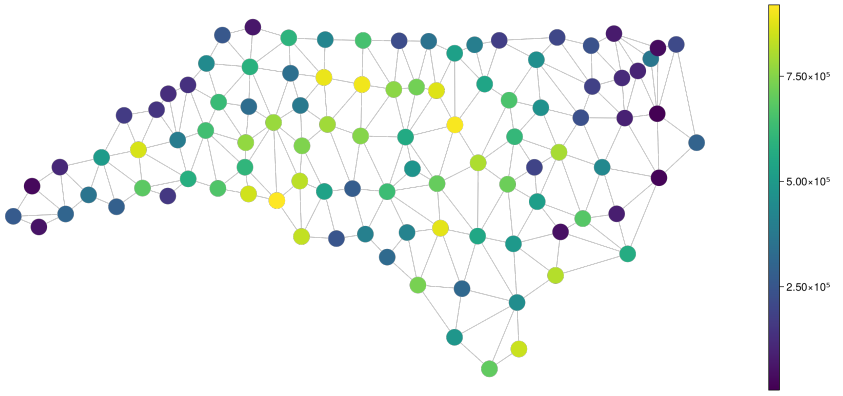


Figure 2.7. Example of geospatial data represented with a graph structure. In this case, each node represents a county, with node coloration representing the population. The data used to generate this graph was taken from [8].

higher dimensional analogues is notable in and of itself. With the aid of filtrations, the "persistent" part of persistent homology, we can say more, extending the ideas of homology to work with structural features induced by *functions* defined on a topological space. This is particularly relevant in domains where the structural characteristics of the space itself is not of particular interest. For instance, geospatial data (see Figure 2.7) is often represented by graphs, where nodes representing a tangible geographic feature, such as cities, are endowed with some value, say population. While the macro graph structures can be of interest in such cases, often, we are often more interested in studying relationships and patterns that arise from the node values, such as the location of local minima or maxima. To extend the idea of homology to functions defined on a simplicial complex, in our case, graphs with the obvious simplicial complex structure, we will need to define a few more bits of terminology:

Definition 2.8 (Sublevel Set)**.** *Let $\mathcal{X}$ be a simplicial complex, and $f : Vert(\mathcal{X}) \to \mathbb{R}$ be a (not necessarily continuous) function. For $t \in \mathbb{R}$, the $t-$sublevel set, denoted $\mathcal{X}_t(f)$ is given by*

$$\mathcal{X}_t(f) = \{\sigma \in \mathcal{X} \mid f(v) \leq t \text{ for all vertices } v \in \sigma\}$$

Each $\mathcal{X}_t(f)$ is clearly a subcomplex of $\mathcal{X}$. Furthermore, for $s < t$, there is a simplicial map $i_s^t : \mathcal{X}_s(f) \to \mathcal{X}_t(f)$, the inclusion map, which in turn induces a maps on homology

as indicated in the following diagram:

$$\begin{array}{ccccccc} \ldots & \longrightarrow & \mathcal{X}_s(f) & \xrightarrow{i_s^t} & \mathcal{X}_t(f) & \longrightarrow & \ldots \\ & & \downarrow & & & & \\ \ldots & \longrightarrow & H_k(\mathcal{X}_s(f)) & \xrightarrow{(i_s^t)_*} & H_k(\mathcal{X}_t(f)) & \longrightarrow & \ldots \end{array}$$

Definition 2.9 (Sublevel Set Filtration)**.** *A sublevel set filtration is an ordered sequence $a_1 < a_2 < \cdots < a_r < \ldots$ together with a sequence of simplicial complexes*

$$\mathcal{X}_{a_1} \subseteq \mathcal{X}_{a_2} \subseteq \cdots \subseteq \mathcal{X}_{a_r} \subseteq \ldots$$

ordered by inclusion.

In practice, filtered simplicial complexes often arise from sublevel sets; if $\mathcal{X}$ is a simplicial complex, with $f : \mathcal{X} \to \mathbb{R}$, given any ordered sequence of real numbers $a_1 < a_2 < \ldots$ we can construct a *sublevel set filtration*, by taking sublevel sets:

$$\mathcal{X}_{a_1}(f) \subseteq \mathcal{X}_{a_2}(f) \subseteq \ldots$$

The inclusions $i_s^t : \mathcal{X}_s \to \mathcal{X}_t$ are simplicial maps, hence induce linear maps on homology i_{*s}^t, with

$$i_{*r}^s \circ i_{*s}^t = i_{*r}^t \quad r \leq s \leq t$$

Such pairings of homology groups $H_k(\mathcal{X}_s)$ and maps i_s^t satisfying the composition rule above are referred to as *persistence modules*. Note that we have placed no restrictions on either the dimension of the homology groups $H_k(\mathcal{X}_s)$, or required the sequence $\cdots \to H_k(\mathcal{X}_s) \to \ldots$ terminates, in the sense that there exists $s \in \mathbb{R}$ such that

$$i_s^t : H_k(\mathcal{X}_s) \to H_k(\mathcal{X}_t)$$

is an isomorphism for all $t \geq s$. If both of these conditions are true, however, we say that the module is of *finite type*; these are the vast majority of modules that arise in practice. Another useful bit of terminology associated with persistence modules is the idea of *tameness*; a persistence modules is said to be *tame* if, for all $s \in \mathbb{R}$, there exists some small half open inverval $[s, t)$ such that the maps

$$i_s^t : H_k(\mathcal{X}_s) \to H_k(\mathcal{X}_r), \quad s \leq r < t$$

are isomorphisms.

We would like to extract a bit more information from these persistence modules;

namely, in order to synthesize the homological perspectives from the various $\mathcal{X}_k$ into a singular, unified perspective, we would like these inclusions to either preserve or annihilate equivalence classes of k-cycles; a representative k-cycle for particular topological feature, say a connected component, should remain as such for as long as the feature is present in the sublevel set filtration, and fact that we may do so underpins all of persistent homology.

Theorem 2.10 (Fundamental Theorem of Persistent Homology [24])**.** *Let*

$$\mathcal{X}_{r_1} \xrightarrow{i_1} \mathcal{X}_{r_2} \xrightarrow{i_2} \dots \xrightarrow{i_{n-1}} \mathcal{X}_{r_n}$$

be a filtered simplicial complex, with induced maps on homology $i_{*1} \dots i_{*(n-1)}$. For each dimension k, we can choose a basis $\mathcal{B}_{r_j}$ for $H_k(\mathcal{X}_{r_k})$ such that, for each $b \in \mathcal{B}_{r_j}$, either

1. $i_{*j}(b) = 0$
2. $i_{*j}(b) \in \mathcal{B}_{r_{j+1}}$, *and if* $i_{*j}(b) = i_{*j}(c) \neq 0$ *for some* $c \in \mathcal{B}_{r_j}$, *then* $b = c$

In essence, the Fundamental Theorem of Persistent Homology gives us a sequence of bases that are consistent from the point of view of homology; consistent in the sense that we may choose a basis that yields a consistent pairing between topological features and corresponding representative class in homology over the entirety of a filtration. Once we have chosen a suitable basis, we can compare, like-for-like, representatives for a topological feature within homology groups across different filtration values; and we want to unambiguously interchange between speaking of topological features, (holes, connected components, the like) and the labels prescribed by homology, i.e the corresponding equivalence class.

As filtered simplicial complexes change, topological features emerge and disappear along with these changes. A feature $[\tau]$. is said to be *born* at r_s if it is not in the image of i_{*s}. On the other hand, a feature is said to *die* at r_s if s is the smallest index for which, either

1. $i_{*s}([\tau]) = 0$, i.e the feature is annihilated at r_s.
2. $i_{*s}([\tau]) = i_{*s}[\sigma]$ for some $[\sigma] \in \mathcal{B}_s$, i.e the feature merges with an older feature σ at r_s. In this case, $[\tau]$ is destroyed, but the older feature σ persists.

The *lifetime* of a feature born at b and destroyed at d is $d - b$ [3]. Note that a feature may have infinite lifetime.

In Figure 2.8, there is a clear hierarchy of equivalence classes in terms of birth

[3]Note $d - b \geq 0$, since a feature cannot be annihilated before it is created!

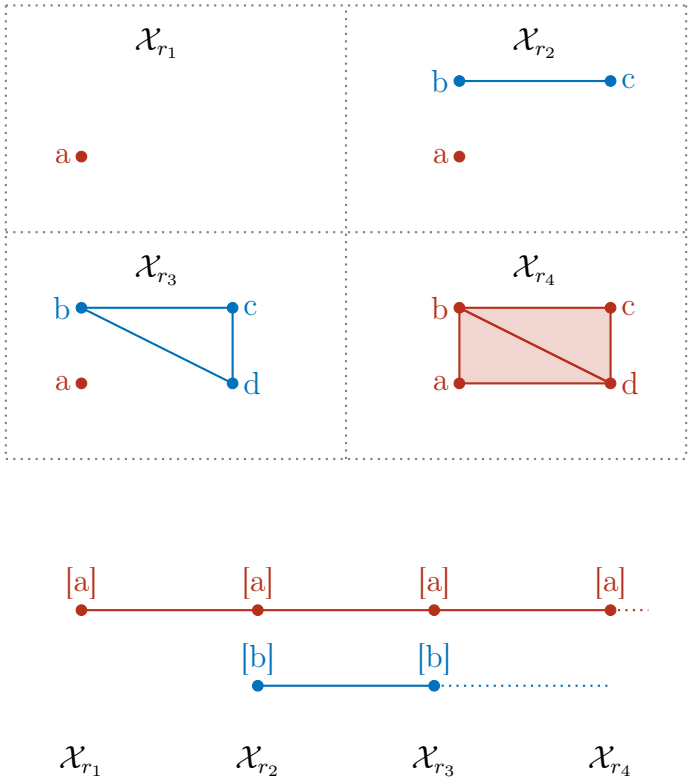


Figure 2.8. (Top) A filtered simplicial complex - (Bottom) - One such possible choice of bases for H_0, guaranteed by the fundamental theorem of persistent homology, with the property that equivalence classes are preserved or annihilated by the maps on homology induced by the inclusion maps in the filtration. In 0th homology, equivalence class $[a]$ is first to appear, and so following the elder rule, it is the component that is preserved when $[a]$ and $[b]$ merge in $\mathcal{X}_{r_4}$.

times; $[a]$ is the first class to appear, and as a result, $[a]$ rather than $[b]$ is the class that is preserved when the two classes merge in $\mathcal{X}_{r_4}$. The convention to preserve the oldest (in terms of birth time) feature when two features are merged is referred to as *the elder rule.* [4]

Having constructed a persistence modules $(H_k(\mathcal{X}_s), i^t_{*s})$, and having subsequently chosen a suitable basis for each homology group $H_k(\mathcal{X}_{r_j})$ in accordance with the Fundamental Theorem of Persistent Homology, we can answer questions as to how many features persist over a given interval $[s,t]$ by examining the rank of the linear map i^t_{*s}; if this map is not of full rank, then some feature present in $\mathcal{X}_s$ is not in $\mathcal{X}_t$, and thus must have merged with another component or been annihilated at some $\mathcal{X}_r$ for $r \in (s,t]$. Following the insights of [12], ranks serve as a persistent analogue of Betti numbers; they tell us how many *bars* like those seen in the bottom of Figure 2.8, are present in a given interval I, and by proxy, how many k-dimensional topological features are present in the simplicial complexes $\{\mathcal{X}_r\}_{r\in I}$. And whether one is interested in detecting local extrema, connected components, holes, or other higher dimensional analogues, the lifetime of a feature, is a descriptor for the feature's importance in the overall structure of the space.

2.5 Persistence Barcodes and Diagrams

Given that we are able to frame much of persistent homology purely in terms of ranks of linear maps, the intervals upon which these maps have full rank, and those that do not, to fully solidify the idea of *persistence bar-codes*, and later, of *persistence diagrams*, we need a more succinct summery of the topological features captured by persistent homology than what is offered by persistence modules as defined. In some sense, there is a great deal of superfluous information built into persistence modules; we need not care about the ranks of *all* maps between homology groups; only the ones which provide meaningful information about the birth and death of topological features. If the types of topological invariants that persistent homology detects are enough to distinguish spaces, we might expect that information about the appearance/annihilation of these features is enough to characterize a persistence module. To that end, we will follow [21] in recalling a special case of persistence modules, *interval modules* and the presentation of an important theorem:

Definition 2.11. *Given a half open interval $(a,b]$, an interval module $I[a,b]$ is a*

[4] It is possible that two or more classes may be born simultaneously; in this case the elder rule is not well-defined. In practice, we tend not to care about such instances; different choices will yield identical barcodes.

persistence module, with

$$I[a,b]_t = \begin{cases} \mathbb{F}, & t \in (a,b] \\ 0 & else \end{cases} \qquad i_s^t = \begin{cases} id_{\mathbb{F}}, & s,t \in (a,b],\ s \leq t \\ 0 & else \end{cases}$$

Theorem 2.12 (Normal Form Theorem [5])**.** *Let $(H_k(\mathcal{X}_s), i_s^t)$ be an $\mathbb{R}$-index persistence module of finite type. Then there exists a finite collection $(I_\alpha, m_\alpha)_{\alpha=1}^N$ of intervals $I_\alpha = (b_\alpha, d_\alpha]$ or $I_\alpha = (b_\alpha, \infty)$ with multiplicities m_α such that*

$$(H_k(\mathcal{X}_s), i_s^t) \cong \bigoplus_{\alpha=1}^{N} I_\alpha^{m_\alpha}[b_\alpha, d_\alpha]$$

The elements of this sum, I_α are called bars.

It is no coincidence that the intervals that are referenced in Theorem 2.12 are expressed in terms of b's and d's; the Normal Form Theorem allows us to completely characterize persistence modules arising from filtered simplical complexes purely in terms of the births and deaths of topological features, with each interval module $I[b,d]$ corresponding to a topological feature that persists on the interval $(b,d]$. The unique interval module decomposition guaranteed by the Normal Form Theorem is referred to as the *barcode* of the filtered simplicial complex $\{\mathcal{X}_s\}$; Barcodes have an equivalent representation, the persistence diagram; recall that "bars" in the barcode are representative of a particular topological feature present in the filtration. The left endpoint of a given bar represents the filtration value b at which the feature appeared, and the corresponding right endpoint, if it exists, represents the filtration value d at which the feature is annihilated. Thus instead of representing a feature with a bar, we can represent a feature with an ordered pair (b,d), and construct a plot of these points. Such plots are called *persistence diagrams* and, although we shall use the terms "persistence barcode" and "persistence diagrams" interchangeably owing to the clear correspondence between the two, it is worth mentioning differences in how the relative significance of a feature is manifested between the two. In persistence barcodes, the relative significance of a feature is seen through the length of its corresponding bar. On the other hand, the relative significance of a feature is seen through its vertical distance from the diagonal in persistence diagrams; nevertheless, characterizations are given by the lifetime of the feature, $d - b$.

2.6 Wasserstein Distance

Notions of "distance" are an integral part of mathematics, both the pure and applied spheres; the same is certainly true within the realm of persistent homology, particularly with regards to persistence diagrams, viewed as mathematical objects unto themselves.

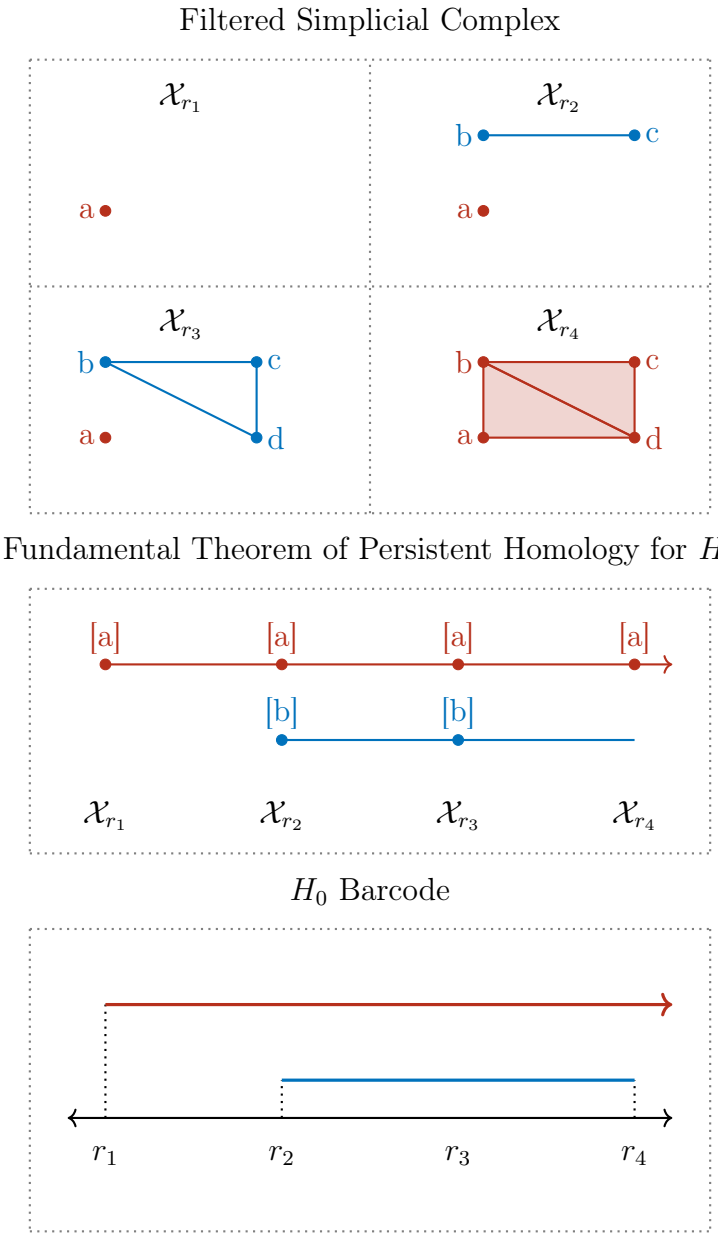


Figure 2.9. The barcode is the last piece of the puzzle in the persistent homology pipeline; the Normal Form Theorem 2.12 guarantees its existence and uniqueness for a filtered simplicial complex, and it provides a digestible summary of all topological features detected by homology at various filtration values. In this instance, the blue bar is finite; it represents the connected component labeled with basis element $[b]$ that appears at filtration value r_2, and is merged with the connected component labeled $[a]$ at r_4 according to the elder rule. Meanwhile, the component associated with $[a]$ is created at filtration value r_1, and is never merged or destroyed; it "dies" at ∞.

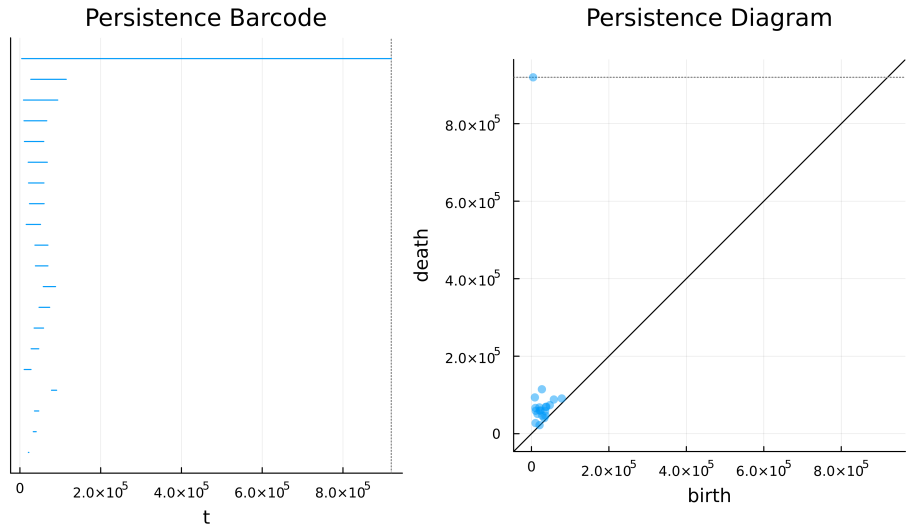


Figure 2.10. The 0-th dimensional persistence barcode (left) and persistence diagram (right) for the graph in Figure 2.7. By convention, we draw a "line at infinity" in persistence diagrams to represent a topological feature with infinite lifetime; this is the horizontal gray dotted line in the persistence diagram.

Defining a distance which meaningfully captures differences and similarities between persistence diagrams is non-trivial. If one is to define a distance on the space of persistence diagrams, they must contend with the fact that persistence diagrams come in all sizes, from diagrams with one point, to diagrams with $10,000$ points, to diagrams with one point with multiplicity $10,000$; all are possible, and any distance defined on the space of persistence diagrams must be able to cope with these different possibilities. More important, such a distance should reflect one of the key heuristics of persistent homology; that a point's lifetime is inherently related to its relative significance, even when persistence diagrams are studied in the absence of any underlying association with topological spaces. To that end, we should like to regard the persistence diagrams in Figure 2.11 as being similar.

Although it is difficult to see how we might define a distance on the space of persistence diagrams, it is not difficult to compute distances for pairs of *points* between two diagrams; persistence diagram points are merely points in $\mathbb{R}^2$, any of the distances induced by the p-norms will suffice. From this point of view, given persistence diagrams [5]

$$\mathcal{P}_X = \{p_1, \ldots, p_N\}, \quad \mathcal{P}_Y = \{p'_1 \ldots p'_N\}$$

[5]We are being a bit sloppy with notation here for the purposes of motivating the Wasserstein distance; persistence diagrams in general are not sets, but *multisets*.

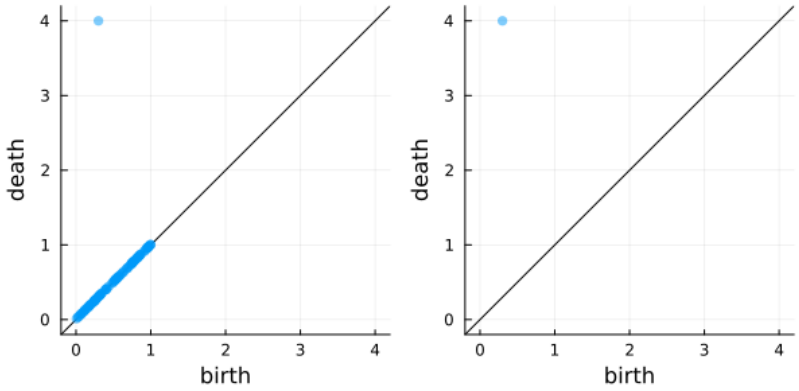


Figure 2.11. These two persistence diagrams differ drastically in terms of the number of points; the diagram on the left has 101, and the diagram on the left has 1. Given the heuristic of associating a point's lifetime with its relative significance however, we should like any distance defined on the space of persistence diagrams to value these diagrams as "close".

each with the same number of points, we can define a distance between $\mathcal{P}_X$ and $\mathcal{P}_Y$ to be the minimal matching cost of between the two diagrams;

$$d(\mathcal{P}_X, \mathcal{P}_Y) = \inf_{\phi:\mathcal{P}_X \to \mathcal{P}_Y} \left(\sum_{i=1}^{N} ||p_i - \phi(p_i)||_q^q \right)^{1/q}$$

where the infimum is take over all bijections $\phi : \mathcal{P}_X \to \mathcal{P}_Y$. Such a notion of distance generally enforces that high persistence points in one diagram are to be matched to high persistence points in the other, and as such, informatively captures differences and similarities between high persistence features in the spaces underlying $\mathcal{P}_X$ and $\mathcal{P}_Y$. On the other hand, persistence diagrams do not all have the same number of points; we must work a bit more to ensure that a matching between persistence diagrams, even those with different cardinalities, is possible. To that end, we will adopt the convention that points within either diagram may be matched to the *diagonal* Δ, where

$$\Delta = \{(a, a) \,|\, a \in \mathbb{R}\}$$

and additionally, that points on the diagonal exist with infinite multiplicity. Intuitively, points close to the diagonal are of relatively low significance from the perspective of persistent homology. If such points happen to be paired as part of an optimal matching,

so be it; otherwise, their contribution to the overall distance between persistence diagrams should be small, since the relative significance of their underlying feature to their underlying space is small. This is the idea behind the q-Wasserstein distance, W_q, and the bottleneck distance, W_∞.

Definition 2.13 (Wasserstein and Bottleneck Distances [23])**.** *Let $\mathcal{P}_X$ and $\mathcal{P}_Y$ be persistence diagrams. The qth-Wasserstein distance, W_q, is given by*

$$W_q(\mathcal{P}_X, \mathcal{P}_Y) = \inf_{\phi:\mathcal{P}_X\cup\Delta\to\mathcal{P}_Y\cup\Delta} \left(\sum_{p_i\in\mathcal{P}_X\cup\Delta} ||p_i - \phi(p_i)||_q^q \right)^{1/q}$$

where $\phi : \mathcal{P}_X \cup \Delta \to \mathcal{P}_Y \cup \Delta$ is a matching. By convention, we take $\infty - \infty = 0$. W_∞ is referred to as the bottleneck distance, and is defined similarly:

$$W_\infty(\mathcal{P}_X, \mathcal{P}_Y) = \inf_{\phi:\mathcal{P}_X\cup\Delta\to\mathcal{P}_Y\cup\Delta} \sup_{p_i\in\mathcal{P}_X\cup\Delta} ||p_i - \phi(p_i)||_\infty$$

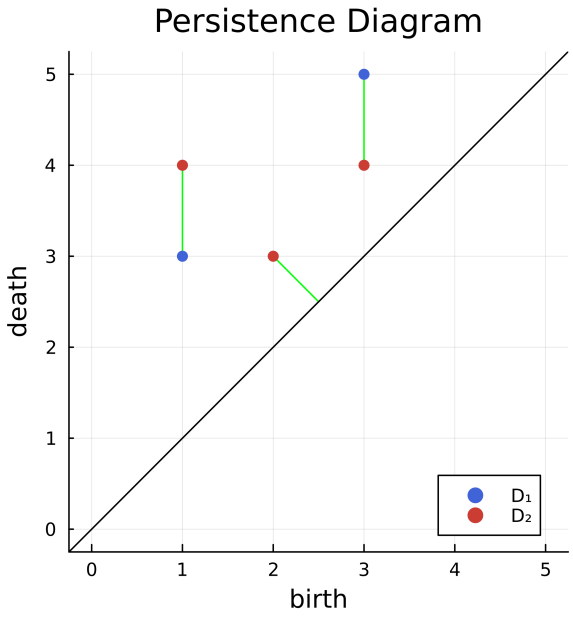
The sum in Definition 2.13 can be though of in three parts;

1. Points in $\mathcal{P}_X$ that are matched to points in $\mathcal{Y}$ under ϕ.
2. Points $p = (b, d) \in \mathcal{P}_X$ that are matched to the diagonal Δ (with cost $\frac{d-b}{2^{(q-1)/q}}$).
3. Points $p' = (b', d') \in \mathcal{P}_Y$ that are matched to the diagonal Δ (with cost $\frac{d'-b'}{2^{(q-1)/q}}$).

2.7 Stability Results

Overcoming the harmful effects of noise is a fundamental problem in data science and machine learning; when we build models, we expect that they will exhibit some degree of stability. If our model is worth its salt, we should be able to make small changes to inputs, whether adversarial or unintentionally, as an artifact of the data collection process, and get roughly proportional small changes in outputs; if we are building a classifier, changing a single pixel within the image of a cat should not substantially impact the result of classifier.

To conclude this overview of persistent homology, we will give an overview of one of the most important facets of the study of persistent homology and persistence diagrams; the stability of persistence diagrams with respect to the bottleneck and q-Wasserstein distances



[Example of optimal matching for the 1-Wasserstein distance.]Two persistence diagrams, D_1 and D_2, along with an optimal matching between the two with regards to W_1; In this case, $W_1(D_1, D_2) = 3$

Theorem 2.14 (Bottleneck Stability Theorem [4]). *Let $\mathcal{X}$ be a triangulable metric space, and let $\mathcal{F}$ and $\mathcal{G}$ be finite-type persistence modules induced by the sublevel set filtations of $f, g : \mathcal{X} \to \mathbb{R}$ respectively. Then*

$$W_\infty(Dgm(\mathcal{F}), Dgm(\mathcal{G})) \leq ||f - g||_\infty$$

Theorem 2.15 (Wasserstein Stability Theorem [23]). *Let $\mathcal{X}$ be a CW-Complex, and let $f, g : \mathcal{X} \to \mathbb{R}$ be monotone functions. Then*

$$W_q(Dgm(\mathcal{F}), Dgm(\mathcal{G})) \leq ||f - g||_{L_q}$$

where $||f - g||_{L_q}$ is the L_q distance,

$$||f - g||_{L_q} = \sqrt[q]{\sum_{\sigma \in \mathcal{X}} \max_{x \in \sigma} |f(x) - g(x)|^q}$$

These results give the following guarantee; if one makes small change to a function f defined over a simplicial complex $\mathcal{X}$ this will result in a small(er) change on the level of persistence diagrams. From a pragmatic point of view, this results means the types of topological features within data that are captured by persistent homology are robust to noise.

Graph Laplacians

Graph Laplacians are relatively easy to define; given a finite simple graph $G = (V, E)$, the *Graph Laplacian of* G is defined by the matrix $L = D - A$, where D is the degree matrix of G, and A is the adjacency matrix of G, with

$$A_{ij} = \begin{cases} 1 & \text{there is an edge between vertices } i and j. \\ 0 & \text{else} \end{cases}$$

This definition, though simple, does little to motivate the connection between the graph Laplacian and the Laplacian operator from multivariable calculus, nor does it ostensibly motivate why the eigenvectors of the graph Laplacian enjoy the properties they do. In this section, we will briefly speak to the connection between the Laplacian operator of calculus and the graph Laplacian, mainly following [22], and additionally, discuss some of the properties the graph Laplacian enjoys.

Connecting the Graph Laplacian to the Laplacian Operator

Given a twice continuously differentiable scalar function $f : \mathbb{R}^n \to \mathbb{R}$, the Laplacian of f is the divergence of the gradient of f,

$$\Delta f = \nabla \cdot \nabla f = \sum_{i=1}^{n} \frac{\partial^2 f}{\partial x_i^2}$$

How can we transplant this definition to a graph $G = (V, E)$, on which we may define functions, but where there is no clear analogue of "gradient" or "directional derivative"? For starters, we can discretize the derivative when working with graphs; given an edge $e = (u, v)$, we can define

$$\frac{\partial f(u)}{\partial v} = f(v) - f(u)$$

The gradient at vertex u, $\nabla_u f$, is simply the row vector of all such partials where e is an edge emanating from u:

$$\nabla_u f = \left[\frac{\partial f(u)}{\partial v} \quad \forall e \in E \text{ s.t } e = (u, v) \text{ for some } v \in V \right]$$

Defining ∇ as such has two important implications; for finite graphs, ∇ has a convenient matrix representation. If we fix an orientation for all edges $e_k \in E$, and an ordering $e_1, \ldots, e_n$, we can define the $|V| \times |E|$ *incidence matrix* $\mathcal{E}$ of G, with rows indexed by vertices, columns indexed by edges, and with

$$\mathcal{E}_{ik} = \begin{cases} 1 & (i, j) = e_k \text{ for some } j \in V \\ -1 & (j, i) = e_k \text{ for some } j \in V \\ 0 & \text{else} \end{cases}$$

and defining $\mathcal{E}$ as such allows us to write $\nabla f = \mathcal{E}^T f$. Secondly, observe that ∇ defines an operator, which takes in functions f defined on the *vertices* of G, and outputs ∇f, a function defined on the *edges* of G, via

$$(\nabla f)(u, v) = \frac{\partial f(u)}{\partial v}$$

We can extend the notion of divergence to graphs as well; in Euclidean space, divergence operator works over vector fields, measuring the flux through an infinitesimal volume about a point. On G, therefore, we expect divergence should measure the net outflow of an edge function at each vertex in G. If ∇ is an operator from vertex functions of G to edge functions of G, we should expect divergence to be an operator in the other direction; from edge functions to vertex functions on G. What could be more natural than the dual of ∇? In this case, we have an explicit matrix representation of divergence; it is given by the incidence matrix E.

At this point, we should verify that our intuition aligns with the standard definition for the graph Laplacian; in other words;

Lemma 2.16. $\mathcal{E}\mathcal{E}^T = \mathcal{L}$

Proof. Obeserve that

$$\mathcal{E}_{ik}\mathcal{E}_{jk} = \begin{cases} 1 & i = j, \quad (i,j) \text{ is the } k\text{th edge of } G \\ -1, & i \neq j, \quad (j,i) \text{ is the } k\text{th edge of } G \\ 0 & \text{else} \end{cases}$$

and so

$$(\mathcal{E}\mathcal{E}^T)_{ij} = \sum_k \mathcal{E}_{ik}\mathcal{E}_{jk} = \begin{cases} \sum_{k|i\in e_k} 1 = \deg(i), & i = j \\ -1, & i \neq j, \quad (i,j) \in E \\ 0 & \text{else} \end{cases}$$

which is precisely $\mathcal{L}$. □

We knew that $\mathcal{L}$ was symmetric, as the difference of two symmetric matrices, but this lemma tells more; namely, that $\mathcal{L}$ is positive semi-definite.

Graph Laplacian Eigenvectors

Eigenvectors, or eigenfunctions, of graph Laplacians play a fundamental role in this work and, as such, it is worth examining some of their properties. Notably, as a

positive semi-definite matrix $\mathcal{L}$ possesses an orthonormal basis of eigenvectors, each with corresponding non-negative eigenvalue. It is worth mentioning that, although there are few closed expressions for eigenvectors of simple graphs in general, zero is an eigenvalue of any graph Laplacian $\mathcal{L}$, with corresponding eigenvector $\mathbf{1} = [1, 1, \ldots, 1]$; in fact, the multiplicity of the eigenvalue 0 is gives the number of connected components of G.

More broadly, graph Laplacian eigenvectors play a fundamental role in connected ostensibly unrelated properties of the underlying graph G; the "smoothness" of functions defined on the nodes of G, the connectivity of G, and graph partitioning. For starters, it's not too difficult to see that

$$(f^T \mathcal{L} f)(i) = \sum_{j \sim i} (f(i) - f(j))^2$$

which, using terminology from [22], is a measure of the "local smoothness" of f; it is large when $f(i)$ differs greatly from $f(j)$ where $i \sim j$, and small when $f(i) \approx f(j)$ for $i \sim j$. Minimizing $f^T \mathcal{L} f$ is a well studied problem in linear algebra; in general, quotients of the form

$$\frac{x^T A x}{x^T x}$$

are referred to as *Reyleigh Quotients*, and for real symmetric matrices, which include graph Laplacians, we have a special case of the Courant-Fischer Theorem to work with:

Theorem 2.17 (Courant-Fischer Theorem)**.** *Let A be a symmetric real $n \times n$ matrix with eigenvalues $\lambda_1 \leq \cdots \leq \lambda_n$, and corresponding eigenvectors $v_1, \ldots v_n$. Then*

$$\begin{aligned}
\lambda_1 &= \min_{||x||=1} x^T A x = \min_{x \neq 0} \frac{x^T A x}{x^T x} \\
\lambda_2 &= \min_{\substack{||x||=1 \\ x \perp v_1}} x^T A x = \min_{\substack{x \neq 0 \\ x \perp v_1}} \frac{x^T A x}{x^T x} \\
&\vdots \\
\lambda_n &= \max_{||x||=1} x^T A x = \max_{x \neq 0} \frac{x^T A x}{x^T x}
\end{aligned}$$

The proof of this result is beyond the scope of this work; see [14] for reference. Nevertheless, the relationship between the magnitude of an eigenvalue and the "smoothness" of its corresponding eigenfunction is now firmly established. From the perspective of persistent homology, "smoothness" means that any local extrema present within

Laplacian eigenfunctions, the very things which persistent homology detects, will be manifested as points close to the diagonal in persistence diagrams. On that note, we can be more specific about the possible values that their local extrema may take;

Lemma 2.18. *Eigenfunctions of a graph Laplacian matrix $\mathcal{L}$ cannot have non-negative local minima or non-positive local maxima.*

Proof. We show that eigenfunctions of $\mathcal{L}$ cannot have non-negative local minima; the other direction follows similarly. BWOC, suppose that eigenfunction f has a positive local minima at i. Then $(\mathcal{L}f)(i) = \lambda f(i) > 0$. On the other hand,

$$(\mathcal{L}f)(i) = \sum_{j \sim i} f(i) - f(j) \leq 0$$

a contradiction. □

2.8 Random Walk Metropolis

Sampling from a continuous probability distribution with unknown probability density function (pdf) is a fundamental problem in statistics. Often, we have a rough idea of how our unknown distribution ought to be parameterized, and we may even get as far as writing down an explicit expression for the unknown density function. Naturally, the expressions we propose must fulfil the most basic of requirements for continuous probability distributions; they must integrate to 1, and computing the integral for our proposed expressions may be utterly intractable. Problems of this type may be formulated more rigorously; given that $p(x) \propto \tilde{p}(x)$, where $\tilde{p}$ is known, how can we sample from p?

The Random Walk Metropolis Algorithm (RWM)[6], although not necessarily a sampling method method in and of itself, is well-suited for situations of this type. Under RWM, we are not forced to preform calculations on the unknown distribution p directly; we only require use of the pseudo-likelihood function $\tilde{p} \propto p$, and a chosen probability distribution q satisfying

$$q(y \,|\, x) = q(x \,|\, y)$$

Essentially, RWM iteratively constructs a Markov Chain $\{X_n\}_{n=0,1,2,\ldots}$ whose distribution, in the long term, approximates p. The first phase of RWM is the initialization phase; a point X_0 is chosen arbitrarily, and we compute $\alpha = \tilde{p}(X_0)$. The second phase is iterative; for each step i where $i = 0, 1, 2, \ldots$ we preform the following:

1. **Generation of Candidate State:** Given the Markov Chain is at state X_i, a new state $\tilde{X}$ is proposed according to the distribution $q(\tilde{X} \,|\, X_i)$.

[6]RWM is sometimes referred to as the Metropolis-Hastings Algorithm

2. **Acceptance/Rejection of Candidate State:** Let

$$\alpha(\tilde{X}, X_i) = \frac{\tilde{p}(\tilde{X}) \cdot q(\tilde{X} \mid X_i)}{\tilde{p}(X_i) \cdot q(X_i \mid \tilde{X})} = \frac{\tilde{p}(\tilde{X})}{\tilde{p}(X_i)}$$

Let $b \sim \mathcal{U}[0, 1]$, that is, b is randomly chosen from the unit uniform distribution. If $b < \alpha(\tilde{X}, X_i)$, we *reject* the proposed move; otherwise, we *accept* the proposed move.

3. **Update the Markov Chain:** Update the Markov Chain

$$X_0, \ldots, X_i$$

by setting the $(i + 1)$th state of the Markov Chain via

$$X_{i+1} = \begin{cases} X_i & \text{the proposed move was rejected.} \\ \tilde{X} & \text{the proposed move was accepted.} \end{cases}$$

Chapter 3: Methods

Persistence diagrams, viewed pragmatically as a tool rather than an object of mathematical interest, enjoy many desirable properties; the stability of persistence diagrams with respect to the various p-Wasserstein and Bottleneck distances and the invarience of persistence diagrams to rotational or translational transformations are practically useful properties; both contribute to persistent homology's position as a noise-agnostic tool within the broader realm of data science. Unfortunately, if one wishes to traverse the TDA pipeline *backwards*, i.e to take a persistence diagram, and deduce what kind of data generated it, these benefits turn into hindrances at best, and make the problem largely intractable at worst. Any method that seeks to explore the space of data that share a common persistence diagram must reckon with the possibility that this space is almost certainly infinite, and ostensibly different data may share a common persistence diagram.

Although the ideas of TDA have been applied to a wide variety of data types, in practice, point clouds, images and weighted graphs are among the most studied. We will examine the difficulties of exploring the space of data corresponding to a pre-specified target persistence diagram in each of these, case by case, and where applicable, explicitly define methodology used.

3.1 Point Clouds

Exploring the space of point clouds that approximate a target persistence diagram is perhaps the most difficult case. Here, key features of persistent homology; invariance to rotation and translation, are completely adversarial to sampling based methods if no restriction are made. Most notably, the space of point clouds that approximate a given persistence diagram is not a probability distribution; if we managed to find a point cloud that generated the persistence diagram we seek, then *all* translates, and *all* rotations, great and small, would do so as well. For these reasons, we do not attempt to sample from the space of point cloud data in this work.

3.2 Images

Grayscale images are another data type which have thoroughly been studied through the lens of persistent homology. Often, images are endowed with a variant of the simplicial complex structure defined in Section 2.2, a *cubical complex structure*, where intuitively, triangular based simplices are replaced with those based on squares, which is more reflective of the inherent pixel structure in image data, and more computationally efficient. The broad persistent homology pipeline remains unchanged if we replace "simplicial complex" with "cubical complex", so we shall not delve further into detail here; see [19] for a comprehensive reference.

Working with image data has several advantages over point clouds. Chiefly, the set of images [1] of a fixed size is, although large, clearly finite if we consider pixel intensities to take value in $\{0, 1, \ldots, 255\}$; this immediately implies that the space of all images which generate a pre-specified persistence diagram, even considering all possible translates and rotations, is finite as well.

It is unlikely that attempting to sample over the space of all possible images, pixel by pixel, will be a tractable endeavor; we would like any algorithm that is able to explore the space of persistence diagrams corresponding to a pre-specified persistence diagram to do so in a reasonable amount of time. It is therefore reasonable to seek a compressed representation of this space; discrete cosine transforms (DCT) have a long history in image compression, allowing an image $\mathcal{I}$ to be expressed as a linear combination of 2-dimensional cosine frequencies. Unlike other methods that decompose an image into a linear combination of basis vectors, like principal component analysis or singular value decomposition, these basis elements are not specific to an image or dataset; they are pre-determined (Figure 3.1). By restricting to a select few of these basis elements, particularly those possessing the largest magnitude coefficients in the expression of $\mathcal{I}$ as a linear combination of two-dimensional frequencies, one can compress the image in a manner that preserves perceptual similarity.

3.2.1 DCT Random Walk Metropolis Implementation

Suppose we are given an $n \times n$ grayscale image $\mathcal{I}$ with corresponding target dimension-0 persistence diagram $\mathcal{T}$. We first apply the discrete cosine transformation to $\mathcal{I}$ to get its corresponding representation in terms of the DCT basis vectors; the output of the

[1] Here, we consider grayscale images, which may be represent by an $n \times n$ matrix whose coefficients take values $0, 1, 2, \ldots, 255$; clearly the result still holds if we consider RGB images of a fixed size

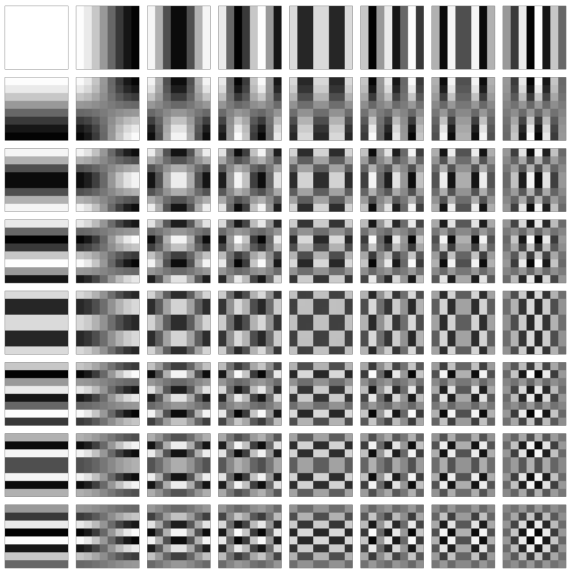


Figure 3.1. A DCT basis for 8×8 images. [9]

DCT is an $n \times n$ matrix $\mathcal{D}$, whose i, jth entry is given by

$$\mathcal{D}_{i,j} = \frac{C(i)C(j)}{\sqrt{2n}} \sum_{k=0}^{n-1} \sum_{l=0}^{n-1} \mathcal{I}_{k,l} \cos\left(\frac{(2k+1)i\pi}{2n}\right) \cos\left(\frac{(2l+1)j\pi}{2n}\right)$$

where $\mathcal{I}_{k,l}$ is the pixel intensity at the (k, l)-th pixel in $\mathcal{I}$ and where

$$C(i) = \begin{cases} \frac{1}{\sqrt{2}} & i = 0 \\ 1 & \text{else} \end{cases}$$

The magnitude of the $\mathcal{D}_{i,j}$ is a measure of that 2-dimensional basis frequency's relative importance to $\mathcal{I}$. This hierarchy provides us a natural way to select $k << n$ of the $n \times n$ possible basis frequencies for the purposes of RWM, with the number of basis frequencies we choose k determines the dimension of the RWM parameter space. If $\{b_1, \ldots b_k\}$ is the set of basis frequencies we have chosen, then we run RWM over the linear combinations

$$\sum_{i=1}^{k} a_i b_i$$

of these vectors as follows:

1. **Initialization:** Let $\alpha_0 = [a_1, \ldots a_k]^T$ be chosen with $\alpha_i \sim \mathcal{N}[0, 1]$. α_0 is the starting point for RWM, whose corresponding image $\mathcal{I}_0$ is given by

$$\mathcal{I}_0 = \sum_{i=1}^{k} a_i b_i$$

Calculate the dimension-0 persistence diagram $\mathcal{P}_0$ for $\mathcal{I}_0$, and set $l = \tilde{p}(\mathcal{P}_0)$, with pseudolikelihood function $\tilde{p}$ given by

$$\tilde{p}(\mathcal{P}_0)) = \exp\left(-\gamma \cdot W_2(\mathcal{P}_0, \mathcal{P})\right)$$

where $\gamma \in \mathbb{R}^+$ is a chosen penalty constant and where $W_2(\mathcal{P}_0, \mathcal{T})$ is 2-Wasserstein distance between the persistence diagram $\mathcal{P}_0$ and a pre-specified target persistence diagram $\mathcal{T}$. Observe that $\tilde{p}(\mathcal{P}) \approx 1$ when $\mathcal{P}$ is close to $\mathcal{T}$ in the 2-Wasserstein distance, and $\tilde{p}(\mathcal{P}) \approx 0$ when $\mathcal{P}$ and $\mathcal{T}$ differ largely. The hyperparameter γ can be tuned to increase or decrease the acceptance rate of the model.

2. **Propose a Move:** Suppose after r iterations our position in parameter space is given by $\alpha_p = [a_{r,1}, \ldots, a_{r,k}]$, with corresponding image

$$\mathcal{I}_r = \sum_{i=1}^{k} a_{r,i} b_i$$

and persistence diagram $\mathcal{P}_r$. Sample $\alpha' = [a'_1, \ldots, a'_k]^T$, where for a chosen step size ϵ,

$$a'_i \sim \mathcal{N}(a_{r,i}, \epsilon)$$

Let $\mathcal{I}'$ be the image corresponding to α', that is

$$\mathcal{I}' = \sum_{i=1}^{k} a'_i b_i$$

and let $\mathcal{P}'$ be its corresponding persistence diagram.

3. **Evaluate the Proposed Move:** Compute the acceptance ratio

$$A = \frac{\tilde{p}(\mathcal{P}')}{\tilde{p}(\mathcal{P}_r)} = \frac{\exp\left(-\gamma \cdot W_2(\mathcal{P}', \mathcal{T})\right)}{\exp\left(-\gamma \cdot W_2(\mathcal{P}_r, \mathcal{T})\right)}$$

4. **Update:** Let $c \sim \mathcal{U}[0, 1]$. If $c < A$, we accept the proposed move to α', and set $\alpha_{r+1} = \alpha'$. Otherwise, we reject the proposed move, and set $\alpha_{r+1} = \alpha_r$.

5. **Iterate:** Repeat steps 2-4 for a predetermined number of iterations.

Algorithm 1 DCT RWM Algorithm

given $\mathcal{I}$ ▷ grayscale image
given $\mathcal{T}$ :: PD($\mathcal{I}$) ▷ target persistence diagram
given $\gamma \in \mathbb{R}^+$ ▷ tuneable penalty hyperparameter

function $\tilde{p}$($\mathcal{P}$ :: PersistenceDiagram)
    return $\exp\left(-\gamma \cdot W_2(\mathcal{P}), \mathcal{T})\right)$
end function

let $\alpha = [a_1, \ldots, a_k]^T \sim \mathcal{N}[0, 1]$
let $\mathcal{I} = \sum_{i=1}^{k} a_i b_i$
let $\mathcal{P} = \mathbf{PD}(\mathcal{I})$ ▷ dim-0 persistence diagram
let chains = $[\alpha]$

for i=1,..., M **do**
    let $\alpha' = [a'_1, \ldots, a'_k]^T \sim \mathcal{N}(\mu = \alpha, \sigma = \epsilon)$ ▷ for step size ϵ
    let $\mathcal{I}' = \sum_{i=1}^{k} a_i v_i$
    let $\mathcal{P}' = \mathbf{PD}(\mathcal{I}')$

    let $\mathcal{A} = \tilde{p}(\mathcal{P}')/\tilde{p}(\mathcal{P})$
    let $c \sim \mathcal{U}(0, 1)$

    if $c < \mathcal{A}$ **then** ▷ proposed move was accepted
        append chains, α'
        $\alpha \leftarrow \alpha'$
        $\mathcal{P} \leftarrow \mathcal{P}'$
    end if
end for

3.3 Weighted Graphs

Throughout the remained of this work, we will consider a weighted graph to be an ordered tuple $G = (V, E, f)$, where V is the set of vertices of G, E is the set of edges, and where $f : V \to \mathbb{R}$ is a function on the vertices of G. To define a sublevel set filtration structure on weighted graphs, we use the following conventions:

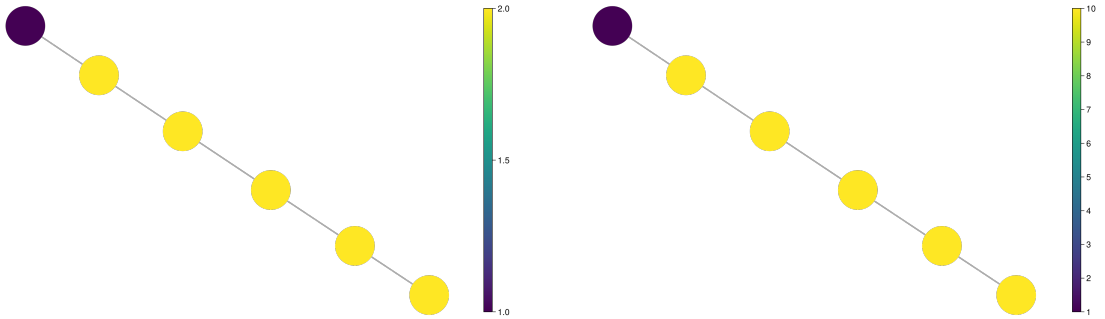
- We consider vertices to be 0-simplices, and edges to be 1-simplices. We do not consider any higher dimensional simplices.
- For $a \in \mathbb{R}$ the sublevel $X_a(f)$ consists of all vertices v which take value $f(v) \leq a$ and edges $e = (i, j)$ satisfying $f(i), f(j) \leq a$.

Weighted graphs face some of the same difficulties as grayscale images; in fact, one may represent a grayscale image via a weighted grid graph, with node values being given by pixel intensities. Just as with images, we might naively try running RWM over the all the nodes of the graph, and in turn, face the same difficulties achieving mixture and convergence when our graph has large numbers of nodes. On the other hand, functions on a graph G have a natural decomposition in terms of graph Laplacian eigenvectors; just as with PCA and DCT, by restricting to linear combinations of the first $k << |V|$ graph Laplacian eigenvectors, [2] we are able to drastically reduce the dimensionality of the RWM parameter space.

All is not solved, however. Persistent homolgy's invariance under translations and rotations makes exploration of the space of point clouds that approximate a given persistence diagram difficult; these data types suffer from an another problem. Namely, values that are beyond the largest finite death value are not not seen by the persistent homology process, and are thus free to increase without bound (see Figure 3.2, for instance). This is yet another reason why running RWM over linear combinations of *all* graph Laplacian eigenvectors is not well-founded; graph Laplacian eigenvectors constitute a basis for all possible functions defined on the vertices of G, including those with arbitrarily large coefficients on vertices ignored by the persistent homology process.

Of course, we are free to impose cutoffs on the linear combinations of graph Laplacian eigenvectors to alleviate this problem somewhat, rejecting all proposed graph functions f during RWM if, for any $v \in V$, $f(v) > \alpha$ for some pre-chosen α. The graph Laplacian eigenvectors themselves have some natural protections against this phenomena. Recall that graph Laplacian eigenvectors corresponding to eigenvalues greater than 0 cannot have non-negative local minima or non-positive local maxima; if

[2] We assume that the graph Laplacian eigenvectors are ordered by increasing eigenvalue.



(a) G_1 (b) G_2

Figure 3.2. Two weighted graphs, G_1 and G_2 with significantly different node functions, but which generate identical persistence diagrams

we wish to scale a node or a collection of nodes of G to be arbitrarily large using linear combinations of a proper subset of the graph Laplacian eigenvectors of G *without changing the underlying persistence diagram*, we must balance the increases that we achieve with decreases that will occur from scaling the various (negative) local minima.

The stability of persistence diagrams with respect to the q-Wasserstein distance presents challenges as well, in that it means we cannot assume that the distributions were are exploring using RWM constitute probability distributions at all under the pseudo-likelihood function $\tilde{p}(\mathcal{P}) = \exp\left(-\gamma W_q(\mathcal{P}, \mathcal{T})\right)$ (Algorithm 1). If there happens to be an infinite subset $\mathcal{A}$ of the full parameter space that perfectly approximates a given persistence diagram, then stability means that all parameters within tubular neighborhood of $\mathcal{A}$ will have high likelihood as well.

Proposition 3.1. *Let $G = (V, E)$ be a graph, with normalized graph Laplacian eigenvectors $v_1, \ldots v_n$. Then the q-Wasserstein distance is stable with respect to persistence diagrams induced by functions defined by linear combinations of graph Laplacian eigenvectors. That is, if*

$$\alpha = [\alpha_1, \ldots, \alpha_n]$$
$$\beta = [\beta_1, \ldots, \beta_n]$$

are sufficiently close in the Euclidean 2-norm, the functions $f_\alpha, f_\beta : V \to \mathbb{R}$ defined by

$$f_\alpha = \sum_{i=1}^{n} \alpha_i v_i$$
$$f_\beta = \sum_{i=1}^{n} \beta_i v_i$$

induce persistence diagrams that are are close in terms of the q-Wasserstein distance.

Proof. Let the number of vertices of G be m, and let V be the $m \times n$ matrix with columns $v_1, \ldots, v_n$. The mapping

$$\alpha = [\alpha_1, \ldots, \alpha_n] \mapsto f_\alpha := \sum_{i=1}^{n} \alpha_i v_i$$

may be viewed as $\alpha \mapsto V\alpha$, which is clearly uniformly continuous.

On the other hand, if G has m vertices, any sublevel set filtration defined on G can have at most m 1-simplices, $\binom{m}{2}$ 2-simplices, and so forth. Thus if $||f - g||_q^q < c$ in the Euclidean q-norm (where we are viewing functions $f, g : V \to \mathbb{R}$ as vectors in $\mathbb{R}^m$), then in L_q norm $||\cdot||_{L_q}$, we have

$$||f - g||_{L_q}^q = \sum_{\sigma \in G} \max_{x \in \sigma} |f(x) - g(x)|^q \leq mc + \binom{m}{2} c + \cdots + \binom{m}{m} c$$

at worst. Thus for any $\epsilon > 0$, we can choose

$$\delta < \frac{\epsilon}{\sum_{i=1}^{m} \binom{m}{i}}$$

so that if $||\alpha - \beta||_q^q < \delta$ for $\alpha, \beta \in \mathbb{R}^n$, we have

$$\begin{aligned} ||f_\alpha - f_\beta||_{L_q} &= \sum_{\sigma \in G} ||f(\sigma) - f(\sigma)||_q \\ &\leq m\delta + \binom{m}{2}\delta + \cdots + \binom{m}{m}\delta < \epsilon \end{aligned}$$

By [23], the q-Wasserstein distance between the persistence diagrams induced by f_α and f_β is bounded above by $||f_\alpha - f_\beta||_{L_q}$, so the claim follows. □

Proposition 3.1 implies the q-Wasserstein distance is stable when we restrict to taking linear combinations of subsets of graph Laplacian eigenvectors $\mathcal{S} = \{v_1, \ldots, v_k\}$ as well. In particular, if $\tilde{\mathcal{P}}$ is a target persistence diagram, and $\alpha = [\alpha_1, \ldots \alpha_k]$ is such that

$$f_\alpha = \sum_{i=1}^{k} \alpha_i v_i$$

induces a persistence diagram $\mathcal{P}_\alpha$ identical to $\tilde{\mathcal{P}}$, then we can choose a neighborhood N about α with the property that for any $\beta = [\beta_1, \ldots, \beta_k] \in N$, the function

$$f_\beta = \sum_{i=1}^{k} \beta_i v_i$$

induces a persistence diagram $\mathcal{P}_\beta$ whose q-Wasserstein distance to $\mathcal{P}_\alpha$, and thus to $\tilde{\mathcal{P}}$ is bounded above. In turn, this means that our specified pseudoliklihood function

$$\tilde{p}(\mathcal{P}) = \exp\left(-W_q(\mathcal{P}, \tilde{\mathcal{P}})\right)$$

will have likelihood bounded *below* on N. In other words, the spaces we are attempting to sample from are not probability distributions if there are infinitely many α_j with $W_q(\mathcal{P}_{\alpha_j}, \tilde{\mathcal{P}}) = 0$.

Taking linear combinations of graph Laplacian eigenvectors is not the only way we can explore the space of weighted graphs that approximate a given persistence diagram, however. And although the naive method of running RWM over all nodes is intractable for the aforementioned reasons, allowing each node to vary independently as we run RWM is a desirable feature if we wish to comprehensively explore the space of weighted graphs that approximate a given persistence diagram. Certainly, we will need to restrict the values nodes can take if we wish to vary them independently of one another. And from the point of view of persistent homology, the only values that matter, and the only values that make one function on the nodes of a graph G distinguishable from another are those values which appear as births values or death values in the corresponding persistence diagrams. To that end, we introduce a result that provides some theoretical justification for this idea.

Theorem 3.2. *Let $f : \mathcal{X} \to \mathbb{R}$ be a tame function on simplicial complex $\mathcal{X}$ inducing a sublevel set filration with persistence diagram $D = \{(b_1, d_1), (b_2, d_2), \ldots, (b_n, d_n)\}$. Denote the collections of all finite birth/death values by b and d, respectively. Then there exists a tame function $\hat{f} : \mathcal{X} \to \mathbb{R}$, taking values only in $b \cup d$, which induces a persistence diagram identical to that of f.*

Proof. Let $f : K \to \mathbb{R}$ define a sublevel set filtration on K, with corresponding persistence diagram P. Collecting all intervals finite birth/death values into sets b, d respectively, define a new function $\hat{f} : K \to \mathbb{R}$ via

$$\hat{f}(x) = \begin{cases} \inf_\gamma \{f(x) \leq \gamma \,|\, \gamma \in b \cup d\}, & f(x) \leq \max\{d\} \\ \max_x f & \text{else} \end{cases}$$

and observe since $f(x) \leq \hat{f}(x)$, there is an induced inclusion on chain complexes $K_i(\hat{f}) \to K_i(f)$ at every threshold i. Moreover, for $\tau \in b \cup d$, $K_\tau(\hat{f}) \cong K_\tau(f)$, for

$$\begin{aligned} x \in f^{-1}((-\infty, \tau]) &\implies f(x) \leq \tau \\ &\implies \inf_\gamma \{f(x) \leq \gamma \,|\, \gamma \in B \cup D\} \leq \tau \\ &\implies x \in \hat{f}^{-1}((-\infty, \tau]) \end{aligned}$$

By the discussion in Section 2.5, it suffices to show that $\text{rank}(p_i^j) = \text{rank}(\hat{p}_i^j)$. Certainly if $i, j \in b \cup d$, the result is trivial, since $K_\tau(f) \cong K_\tau(\hat{f})$ for $\tau \in b \cup d$. Otherwise, by tameness, let

$$\gamma = \sup_{\beta \in b \cup d} \{\beta \leq i\}$$
$$\tau = \sup_{\beta \in b \cup d} \{\beta \leq j\}$$

The inclusions $K_\alpha(\hat{f}) \to K_\alpha(f)$, $\alpha \in \mathbb{R}$ constitute a chain map, and so the following diagram commutes:

$$\begin{array}{ccccccc}
H(K_\gamma(f)) & \xleftarrow[\cong]{p_\gamma^i} & H(K_i(f)) & \xrightarrow{p_i^\tau} & H(K_\tau(f)) & \xleftarrow[\cong]{p_\tau^j} & H(K_j(f)) \\
\uparrow \cong & & \uparrow g & & \uparrow \cong & & \uparrow f \\
H(K_\gamma(\hat{f})) & \xleftarrow[\hat{p}_\gamma^i]{\cong} & H(K_i(\hat{f})) & \xrightarrow[\hat{p}_i^\tau]{} & H(K_\tau(\hat{f})) & \xleftarrow[\hat{p}_\tau^j]{\cong} & H(K_j(\hat{f}))
\end{array}$$

The commutivity of the leftmost square implies that g is an isomorphism, and similarly, the commutivity of the rightmost square implies that f is an isomorphism. Since

$$p_\tau^j \, p_i^\tau \, g = f \, \hat{p}_\tau^j \, \hat{p}_i^\tau$$

with f, g isomorphisms, it follows $\text{rank}(p_\tau^j \, p_i^\tau) = \text{rank}(\hat{p}_\tau^j \, \hat{p}_i^\tau)$, which in turn implies $\text{rank}(p_i^k) = \text{rank}(\hat{p}_i^j)$. □

By Theorem 3.2 we can restrict the possible values each vertex in a weighted graph takes to values that are finite birth/death values in the target persistence diagram. Under this perspective, there are only finitely many different functions $f : V \to \mathbb{R}$ defined on the vertices of a graph G, and, while the number of such functions may be far too large to enumerate exhaustively, we can modify the RWM methods detailed in 2 to explore different arrangements of node values that yield approximations of a target persistence diagram, detailed in 2. Under the discrete perspective, we are not running RWM over the vertices themselves; rather, we fix an ordering of possible values each node may take, say

$$\mathcal{S} = [v_1, v_2, \ldots, v_N]$$

and "move" about the product space

$$\underbrace{\mathcal{S} \times \cdots \times \mathcal{S}}_{|V| \text{ times}}$$

by varying the index $i = 1, \ldots, N$ from which a node draws its value from $\mathcal{S}$.

Algorithm 2 Discrete Birth/Death RWM Algorithm

given $G = (V, E, f_{\mathcal{T}})$ ▷ $|V| = k$
given $\mathcal{T}$ :: PD(G) $= \{(b_1, d_1), \ldots, (b_k, d_k)\}$, $b_i, d_i < \infty$
given $\gamma \in \mathbb{R}^+$ ▷ tuneable penalty hyperparameter

function $\tilde{p}(\ \mathcal{P}$ :: PersistenceDiagram $)$
    return $\exp\left(-\gamma \cdot W_2(\mathcal{P}), \mathcal{T})\right)$
end function

let $\mathcal{S} = \text{sort}([b_1, d_1, \ldots b_k, d_k])$ ▷ (sorted) possible node values
let $M = \text{length}(\mathcal{S})$.

let inds $= [i_1, \ldots i_k]$, $i_s \sim \text{randint}(0 : M)$ ▷ starting indices
let $f = [\mathcal{S}[i_1], \ldots, \mathcal{S}[i_k]]$ ▷ initial node weights
let $G_0 = (V, E, f)$ ▷ initial weighted graph
let $\mathcal{P} = \mathbf{PD}(G_0)$ ▷ initial persistence diagram

let chains $= [G_0]$

for i=1, …, N **do**
    let inds$' = [i'_1, \ldots, i'_k]$, $i'_s = \text{randchoice}(\{i_s - 1, i_s, i_s + 1\})]$ [3]
    let $f' = [\mathcal{S}[i'_1], \ldots, \mathcal{S}[i'_k]]$
    let $G' = (V, E, f'$
    let $\mathcal{P}' = \mathbf{PD}(G')$

    let $\mathcal{A} = \tilde{p}(\mathcal{P}')/\tilde{p}(\mathcal{P})$
    let $c \sim \mathcal{U}(0, 1)$

    if $c < \mathcal{A}$ **then** ▷ proposed move was accepted
        append chains, G'
        inds $\leftarrow$ inds$'$
        $\alpha \leftarrow \alpha'$
        $\mathcal{P} \leftarrow \mathcal{P}'$
    end if
end for

Chapter 4: Results

In this section, we will showcase how our methods perform on a mixture of real-world and synthetic data, presenting instances in which our methods find success, and, where relevant, providing commentary on limitations of our methods. All code for this section was written in Julia, and persistence diagram computations were performed using the Ripserer library [6].

4.1 DCT Random Walk Metropolis Results

All images used throughout this section are from the KTH-TIPS dataset [15], which has seen some use in other TDA works, such as [3]. All experiments in this section were performed in accordance with the following procedure:

1. **Image Preprocessing:** Although the procedure described in Algorithm 1 works with images of any size, we choose to down-sample the 200×200 pixel KTH images to 32×32 pixels to reduce computation time. A target persistence diagram $\mathcal{T}$ is derived from the down-sampled image.

2. **DCT RWM Parameters:** We run DCT RWM (Algorithm 1) for a total of 100,000 iterations; the first 10,000 iterations are not tracked to allow the model time to explore the parameter space before settling into areas of the parameter space with high pseudoliklihood. Proposed moves follow a normal distribution with standard deviation 0.05 and the penalty hyperparameter γ was experimentally chosen to be 40.0.

3. **Experiments:** We repeat DCT RWM runs using the hyperparameters specified in 2 with 4, 8, 16 and 32 DCT basis vectors to demonstrate the impact that the choice of number of DCT basis elements to use has on the ability of the DCT RWM procedure to find images which have underlying persistence diagram close to $\mathcal{T}$.

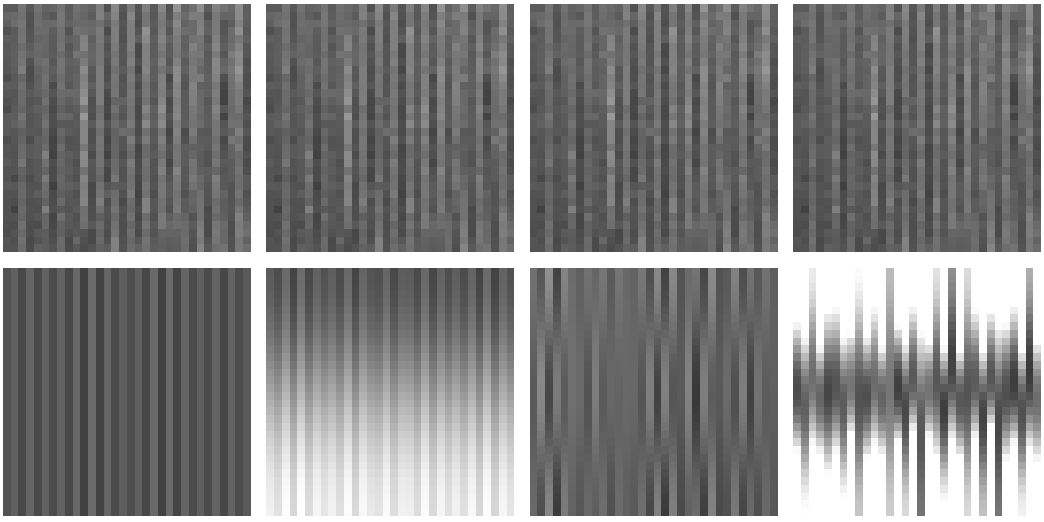
The effectiveness of the proposed methods varies greatly depending on the type of image from which the underlying target persistence diagram is generated. In Figure

4.1, for example, DCT is able to provide a good reconstruction of the given image, even when the number of DCT basis vectors we choose is heavily restricted. In such cases, the DCT RWM method has a tendency to favor linear combinations of these basis vectors which roughly reproduce the given image. The rightmost image in Figure 4.1a is indicative of a larger trend with the DCT RWM method; there is a tendency for the sampling process to favor linear combinations with large positive values for the coefficient of the first DCT basis vector (the constant vector), and attempt to have the other vectors compensate. This is made more evident by the selection of trace plots from a DCT RWM run shown in Figure 4.4.

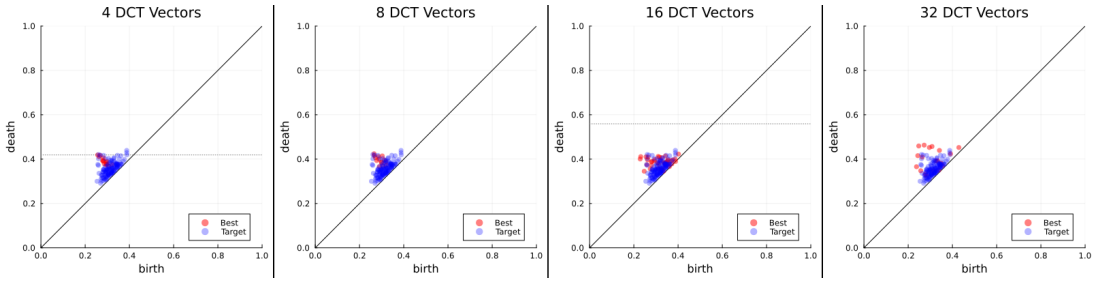
As can be see in Figures 4.1, 4.2 and 4.3, the number of DCT basis vectors chosen has a drastic effect on the images found by DCT RWM, and in turn, the quality of persistence diagrams found by the method. Choosing too few basis elements results in simplistic images, and therefore, persistence diagrams that do not fully reflect the topology captured within the target persistence diagram. On the other hand, there appears to be diminishing returns found in increasing the number of DCT basis vectors over which to run RWM; choosing too many basis vectors not only inhibits the mixing potential of the DCT RWM model, it also can lead to spurious artifacts being found in even the best persistence diagrams found by the method (see Figure 4.2b).

Regardless of the number of basis vectors chosen, however, the DCT RWM method struggles in general to mix, that is, to settle into concentrated, discernible regions in parameter space, despite finding strong candidate images whose persistence diagrams closely approximate the given target; this is indicative that our model is better suited for finding individual images whose persistence diagram approximates a given target, rather than sampling from some broader distribution. Most curiously, the parameter corresponding to the coefficient of the constant DCT basis vector, which is almost always the most prominent DCT basis vector in the DCT decomposition of a given real-world image, has a tendency to increase dramatically over time (for instance, see Figure 4.4). It is therefore likely that the addition of a prior distribution that restricts the ability of parameters to charge off would result in better mixing of the model. The reason for such behavior is not yet understood, and perhaps warrants further investigation in future works.

One might intuitively expect that using a small number of DCT basis vectors ought to result in DCT RWM finding images of relatively low topological complexity; indeed, this is reflected in Figures 4.2a and 4.3a. As we increase the number of DCT basis vectors, we that DCT RWM is able to find increasingly more complicated images whose peristence diagrams more closely align with the given target. There appears to be diminishing returns to increasing the number of DCT basis vectors used in DCT



(a) Images found by the DCT RWM method: (top) - corduroy texture image from which the target persistence diagram was generated (bottom, left to right) - images corresponding to the best persistence diagram found by DCT RWM with use of 4, 8, 16 and 32 DCT vectors, respectively.



(b) Persistence diagrams corresponding to the images in Figure 4.1a

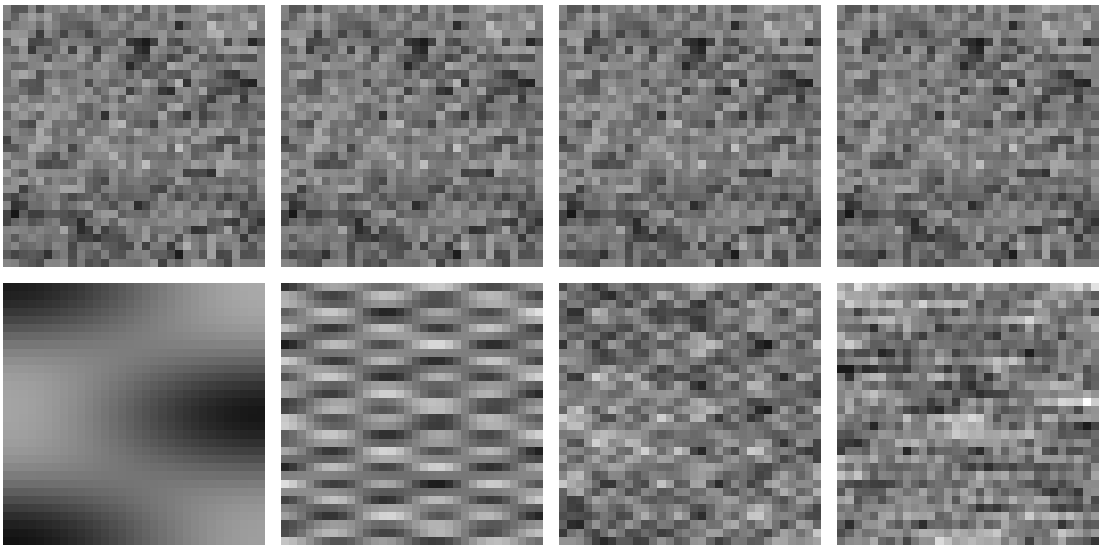
Figure 4.1

RWM; in Figure 4.1a, Figure 4.2a and Figure 4.3a, runs utilizing 32 DCT basis vectors found images with extraneous persistence diagram points even in the best case.

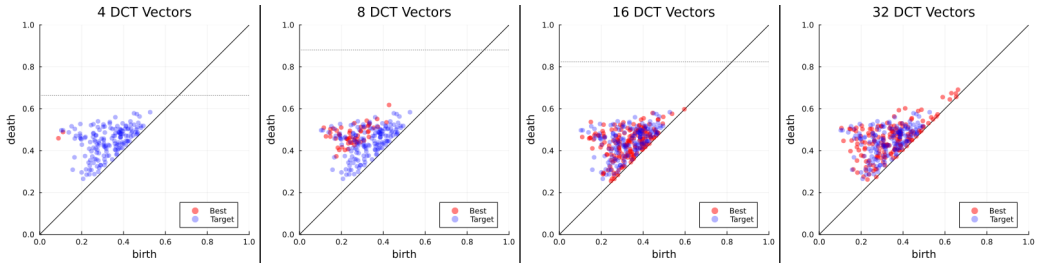
4.2 Graph Laplacian Random Walk Metropolis

In short, the graph Laplacian Random Walk Metropolis (GL RWM) method shares many of the same positive and negative qualities seen in DCT RWM, albeit with better mixing in toy examples. The methods and parameters used to generate the results of this section are detailed as follows:

1. **Generation of Target Data:** The methods of GL RWM are amendable to

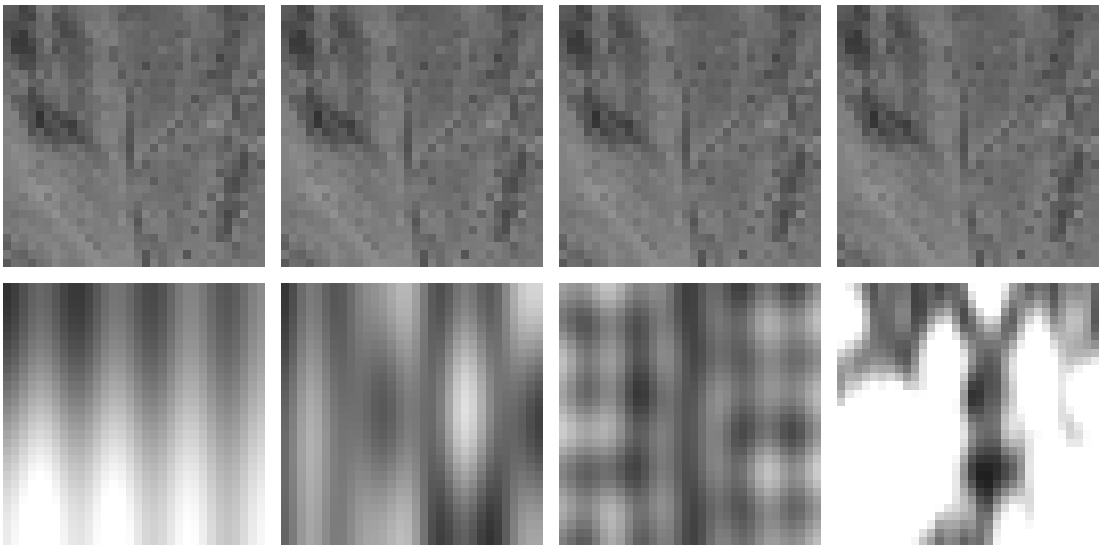


(a) Images found by the DCT RWM method: (top) - brown bread texture image from which the target persistence diagram was generated (bottom, left to right) - images corresponding to the best persistence diagram found by DCT RWM with use of 4, 8, 16 and 32 DCT vectors, respectively.

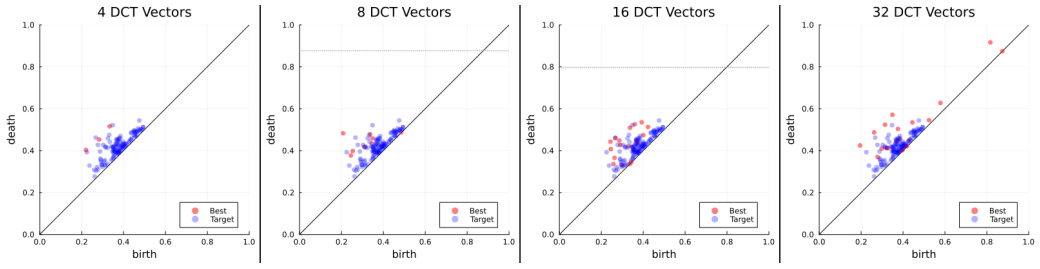


(b) Persistence diagrams corresponding to the images in Figure 4.2a

Figure 4.2



(a) Images found by the DCT RWM method: (top) - lettuce leaf texture image from which the target persistence diagram was generated (bottom, left to right) - images corresponding to the best persistence diagram found by DCT RWM with use of 4, 8, 16 and 32 DCT vectors, respectively.



(b) Persistence diagrams corresponding to the images in Figure 4.3a

Figure 4.3

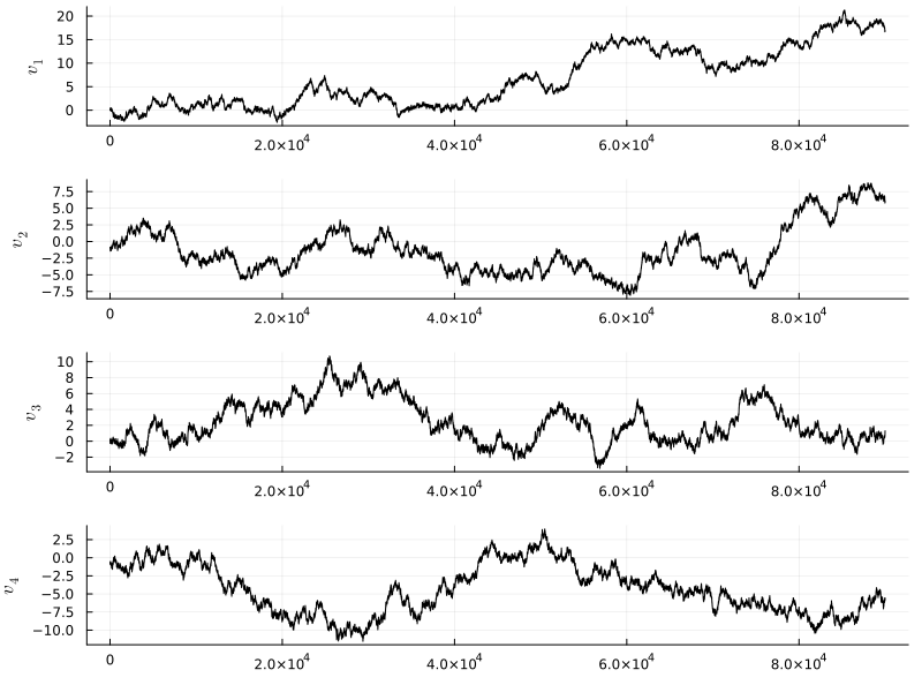


Figure 4.4. Trace plots for coefficients of the first 4 DCT basis elements, where sampling is attempting to approximate the persistence diagram corresponding to the given image in Figure 4.2 with 4 DCT basis vectors. In particular, note that coefficient of the first eigenvector grows larger and larger over time, which in turn inhibits the overall mixing of the model.

any graph structure, but we showcase two particular graph types in this work; Erdős-Renyi graphs [10], where we pre-specify a number of nodes n and an edge probability p, and binary trees with a pre-specified node valence. We then choose a vector f whose length is given by the number of nodes in the graph G, with each entry sampled from the standard uniform distribution $\mathcal{U}[0,1]$. f defines a function on the vertices of G which induces a target persistence diagram $\mathcal{T}$.

2. **GL RWM Parameters:** We run GL RWM for a total of 200,000 iterations; the first 10,000 iterations are not tracked to allow the model time to explore the parameter space before settling into areas of the parameter space with high pseudoliklihood. Proposed moves follow a normal distribution with standard deviation 0.01 and the penalty hyperparameter γ was experimentally chosen to be 10.0.

3. **Experiments:** We preform GL RWM with the hyperparameters specified in 2. In each of the examples presented, we use 8 graph Laplacian eigenvectors.

Figures 4.5 and 4.7 provide examples of how GL RWM preforms on simple graphs. In such instances, one should expect GL RWM to find multiple variations of weighted graphs structures, each generating the same target persistence diagram. While the search capabilities of GL RWM in these cases are by no means comprehensive, it is able to find distinct graphs, each with a persistence diagram approximating a given target, albeit with some extraneous artifacts in the case of 4.5.

The situation changes drastically when we seek to work with more complicated data; in particular, GL RWM systematically produces persistence diagrams that underestimate the number of persistence diagram points in the target persistence diagram. Given the relative smoothness of graph Laplacian eigenvectors, one might expect that the choice of the number of graph Laplacian eigenvectors to use in the GL RWM process should be based on the number of persistence points in the target persistence diagram. This approach has two main flaws. Most important, increasing the number of graph Laplacian eigenvectors often results in poor mixing. Experimentally, increasing the number of graph Laplacian eigenvectors not only suffers from diminishing returns, if taken too far, it actively harms the ability of the model to find reasonable graph structures at all.

The second problem with this approach is that, although the graph Laplacian eigenvectors are relatively smooth, there are no results that bound the number of persistence diagram points each eigenvector can contribute.

Remark. The fact that we cannot bound the number of persistence diagram points of each graph Laplacian eigenvector ties into a conjecture of Courant and Herrman

[13], which, in the case of discrete graphs, posits that linear combinations of the first k graph Laplacian eigenvectors of a graph G can have at most k nodal domains. This is tangentially related to the problem of choosing an optimal number of graph Laplacian eigenvectors from which to run GL RWM; that is, we might hope if the Courant-Herrman conjecture is true, it would provide a theoretical lower bound for the number of graph Laplacian Eigenvectors necessary to replicate a persistence diagram with a given number of points. Unfortunately, the conjecture is not true in general, a 5-pointed star graph suffices as a counter-example, and, to the best of our knowledge, there has been little progress made in finding restricted cases of discrete graphs in which it holds true.

4.3 Discrete Birth/Death Random Walk Metropolis

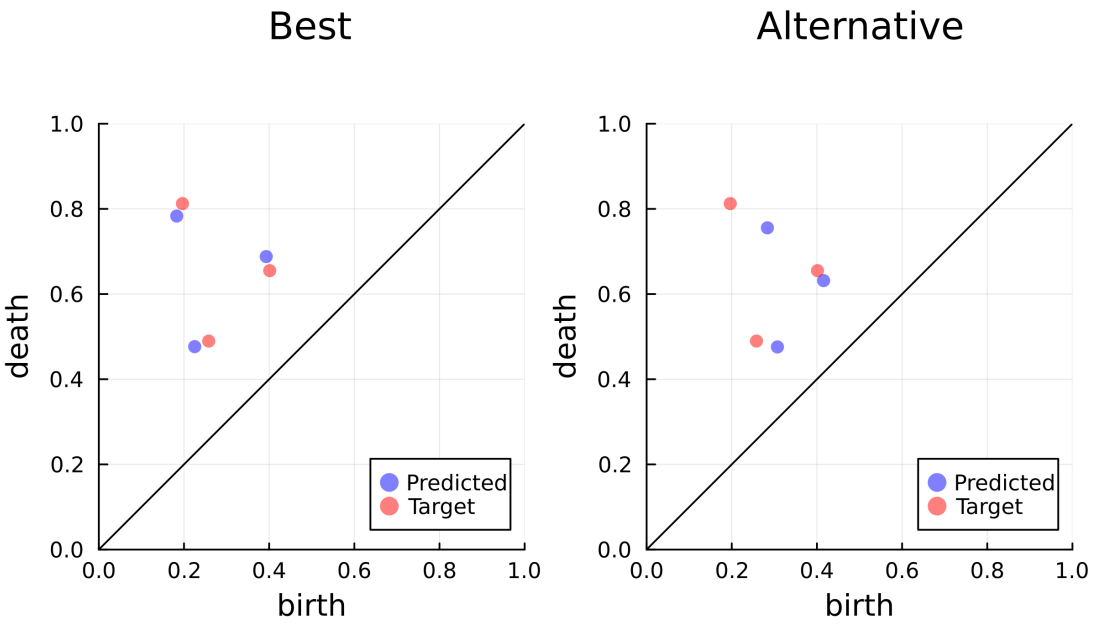
We provide implementation details for all Discrete Birth/Death Random Walk Metropolis (DRWM) experiments:

1. **Generation of Target Data:** In general, the procedure for generating graphs and target persistence diagrams follows the GL RWM setup exactly; DRWM can be used on arbitrary weighted graphs, but for the purposes of comparison with GL RWM, we demonstrate its exploration capabilities on the same Erdős-Renyi graph G and target persistence diagram $\mathcal{T}$ as was seen in Figure 4.7a. We extract all finite birth/death values from $\mathcal{T}$, and sort them in increasing order to specify the possible values the vertices of G can take.

2. **DRWM Parameters:** We run DRWM for a total of 200,000 iterations for the experiment in this section; the first 10,000 iterations are not tracked to allow the model time to explore the parameter space before settling into areas of the parameter space with high pseudoliklihood. Proposed moves follow a normal distribution with standard deviation 0.01 and the penalty hyperparameter γ was experimentally chosen to be 10.0.

Moving from the continuous sampling methods discussed previous to a more discrete approach comes with benefits and drawbacks alike. On the one hand, restricting nodes to values only found in the target persistence diagram allows for enhanced exploration capabilities of the model, seen in the large variety of graphs found by the model which share a common persistence diagram in 4.9. In many instances, the different weighted graphs found by the discrete birth/death RWM method can be slightly tweaked themselves without changing the underlying persistence diagram such instances may be useful from the perspective of data augmentation, but we are more interested in discovering largely distinct weighted graph structures than variants of a single type.

(a) (Left to right) - target graph, the graph whose persistence diagram best approximates that of the target, and an alternative weighted graph configuration found by the model whose persistence diagram also approximates that of the target. The presence of these different graphs is reflected in the density plots in Figure 4.6



(b) Persistence diagrams corresponding to the middle and right graphs in Figure 4.5a.

Figure 4.5

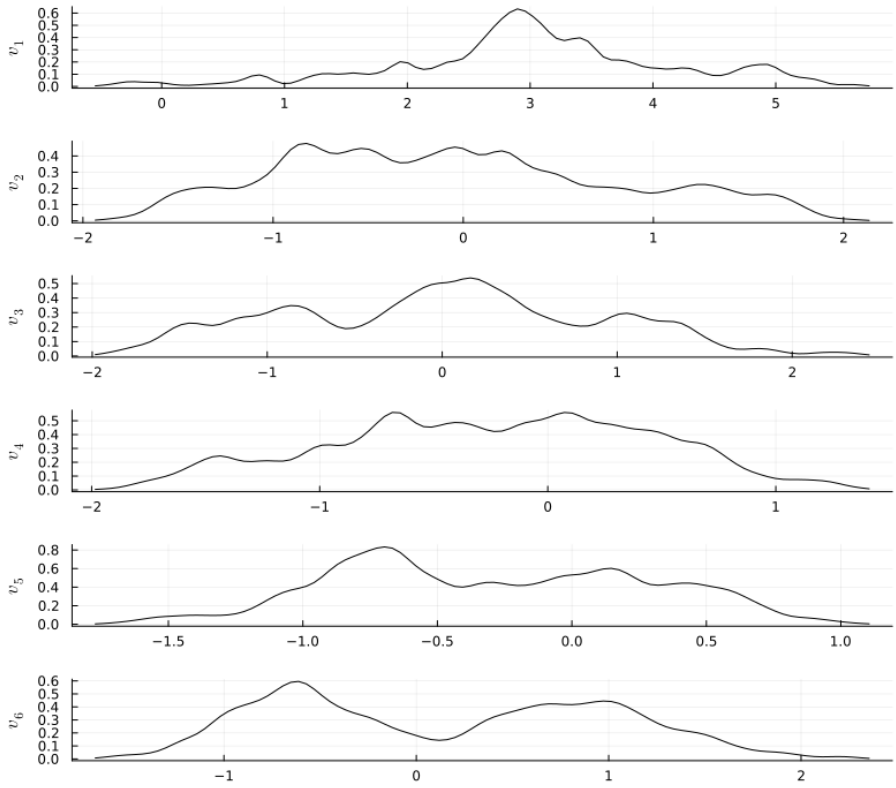
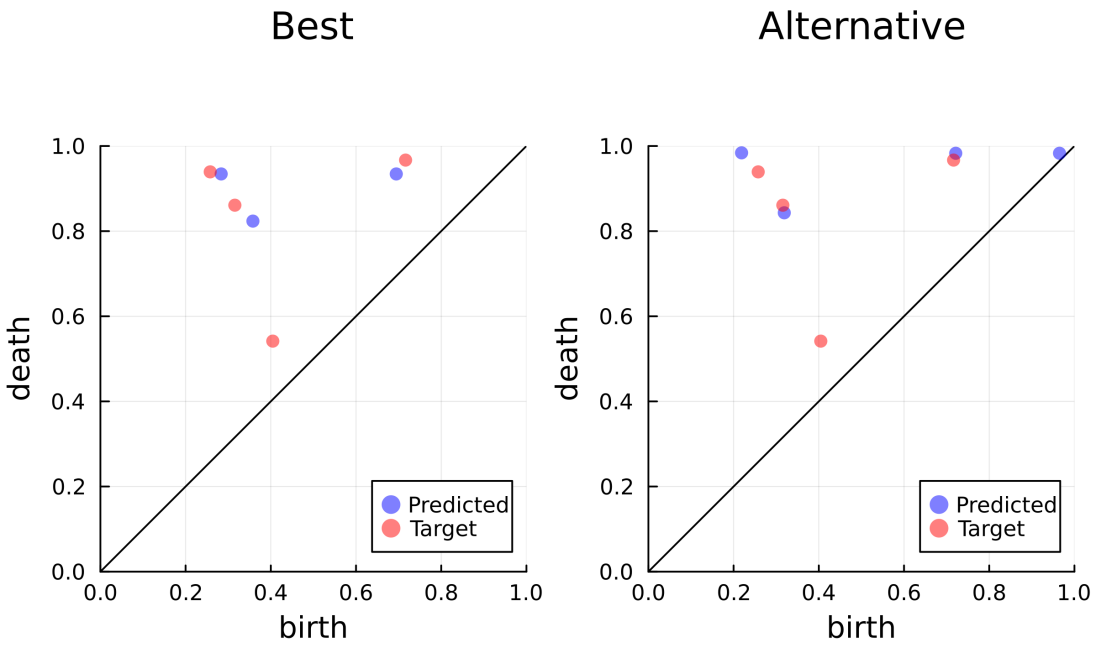


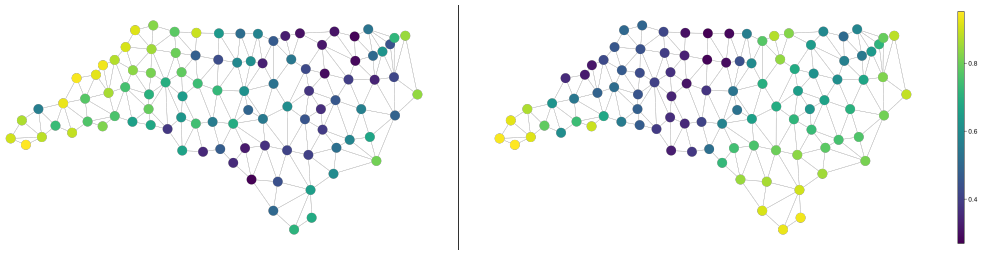
Figure 4.6. Density plot for the GL RWM run in Figure 4.5utilizing 8 graph Laplacian eigenvectors.

(a) (Left to right) - target graph, the graph whose persistence diagram best approximates that of the target, and an alternative weighted graph configuration found by the model whose persistence diagram also approximates that of the target.

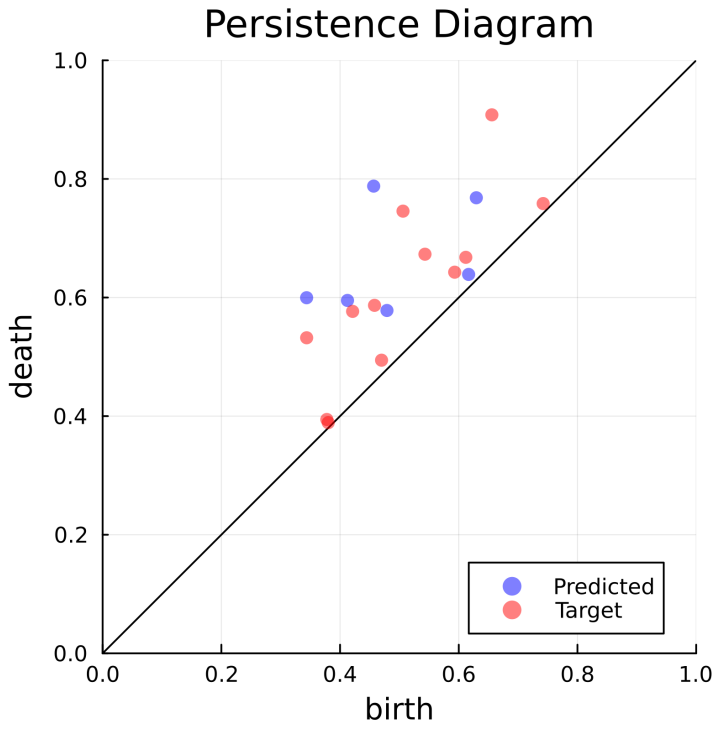


(b) Persistence diagrams corresponding to the middle and right graphs in Figure 4.7a

Figure 4.7



(a) (Left to right) - target graph, the graph whose persistence diagram best approximates that of the target. Note that even under such optimal conditions, GL RWM is not able to generate a graph whose corresponding persistence diagram has a similar amount of persistence diagram points as the target.



(b) Comparison of the target and best persistence diagrams corresponding to the graphs in 4.8a

Figure 4.8

On the other hand, this exploration capability comes at the expense of mixing, which is virtually non-existent 4.10. We believe that the chief culprit of this phenomena is the dimensionality of the search space, and the admittedly ad hoc method by which we allow the node values to change. Although we are able to restrict each node to take only a finite range of values, even if future work devises methods that allow mixing on small scale graphs, it is unlikely that this method will be able to scale to large-scale graph-based data science problems; for this reason, we do not attempt to apply this method on images, or the graph in 4.8a.

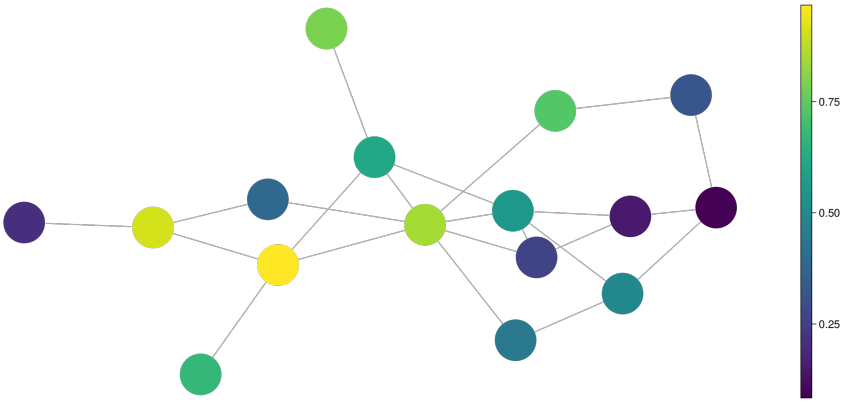
4.4 Closing Remarks

In this work, we presented methods for exploring the spaces of grayscale image and weighted graph data whose underlying persistence diagrams approximate a given target. We observed that all methods were capable of finding distinct examples of data whose persistence diagrams closely approximate a given target, and, simultaneously, all methods suffered from relatively poor mixing.

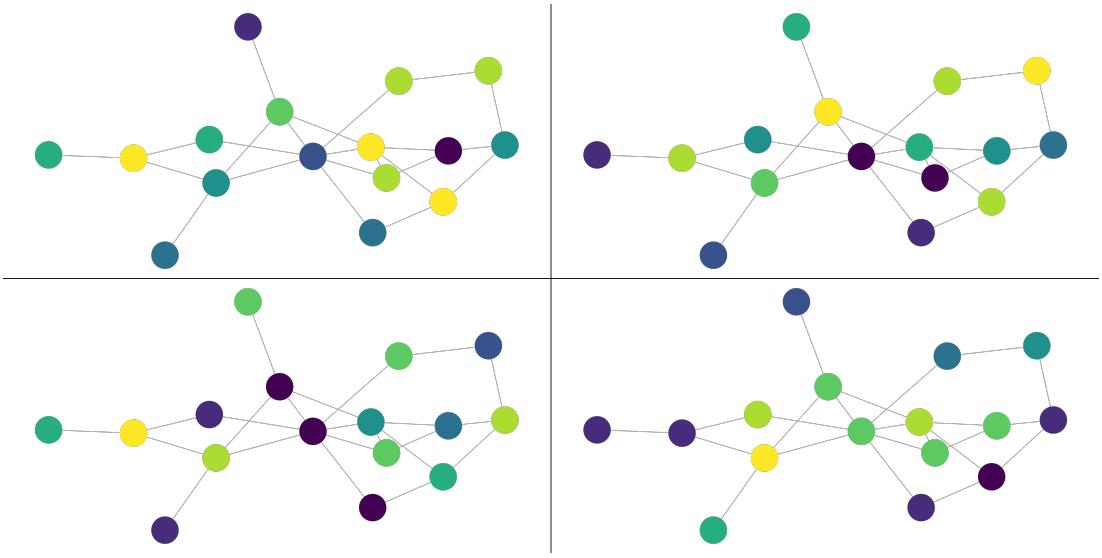
We see two avenues for future works to improve upon the methods presented. Certainly, a more sophisticated pseudo-likelihood function would likely improve mixing, as undoubtedly would the imposition of further restrictions on what values a vertex or pixel may take, perhaps in accordance with some prior distribution.

With regard to the DCT RWM and GL RWM methods, perhaps the biggest improvements may be found in choosing better basis for the respective data types. Both the DCT basis and graph Laplacian eigenvectors enjoy a variety of nice properties, including having an obvious relationship with the data they are used to represent, but there may exists other bases which can be fashioned to be more topologically descriptive or relevant. To that end, many of the topological properties that one may expect linear combinations of graph Laplacian eigenvectors to have, such as bounds for the number of local minima exhibited by such linear combinations, need not hold in general. Such bounds would place the problem of how many basis vectors to use upon firmer theoretical ground.

The inherent high-dimensionality of the search space in the proposed discrete birth/death RWM method limits severely limits its practicality, as stated. At the same time our approach was quite simplistic; it is possible that the theoretical guarantees presented in 3.2 can be adapted to a more sophisticated form, or perhaps used as a preliminary exploratory phase in a larger ensemble of statistical models.



(a) A target graph, the same that was seen in 4.7a



(b) Each of these graphs found by the discrete RWM method has a persistence diagram that matches that in 4.9a.

Figure 4.9

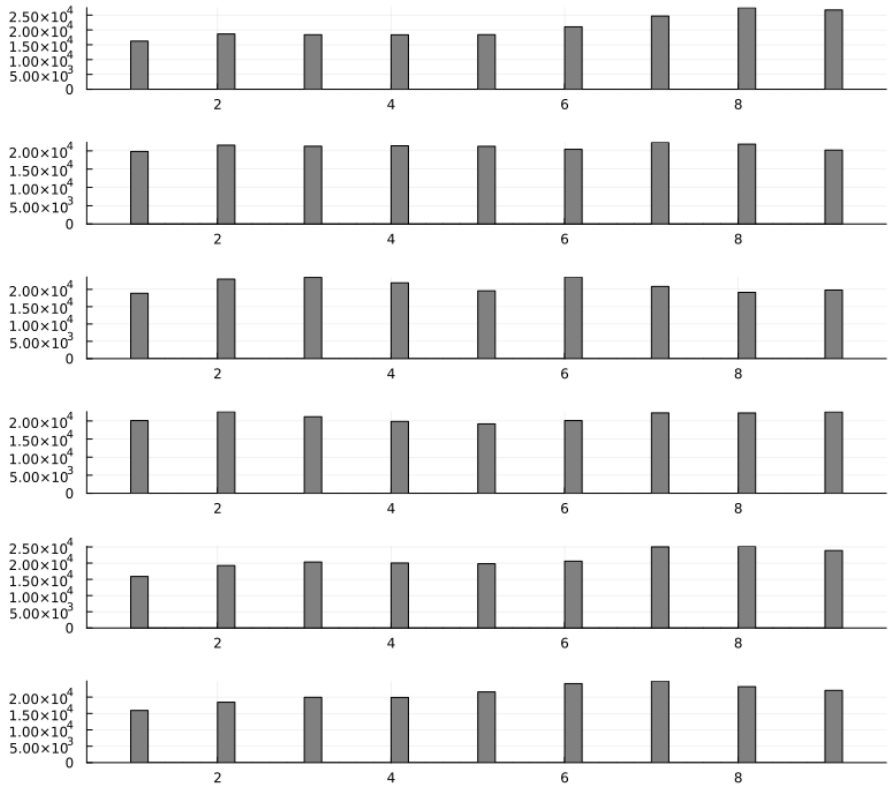


Figure 4.10. A plot of the number of visits each node has to each different value in a given target persistence diagram. Note that the distribution of visit is nearly uniform, indicating that the model is completely failing to settle into a stationary distribution over the course of a run.

Bibliography

[1] Robert J. Adler, Sarit Agami, and Pratyush Pranav. "Modeling and replicating statistical topology and evidence for CMB nonhomogeneity". In: *Proceedings of the National Academy of Sciences* 114.45 (Oct. 2017), pp. 11878–11883. ISSN: 1091-6490. DOI: 10.1073/pnas.1706885114. URL: http://dx.doi.org/10.1073/pnas.1706885114.

[2] Chao Chen et al. "TopoReg: A Topological Regularizer for Classifiers". In: *CoRR* abs/1806.10714 (2018). arXiv: 1806.10714. URL: http://arxiv.org/abs/1806.10714.

[3] Yu-Min Chung and Austin Lawson. "Persistence Curves: A canonical framework for summarizing persistence diagrams". In: *CoRR* abs/1904.07768 (2019). arXiv: 1904.07768. URL: http://arxiv.org/abs/1904.07768.

[4] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. "Stability of persistence diagrams". In: *Proceedings of the Twenty-First Annual Symposium on Computational Geometry*. SCG '05. Pisa, Italy: Association for Computing Machinery, 2005, pp. 263–271. ISBN: 1581139918. DOI: 10.1145/1064092.1064133. URL: https://doi.org/10.1145/1064092.1064133.

[5] William Crawley-Boevey. "Decomposition of pointwise finite-dimensional persistence modules". In: *Journal of Algebra and its Applications* 14.05 (2015), p. 1550066.

[6] Matija Čufar. "Ripserer.jl: flexible and efficient persistent homology computation in Julia". In: *Journal of Open Source Software* 5.54 (2020), p. 2614. DOI: 10.21105/joss.02614. URL: https://doi.org/10.21105/joss.02614.

[7] Justin Curry. *The Fiber of the Persistence Map for Functions on the Interval*. 2019. arXiv: 1706.06059 [math.AT].

[8] Daryl Deford. *Dual Graphs for 2010 Census Units*. URL: https://people.csail.mit.edu/ddeford/dual_graphs.html.

[9] Devcore. *8x8 DCT (discrete cosine transformation)*. Wikipedia. 2012. URL: https://commons.wikimedia.org/wiki/File:DCT-8x8.png.

[10] P ERDdS and A R&wi. "On random graphs I". In: *Publ. math. debrecen* 6.290-297 (1959), p. 18.

[11] Marcio Gameiro, Yasuaki Hiraoka, and Ippei Obayashi. "Continuation of point clouds via persistence diagrams". In: *Physica D: Nonlinear Phenomena* 334 (Nov. 2016), pp. 118–132. ISSN: 0167-2789. DOI: 10.1016/j.physd.2015.11.011. URL: http://dx.doi.org/10.1016/j.physd.2015.11.011.

[12] Robert Ghrist. "Barcodes: The persistent topology of data". In: *BULLETIN (New Series) OF THE AMERICAN MATHEMATICAL SOCIETY* 45 (Feb. 2008). DOI: 10.1090/S0273-0979-07-01191-3.

[13] Graham ML Gladwell and H Zhu. "Courant's nodal line theorem and its discrete counterparts". In: *Quarterly Journal of Mechanics and Applied Mathematics* 55.1 (2002), pp. 1–15.

[14] Jonathan Kelner. *An Algorithmist's Toolkit: Lecture 3*. Sept. 2009.

[15] P Mallikarjuna et al. "THE KTH-TIPS2 database". In: (July 2006).

[16] Vasileios Maroulas, Farzana Nasrin, and Christopher Oballe. *A Bayesian Framework for Persistent Homology*. 2019. arXiv: 1901.02034 [stat.ME].

[17] Michael Moor et al. "Topological Autoencoders". In: *CoRR* abs/1906.00722 (2019). arXiv: 1906.00722. URL: http://arxiv.org/abs/1906.00722.

[18] Arnur Nigmetov and Dmitriy Morozov. *Topological Optimization with Big Steps*. 2023. arXiv: 2203.16748 [cs.CG].

[19] Evan Oman. "An Introduction to Computational Cubical Homology". PhD thesis. May 2013. DOI: 10.13140/RG.2.1.4417.7761.

[20] Theodore Papamarkou et al. "A Random Persistence Diagram Generator". In: *arXiv e-prints*, arXiv:2104.07737 (Apr. 2021), arXiv:2104.07737. DOI: 10.48550/arXiv.2104.07737. arXiv: 2104.07737 [stat.ML].

[21] Leonid Polterovich et al. *Topological Persistence in Geometry and Analysis*. 2021. arXiv: 1904.04044 [math.AT].

[22] David I. Shuman et al. "Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Data Domains". In: *CoRR* abs/1211.0053 (2012). arXiv: 1211.0053. URL: http://arxiv.org/abs/1211.0053.

[23] Primoz Skraba and Katharine Turner. *Wasserstein Stability for Persistence Diagrams*. 2023. arXiv: 2006.16824 [math.AT].

[24] Afra Zomorodian and Gunnar Carlsson. "Computing persistent homology". In: *Proceedings of the twentieth annual symposium on Computational geometry*. 2004, pp. 347–356.