2019-12-04

Coordinated Persistent Homology and an Application to Seismology

Nickolas Brenten Callor
Brigham Young University

Coordinated Persistent Homology and an Application to Seismology

Nickolas Brenten Callor

A dissertation submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Greg Conner, Chair
Blake H Barker
Curtis Andrew Kent
Nathan C Priddis
Jared P Whitehead

Department of Mathematics

Brigham Young University

ABSTRACT

Coordinated Persistent Homology and an Application to Seismology

Nickolas Brenten Callor
Department of Mathematics, BYU
Doctor of Philosophy

The theory of persistent homology (PH), introduced by Edelsbrunner, Letscher, and Zomorodian in [1], provides a framework for extracting topological information from experimental data. This framework was then expanded by Carlsson and Zomorodian in [2] to allow for multiple parameters of analysis with the theory of multidimensional persistent homology (MPH). This particular generalization is considerably more difficult to compute and to apply than its predecessor. We introduce an intermediate theory, coordinated persistent homology (CPH), that allows for multiple parameters while still preserving the clarity and coherence of PH.

In addition to introducing the basic theory, we provide a polynomial time algorithm to compute CPH for time series and prove several important theorems about the nature of CPH. We also describe an application of the theory to a problem in seismology.

Keywords: persistent homology, multidimensional persistent homology, coordinated persistent homology, multifiltrations, time series

Acknowledgments

Completing this thesis would have been impossible without the help and support of so many in my life. First and foremost, I am grateful to my God for my understanding and any talent I possess, as well as the constant inspiration and sustaining influence of His Spirit.

I am particularly grateful to my advisor, Greg Conner, and the many professors from whom I have learned so much. I am especially indebted to Dr. Conner for the countless hours he has spent with me in counsel. I also acknowledge Lawrence Fearnley, David Wright and Denise Daniels for their roles in introducing me to the beauty of mathematics.

I am also grateful for the opportunity provided by Sandia National Laboratories to apply my work to a meaningful problem. It was inspiring to see my algorithm work on such interesting data. The process of writing actual code to implement the algorithm was also very informative and influenced the presentation of the algorithm provided herein. The Geoscience Research Foundation of Sandia National Laboratories' Laboratory Directed Research and Development program funded the specific application of my work to the problem described in Chapter 6. I also wish to thank Jason Heath, Erik Webb, and Steven Vigil for setting up and coordinating my work with the lab; Jason Heath, Brian Young, Katherine Aur, and Christian Poppeliers for their expertise and contributions to that project; and Christopher Young and Neill Symons for sharing their perspectives on additional applications of my work to seismology.

My family, friends and students were also indispensable. I will forever be indebted to my parents for the time, love and money they put into raising me and supporting me in all my ambitions. I am especially grateful for their prayers on my behalf. My siblings and extended family were not a whit behind them in their support of me as well.

I truly cannot express the love and gratitude I feel for these individuals and the many others who have not been named. Thank you.

Contents

List of Figures

Chapter 1. Introduction

1.1 Introduction

The theory of *persistent homology* (PH), developed by Robins [3], Edelsbrunner [1], and Carlsson [4], is a relatively new tool for applying topological ideas to data analysis. This theory utilizes the algebraic structure of homology to describe how the topological properties of a dataset change relative to some parameter. Of particular importance in this theory is the idea of stability: a small change in the original dataset will only produce a small change in the result. This stability allows us to determine what topological features are likely to be real features of the dataset, by considering how much the original dataset would need to change in order to remove such a feature.

More recently, Carlsson and Zomorodian, in [2] and [5], introduced a generalization of PH to involve multiple parameters simultaneously, the theory of *multidimensional persistent homology* (MPH). They also showed that any generalization of persistent homology to multiple parameters must compromise on at least one of the following properties: invariance, completeness, or discreteness. The rank invariant was then put forth as a good compromise, being a discrete invariant that agrees with PH in one dimension. Cerri et al, in [6], construct a metric on the space of rank invariants, so that this distance is also stable.

Our primary result is a new generalization of PH, which we call *coordinated persistent homology* (CPH). The compromise we make is to consider invariants on a different class of objects than MPH, which allows us to create a complete, discrete invariant that agrees precisely with PH in the single dimensional case, but is easier to compute and analyze than MPH in higher dimensions. We provide a polynomial time algorithm for computing CPH, show that the output is stable, and demonstrate an application of this theory to a classification problem in seismology [7]. The results of this application suggest several additional interesting characteristics of the theory and these are presented as conjectures at the end.

1.2 Outline

In Chapter 2, we will give an overview of our motivation and approach as well as some relevant basic facts that will be of use to us.

In Chapter 3, we will review the basic concepts of persistent homology and multidimensional persistent homology. The concepts will be presented in close parallel to our development of coordinated persistent homology, in order to highlight the similarities and differences between the three theories.

In Chapter 4, we will define a labelled linear graph and show how to construct a certain total order, which are the objects of study under coordinated persistent homology. We will prove that these constructions are stable in a certain sense, that is to say, small changes in the input will produce small changes in the resultant objects.

In Chapter 5, we will provide the algorithm for computing coordinated persistent homology. We will prove that the algorithm is well-defined and agrees with persistent homology when applied to single dimensional data. We close this chapter with a stability result for the output of the algorithm.

In Chapter 6, we discuss an application of coordinated persistent homology to a classification problem in seismology. The full project is detailed in [7], but we will focus on the mathematics involved, with special emphasis on the interpretation of the results. We close this chapter by formalizing certain conjectures about coordinated persistent homology that follow from several observations from this study.

Chapter 2. Background Information

We assume the reader has a basic knowledge of the real numbers, denoted $\mathbb{R}$, and p-dimensional real space, $\mathbb{R}^p$. We will first introduce multifiltrations, which are the basic object of study in persistent homology, multidimensional persistent homology, and coordinated persistent homology, and then we will discuss the desired output of these theories. We will close this chapter with two sections on order theory that are necessary to describe the coordinated persistent homology algorithm.

2.1 Motivation

Consider any experiment in which we make a set of p measurements at fixed times. Assuming these measurements are each numeric, we can think of this as taking samples of a curve immersed in p-dimensional real space, $\mathbb{R}^p$. Let T be the set of times at which measurements are made and choose an order for the measurements. Then the results of our experiment are encoded as a function $f : T \longrightarrow \mathbb{R}^p$, such that the projection to the ith coordinate gives the ith measurement taken at time t.

Given such a function, we naturally obtain a *multifiltration*, as in [2], by using each dimension as a parameter. More precisely, a multifiltration is a collection of objects $\{A_u | u \in \mathcal{U} \subset \mathbb{R}^p\}$ together with maps $\phi_u^v : A_u \to A_v$ for pairs $u, v \in \mathbb{R}^p$ where the coordinates of u are less than or equal to the coordinates of v, such that ϕ_u^u is the identity map on A_u and the following diagram commutes whenever the maps $\phi_u^v, \phi_u^w, \phi_v^x, \phi_w^x$ are defined:

$$\begin{array}{ccc} A_v & \xrightarrow{\phi_v^x} & A_x \\ {\scriptstyle \phi_u^v}\uparrow & & \uparrow{\scriptstyle \phi_w^x} \\ A_u & \xrightarrow{\phi_u^w} & A_w \end{array}$$

These conditions imply that whenever ϕ_w^x and ϕ_u^w are defined, we have $\phi_u^x = \phi_w^x \circ \phi_u^w$. Given any topological space X and a function $f : X \to \mathbb{R}^p$, we may obtain a multifiltration from $X_u = \{x \in X | f(x) \text{ has coordinates less than or equal to } u\}$ with inclusion maps for ϕ_u^v.

Example 2.1.1. *Suppose we have recordings of birds vocalizing, with 2 audio channels for stereo sound. Then we can treat each audio channel as a separate measurement, giving us a function* $f : T \longrightarrow \mathbb{R}^2$.

This gives us a multifiltration based on the amplitude of each channel,

$$X_{(\alpha,\beta)} = \{t \in T | f(t) = (a, b), a < \alpha, b < \beta\}.$$

In our example, $f(t) = (a, b)$ *means that the first audio channel has amplitude* a *and the second channel has amplitude* b. *Then* $X_{(\alpha,\beta)}$ *consists of all times when each channel is below the associated threshold and we have inclusion maps from* $X_{(\alpha,\beta)}$ *to* $X_{(\gamma,\delta)}$ *whenever* $\alpha \leq \gamma$ *and* $\beta \leq \delta$. *Studying this set of times is analogous to studying the frequencies present in the recording.*

Example 2.1.2. *Suppose we record the daily open, high, low, and close values for a stock over the course of a month. Then we can obtain a function* $f : T \longrightarrow \mathbb{R}^4$ *by choosing an order for those 4 values. For instance, if we order the values as listed above,* $f(t) = (a, b, c, d)$ *means that on day* t *of our recording, this stock opened at* a, *had a high value of* b, *a low value of* c, *and closed at* d. *Of course, there are* $4! = 24$ *ways to choose the order of our coordinates, but the multifiltrations we obtain correspond naturally to each other. A more interesting variation comes from changing the order we use for each coordinate. For instance, we could consider*

$$X_{(\alpha,\beta,\gamma,\delta)} = \{t \in T | f(t) = (a, b, c, d), a < \alpha, b < \beta, c > \gamma, d > \delta\}.$$

In this case, as we increased γ *or* δ, *we would have fewer days included in* $X_{(\alpha,\beta,\gamma,\delta)}$, *considering only the days where the low value and closing value were sufficiently high. Accordingly we have inclusion maps from* $X_{(\alpha,\beta,\gamma,\delta)}$ *to* $X_{(\alpha',\beta',\gamma',\delta')}$ *whenever* $\alpha \leq \alpha'$, $\beta \leq \beta'$, $\gamma \geq \gamma'$ *and* $\delta \geq \delta'$. *See Chapter 7 for a visualization of such data using two parameters at a time, as well as a walkthrough of CPH on the same example.*

Example 2.1.3. *Suppose that we have obtained seismograms for a given period of time, which is the setting for [7]. In particular, suppose we have ground motion information in the east-west direction, the north-south direction, and the vertical direction. Then our seismogram is in fact a function* $f : T \longrightarrow \mathbb{R}^3$.

In this instance, though, we could consider a more interesting function. Suppose we know the direction from our recording instrument to the site of a seismic event that occurred during our recording time. We can then define the radial axis to be the projection of that direction onto the plane perpendicular to the vertical axis, and the transverse axis as the direction perpendicular to both the radial axis and the transverse axis. We could then apply a rotation to our data and obtain a new function $g : T \longrightarrow \mathbb{R}^3$ *that gives the ground motion information along the radial, transverse, and vertical axes instead. Figure 2.1 illustrates a simplified seismogram curve, with both sets of axes, where the radial axis points in the direction of the event.*

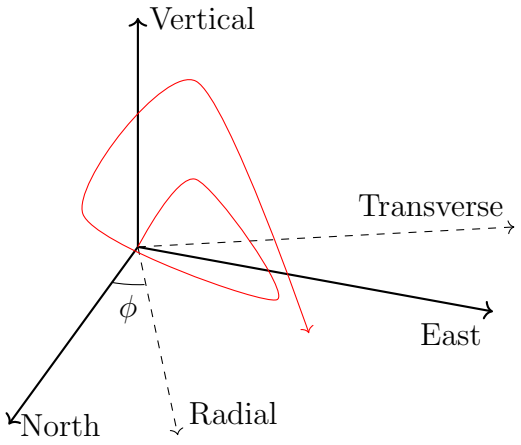

Figure 2.1: An Example Seismogram

This choice of function prompts us to consider how the multifiltration structures for f *and* g *compare, and subsequently how our analysis of these structures depends on our choice. We will call a process stable if we can bound the difference in our analysis by a multiple of distance between* f *and* g*. That is, a process is stable if we can define a distance on the output and on the input so that the former is bounded by the latter.*

Many methods exist for studying such structures by reducing them first to single-dimensional structures. Usually these methods involve fixing certain parameter values or calculating some new parameter value from the original inputs. Of course, this requires knowing beforehand the right choice for these parameter values or the correct formula for combining them. Multidimensional persistence and CPH bypass these concerns by considering each parameter simultaneously and then applying persistence to analyze how the multifiltration changes over every observed value of the parameters.

2.2 Approach

We seek to employ the general mathematics approach of creating a context-free tool for studying the structure of a multifiltration. In particular, we follow a similar approach to [2] by identifying multifiltrations using an *invariant*, a rule that associates equivalent objects to equivalent multifiltrations. Of course, we would also like our invariant to assign different objects to different multifiltrations. Such an invariant is called *complete*.

Two easy examples of invariants are the *trivial* invariant and the *identity* invariant. The trivial invariant assigns the same object to all multifiltrations. This is an incomplete invariant and cannot be used to distinguish the structure of a multifiltration. The identity invariant assigns a multifiltration to itself. This is a complete invariant, but does not contribute anything new to our ability to distinguish the structure of a multifiltration. The primary issue with the identity invariant is that it preserves the context of the multifiltration. Accordingly, we also desire our invariant to be context-free and, in order to actually compute the invariant, finite. We shall follow the convention from [2] and call such invariants *discrete*.

We might now state our goal as finding a complete discrete invariant for multifiltrations. However, [2] demonstrates that such an invariant does not exist for multifiltrations in general. Therefore, we refine our objective in two ways. First, we consider only multifiltrations of time-series as described earlier. Second, our classification will be for multifiltrations together with a total order. A *total order* is a rule that decides how to compare any two objects in a

consistent manner. A *partial order*, by contrast, only allows comparisons for certain pairs of objects. These concepts are explicitly defined in Section 2.3.

In a multifiltration where each parameter is ordered, the *product order* provides a natural partial order: level $(a_1, a_2, \ldots, a_n)$ precedes level $(b_1, b_2, \ldots, b_n)$ if $a_i \leq b_i$ for all i and for some k, $a_k < b_k$. However, this rule does not allow us to compare, for instance, $(1, 2, 3)$ and $(3, 2, 1)$, so this is only a partial order. If we denote the product order by $\prec$, then $(1, 2, 3) \prec (1, 3, 3)$ and $(1, 2, 3) \not\prec (3, 2, 1)$.

To obtain a complete and discrete invariant, we will choose a total order on the multifiltration that agrees with the product order in the following sense: if $a \prec b$, then a is less than b in the total order as well. The *lexicographical order* on a product space is one example of a total order that extends $\prec$, namely, $(a_1, a_2, \ldots, a_n)$ is less than $(b_1, b_2, \ldots, b_n)$ if $a_i < b_i$ the first time that $a_i \neq b_i$. For example, $(1, 2, 3)$ is less than $(1, 3, 3)$ and these are both less than $(3, 2, 1)$.

2.3 Orders

A *weak partial order* is a binary relation $\preceq$ that is reflexive, antisymmetric, and transitive. That is to say, given objects u, v, w, (1) $u \preceq u$, (2) if $u \preceq v$ and $v \preceq u$, then $u = v$, and (3) if $u \preceq v$ and $v \preceq w$, then $u \preceq w$. We obtain a strong partial order $\prec$ from $\preceq$ by defining $u \prec v$ if $u \preceq v$ and $u \neq v$.

On the other hand, a *weak total order* is a weak partial order that is also *connex*, that is to say, for any u, v, either u precedes v or v precedes u. We obtain a strong total order from a weak total order in the same way we did for the partial orders: $u < v$ if $u \leq v$ and $u \neq v$. As explained above, the lexicographic order on $\mathbb{R}^p$ is a strong total order.

Connecting these two types of orders is the concept of *extending an order*. We say that a total order extends a partial order if u preceding v in the partial order implies u precedes v in the total order. For example, with $\preceq$ as the product order on $\mathbb{R}^p$ and $\preceq\cdot$ as the lexicographic order, we see that $u \preceq v$ implies $u \preceq\cdot v$.

It is important to note that in an application, $\preceq$ does not depend on the order we have chosen for recording measurements, since it considers all coordinates simultaneously. On the other hand, the lexicographic order does respect the order in which we have recorded measurements. In this sense, one may consider multidimensional persistence as the study of multifiltrations given only the product order, while CPH is the study of multifiltrations given a total order that extends the product order.

Before leaving the discussion on orders, we should also present the idea of an *induced order*. Suppose $\preceq$ is an order on a set A and $f : B \to A$. Then we say an order $\sqsubseteq$ on B is induced by f from $\preceq$ if $u \sqsubseteq v$ implies $f(u) \preceq f(v)$. If it is necessary to indicate what function is being used, we will write $\sqsubseteq_f$. Note that there are in general multiple orders that are induced by the same function f, but we will not need to consider more than one such order at a time.

With these notions in mind, we will adopt the following conventions: we will take the product order as our partial order for $\mathbb{R}^p$ and use $\preceq$ to refer exclusively to this order. We will use $\preceq\cdot$ to denote a total order on $\mathbb{R}^p$ that extends $\preceq$, and we will use $\sqsubseteq$ to denote a partial order induced by $\preceq$. Finally, for a particular choice of $\sqsubseteq$, we will show that for any $\preceq\cdot$, there exists a total order $\sqsubseteq\cdot$ that is induced by $\preceq\cdot$ and extends $\sqsubseteq$.

2.4 Joins, Meets, and Scott Continuity

Given any partially ordered set, P, the *join* of a subset S is the least upper bound of S in the given partial order, if it exists, and is denoted $\bigvee S$. The *meet* of S is the greatest lower bound, if it exists, and is denoted $\bigwedge S$. In particular, if we consider the product order on $\mathbb{R}^p$, and let π_i denote projection to the ith coordinate, then

$$\pi_i\Big(\bigvee S\Big) = \sup\{\pi_i(a) : a \in S\} \quad \text{and} \quad \pi_i\Big(\bigwedge S\Big) = \inf\{\pi_i(a) : a \in S\}.$$

For example, if $S = \{(1,2,3),(3,1,2),(2,3,1)\}$, then $\bigvee S = (3,3,3)$ and $\bigwedge S = (1,1,1)$.

A subset S is called an *upward directed set* if for every pair of elements $a, b \in S$, there exists some $c \in S$ such that a and b both precede c. Now, given a function $\phi : \mathbb{R}^p \longrightarrow \mathbb{R}^p$, we say ϕ is *Scott-continuous* if for every directed set $S \subset \mathbb{R}^p$, we have

$$\phi\Big(\bigvee S\Big) = \bigvee \phi(S).$$

For example, the map ϕ that swaps a pair of coordinates is Scott-continuous with respect to the product order on $\mathbb{R}^p$, while the map that negates each coordinate is not Scott-continuous with respect to that order. The interested reader is referred to [8] for more details on Scott-continuity and related concepts, though we will only need the definition of Scott-continuity and the next definition for this paper.

Given functions $f, g : T \to \mathbb{R}^p$, we say that g is a *Scott-adjustment* of f if there exists a Scott-continuous function $\phi : \mathbb{R}^p \to \mathbb{R}^p$ such that $g = \phi \circ f$. We call the function ϕ the *Scott-adjuster*. If ϕ is injective on the image of f, then we say ϕ is an *injective Scott-adjuster*.

Chapter 3. Classical Persistence Theories

In this chapter, we briefly review the major points from the theories of persistent homology (PH) and multidimensional persistent homology (MPH). We will present the theories in parallel with how we develop coordinated persistent homology (CPH), rather than give a full exposition. The interested reader is referred to the excellent survey article [9].

3.1 Filtrations and Multifiltrations from Data

The first step in either persistence theory is to obtain a filtration or multifiltration. With our notation for the product order $\preceq$, we can restate the definition of a multifiltration as follows. A *multifiltration* is a collection of objects $\{A_u | u \in \mathcal{U} \subset \mathbb{R}^p\}$ together with maps $\phi_u^v : A_u \to A_v$ whenever $u \preceq v$ such that $\phi_v^x \circ \phi_u^v = \phi_w^x \circ \phi_u^w$ whenever $u \preceq v \preceq x$ and $u \preceq w \preceq x$. A *filtration* is a multifiltration in which the indexing set $\mathcal{U}$ is a subset of $\mathbb{R}$, and so we will henceforth use the term multifiltration as the general term.

While there are many ways to obtain a multifiltration from data, one simple method is to consider a function $f : X \to \mathbb{R}^p$, where X is a topological space representing the data points and f measures some quantity of interest, and then define $X_u = \{x \in X | f(x) \preceq u\}$, with ϕ_u^v being the inclusion map. For example, we could consider a function that gave the elevation and temperature of the surface of the earth at some fixed time given a latitude, longitude. Then our function could be described as $f : S^2 \to \mathbb{R}^2$. In turn this would give us a multifiltration depending on elevation and temperature. Of course, we usually either do not such a function or we might not know which space best represents our data.

Because experimental data is discrete, we typically envision our domain as a simplicial complex X with a 0-simplex for each data point and then including higher-dimensional simplices as needed. Given the values of f on the 0-simplex, we must then extend the map to any simplices that we introduce. While there are various methods for doing so, we will limit our current discussion to a generalization of the *Rips complex* described in [10]. That

is, given a function f defined on the 0-skeleton of a simplicial complex, we extend f to the entire complex by

$$f([x_1x_2\cdots x_k]) = \bigvee\{f(x_i)|1 \leq i \leq k\}.$$

In other words, the value assigned to a k-simplex is the least upper bound of the values on its faces, and so in turn it is the least upper bound of the values on its vertices. Our corresponding multifiltration is then constructed from spaces $X_u = \{\Delta \in X | f(\Delta) \preceq u\}$ with the natural inclusion maps as ϕ_u^v. More generally, we might start with a function defined on the n-skeleton of a complex and then extend it to the higher dimensional simplices. We illustrate the latter approach in Example 3.1.1.

Example 3.1.1. *Suppose we have collected survey data, where each response is encoded as a pair of coordinates. We may wish to know how the responses cluster, so we let $f(a) = 0$ for each vertex a, $f([a,b])$ to be the distance between a and b, and then follow the Rips construction for higher simplices. We can visualize this particular construction quite nicely using circles around each point and increasing the radius gradually. In Figure 3.1, we see two levels of the multifiltration represented. If two circles intersect, then the edge between the corresponding points is present at that level. Likewise, if three circles all pairwise intersect, then we have the corresponding triangle as well.*

Figure 3.1: Two levels of the Rips multifiltration

We see that at the second level shown, we have captured the two cycles that appear to represent our data. However, we do not yet have a reason to prefer the second level over the first. This is where the various persistence theories come in, by providing a way to measure how strong a given pattern is. In this example, persistence for a cycle corresponds to the range of radii for which the cycle is present and not yet filled in.

3.2 Taking Homology

Given a multifiltration of simplicial complexes modeling our data, we may now compute the simplicial homology of each complex X_u and form a new multilfiltration from $H(X_u)$ and the induced maps on homology $(\phi_u^v)_* : H(X_u) \to H(X_v)$. It is in this setting that we may identify when topological features appear and disappear from the multifiltration. Intuitively, if we think of the generators of each $H(X_u)$ as a topological feature, then the feature a appears at u if a is a generator of $H(X_u)$ and for all $v \preceq u$, $a \notin (\phi_v^u)_*(H(X_v))$. Likewise, if a is a generator that appears at u, then we might think of a disappearing at v if $(\phi_u^v)_*(a) = 0$ and for $u \preceq w \preceq v$ we have $(\phi_u^w)_*(a) \neq 0$.

If we are considering one-dimensional data, as with PH, this intuition is pretty much correct if we choose the correct set of generators. That is, there exists a choice of generators of $H(X_u)$ for each u so that ϕ_u^w always sends a generator to either another generator or to 0 and if two generators map to the same element, then they both map to 0. Given such a choice of generators, we identify generators that are mapped to one another and call this equivalence class a *feature*. Then the *birth* value of a feature is the minimal u for which $H(X_u)$ contains a generator in the feature, and the *death* value of the feature is the maximal u for which $H(X_u)$ contains a generator in the feature. In particular, because $\preceq$ is in fact a total order on $\mathbb{R}$, each feature has at most one birth value and one death value, so there is no ambiguity in the assignment. Thus we obtain a list of birth-death pairs corresponding to each equivalence class of generators of the various $H(X_u)$ groups.

If we are considering multidimensional data, our biggest issue with this intuition is that a feature a may disappear at multiple incomparable locations $v_1, v_2, \ldots v_n$. The second issue is that there may be interactions between $H(X_{v_1})$ and $H(X_{v_2})$ in $H(X_{\curlyvee v_1 v_2})$. Fortunately the first issue is easily remedied by considering a list of birth-deaths pairing when each generator is allowed to have an arbitrary number of death values. Unfortunately, for general multifiltrations the interaction between different incomparable homology groups is not easily resolved, as proven in [2].

3.3 One Dimensional Persistent Homology Algorithm

We shall describe the PH algorithm following a slight modification of the sparse matrix implementation given in [11]. Our presentation is designed to highlight the similarity between the classic algorithm and the CPH algorithm we introduce in Chapter 5.

Let X be a p-dimensional simplicial complex with a map $f : X \to \mathbb{R}$ and a filtration $\emptyset = K_0 \subset K_1 \subset \cdots \subset K_m = X$ such that

- f is constant on each simplex and injective;
- for all $\sigma_i \in K_\ell$, $\sigma_j \in X - K_\ell$, $f(\sigma_i) \leq f(\sigma_j)$; and
- if σ_i is a face of σ_j, then $f(\sigma_i) \leq f(\sigma_j)$.

Then we pull back the total order on $\mathbb{R}$ to a total order X by defining $\sigma_i \leq \sigma_j$ if $f(\sigma_i) \leq f(\sigma_j)$.

Now fix some dimension n with $0 \leq n < p$ and consider the ordered list of $(n+1)$-simplices. We will eventually include chains of simplices in this list as well, so we will call it OrderedChainList. Given a chain α, we will define $\max(\alpha)$ to be the maximal simplex that appears in the boundary of α. Given two chains α, β with a common boundary element, let $\alpha \sqcup \beta$ denote the formal sum of these chains with coefficients in $\mathbb{Z}_2$.

We will now recursively define a function $\mathcal{D}_K$ that sends n-simplices to lists of $(n+1)$-chains. In describing the algorithm that defines $\mathcal{D}_K$, we will use $a \leftarrow b$ to mean that the variable a is assigned the value of b. We will use OrderedChainList$[i]$ to denote the ith element of OrderedChainList, where indexing begins at 0. The actual algorithm is provided in Algorithm 1.

Now for each simplex $\alpha \in X$, $(f(\alpha), f(\mathcal{D}_K[\alpha]))$ gives us a (birth, death) pair of real numbers if $\mathcal{D}_K[\alpha]$ has been defined, and otherwise we use the pair $(f(\alpha), \infty)$. The list of all such pairs is commonly referred to as the *barcode* for X, since it can be visualized as a series of parallel intervals, as demonstrated in Example 3.3.1. We can also consider these pairs as points in the extended real plane. Formally, we also include the diagonal $\mathbb{R} \times \mathbb{R}$ with infinite

Algorithm 1 n-th Persistent Homology Algorithm

```
while OrderedChainList ≠ ∅ do
    curChain ← OrderedChainList[0]
    Delete OrderedChainList[0] from OrderedChainList.
    target ← max(curChain).
    while D_K[target] ≠ ∅ do
        oldChain ← the 0th entry of D_K[target].
        curChain ← curChain ⊔ oldChain.
        target ← max(curChain).
    end while
    D_K[target] ← [curChain].
end while
```

multiplicity as part of the barcode. This means that if Y is obtained from X by subdivision, then the barcodes for X and Y are equivalent since Y can only gain points on the diagonal.

Example 3.3.1. *Suppose X is the 3-simplex with vertices $\{\sigma_1, \sigma_2, \sigma_3, \sigma_4\}$, with $f(\sigma_i) = i$, $f([\sigma_i, \sigma_j]) = ij + 3$, $f([\sigma_i, \sigma_j, \sigma_k]) = ijk + 10$, and $f([1,2,3,4]) = 35$. It is easy to verify that f satisfies the requirements for the PH algorithm, and the total order on the simplices is*

$$1 < 2 < 3 < 4 < [1,2] < [1,3] < [1,4] < [2,3] < [2,4] < [3,4] <$$

$$[1,2,3] < [1,2,4] < [1,3,4] < [2,3,4] < [1,2,3,4].$$

Now applying Algorithm 1 to the 0-simplices proceeds as follows. We start with all six 1-simplices in OrderedChainList, and consider the smallest of these, $[1,2]$. $\max([1,2]) = 2$, and since we have not yet defined any values for $\mathcal{D}_K$, we make the assignment $\mathcal{D}_K[2] = [1,2]$. Now we proceed similarly for $[1,3]$ and $[1,4]$ since their maximum faces are 3 and 4, respectively, and $\mathcal{D}_K$ has not been defined for either of these values.

The next edge on our list, though, is $[2,3]$ and this is where things become interesting. Since $\max([2,3]) = 3$ and $\mathcal{D}_K[3]$ is already defined as $[1,3]$, we enter the inner loop and consider $[1,3] \sqcup [2,3]$. The boundary of $[1,3] \sqcup [2,3]$ consists of 1 and 2, so $\max([1,3] \sqcup [2,3]) = 2$. $\mathcal{D}_K[2] = [1,2]$, so we repeat the inner loop once more to obtain the chain $[1,3] \sqcup [2,3] \sqcup [1,2]$. However, this has no boundary, so there is no target and no assignment is made before we repeat the main loop and consider $[2,4]$. We do note, though, that we have found a possible

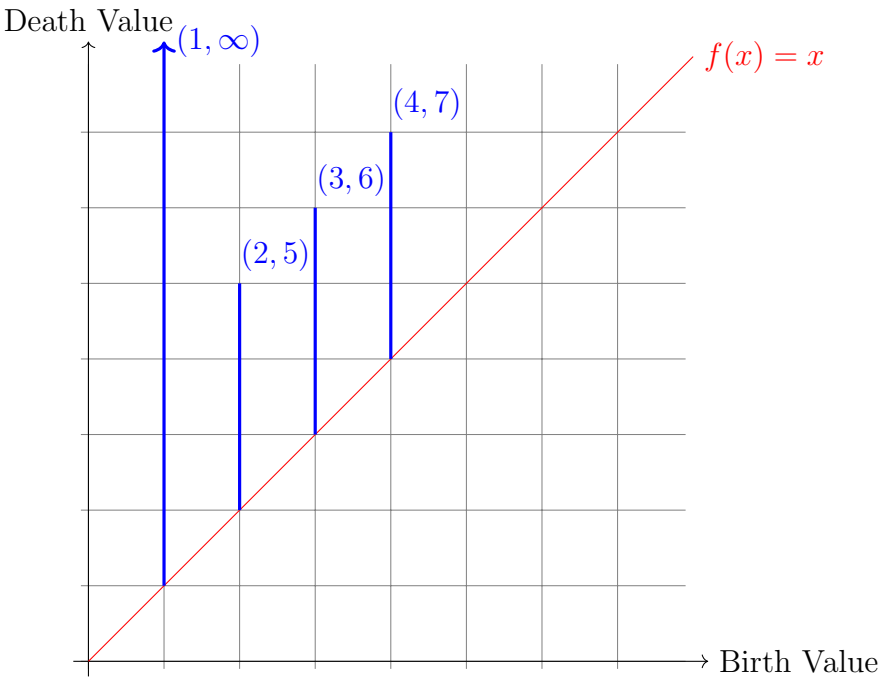

Figure 3.2: A visualization of the barcode

generator for $H_1(X; \mathbb{Z}_2)$ *and if desired we could track this information to provide the actual choice of generators for* $H(X; \mathbb{Z}_2)$ *described in Section 3.2.*

The algorithm proceeds similarly on $[2, 4]$ *and* $[3, 4]$*:* $\max([x, 4]) = 4$ *and* $\mathcal{D}_K[4] = [1, 4]$*, so we enter the inner loop and consider* $[1, 4] \sqcup [x, 4]$*. Since* $\max([1, 4] \sqcup [x, 4]) = x$ *and* $\mathcal{D}_K[x] = [1, x]$*, we form another chain with no boundary,* $[1, 4] \sqcup [x, 4] \sqcup [1, x]$*, so no new assignments are made.*

This leaves us with $\mathcal{D}_K$ *sending* $1 \mapsto \emptyset$*,* $2 \mapsto [1, 2]$*,* $3 \mapsto [1, 3]$ *and* $4 \mapsto [1, 4]$*. This corresponds precisely with the fact that the component generated by* 2 *is identified with the component generated by* 1 *when the edge* $[1, 2]$ *enters the filtration, and likewise for* 3 *and* 4*, while the component generated by* 1 *yields a generator of* $H_0(K_\ell)$ *for all* ℓ *once* 1 *has appeared. Conversely, edges such as* $[2, 3]$ *appear only after the components generated by* 2 *and* 3 *have already been identified, and hence do not appear as the output of* $\mathcal{D}_K$*.*

Thus the 0*-dimensional part of our barcode is* $(1, \infty)$*,* $(2, 5)$*,* $(3, 6)$*, and* $(4, 7)$*. Figure 3.2 illustrates one way to visualize this information, and helps explain why this is called a barcode.*

We close this section with a comment on the complexity of the algorithm. As noted in [1] and [11], the PH algorithm is a cubic time algorithm in the worst case. Furthermore, [12] provides an explicit example of a filtration with N simplices for which the PH algorithm requires $\Omega(N^3)$ operations.

3.4 Multidimensional Persistent Homology Algorithm

For multidimensional persistent homology, the description of the algorithm itself is less enlightening and so we refer the reader to [5] for an implementation of such an algorithm. However, we will give an overview of the process and discuss the complexity of the algorithm given in [5] so that we may later compare it with the complexity of the algorithm we present for coordinated persistent homology.

Let K be a p-dimensional multi-filtered complex with some multifiltration consisting of subcomplexes $K_u \subset K$ with inclusion maps $\phi_u^v : K_u \to K_v$. In order to compute the multidimensional persistent homology of K, we must compute the boundary module $\operatorname{im} \partial_{i+1}$, the cycle module $\ker \partial_i$, and finally the quotient $H_i = \ker \partial_i / \operatorname{im} \partial_{i+1}$. While the sparse matrix implementation for one dimensional persistence does this computation quite nicely, the presence of multiple parameters means that we would, at the least, require a sparse multi-dimensional array. However, [2] shows that such an implementation would still be insufficient to capture a discrete, complete invariant for K.

Accordingly, we are less concerned with the actual implementation of multidimensional persistent homology than with the inherent complexity of such solutions. In other words, we wish to generalize persistent homology in a simpler fashion than multidimensional persistence does. To that end, it suffices to cite the complexity estimates given in [5]. For a given dimension, let m be the number of simplices of that dimension, and let n be the number of simplices one dimension higher. Then from Lemmas 4 and 6 of [5], the run-time of the primary components of the multidimensional persistence algorithm are $O(n^4m^3)$ and $O(n^4m^2)$ in the worst cases, so the overall algorithm is $O(n^4m^3)$.

3.5 Stability

Perhaps the most useful fact about persistent homology (PH) and multidimensional persistent homology (MPH) are the stability theorems, which state how small changes of the input affect the output. Since both theories arose from a need to analyze real world data, it is important to understand how things like background noise and instrument error affect our analysis. In essence, the stability theorems state that such errors translate to either equal error on our analysis or less. We first need to understand how to measure distances for functions and for barcodes. We will then state the stability theorems and discuss their implications, but we will refer the reader to [13] and [6] for rigorous proofs of the PH and MPH stability theorems, respectively.

First, given $x, y \in \mathbb{R}^p$, let $\|x-y\|_\infty = \max\{|x_i - y_i| : 1 \leq i \leq p\}$. Then, for functions $f, g : X \longrightarrow \mathbb{R}^p$, let $\|f-g\|_\infty = \sup_x \|f(x) - g(x)\|_\infty$. Now for two barcodes B_f and B_g, based on f and g respectively, we define the *bottleneck distance*, $d_B(f,g)$, by

$$d_B(f,g) = \inf_\gamma \sup_x \|x - \gamma(x)\|_\infty,$$

where x ranges over all points in B_f and γ ranges over all bijections from B_f to B_g.

Next, for a topological space X and a continuous function $f : X \to \mathbb{R}$, a *homological critical value* of f is a value $a \in \mathbb{R}$ such that for some $k \in \mathbb{Z}$ and all sufficiently small $\epsilon > 0$, the map $H_k(f^{-1}(-\infty, a-\epsilon]) \to H_k(f^{-1}(-\infty, a])$, induced by inclusion, is not an isomorphism. In other words, the homology of the corresponding filtration changes at a. We say f is *tame* if there are a finite number of homological critical values and $H_k(f^{-1}(-\infty, a])$ is finite-dimensional for all $k \in \mathbb{Z}$ and $a \in \mathbb{R}$. These conditions are satisfied by Morse functions on compact manifolds and piecewise linear functions on finite simplicial complexes.

Theorem 3.5.1 (PH is Stable with Respect to the Bottleneck Distance)**.** *If X is a triangulable space with continuous tame functions $f, g : X \to \mathbb{R}$, then the barcodes B_f and B_g satisfy $d_B(f,g) \leq \|f-g\|_\infty$.*

Theorem 3.5.2 (MPH is Stable With Respect to some Distance)**.** *If X is a triangulable space with continuous functions $f, g : X \to \mathbb{R}^p$, then there exists a metric on rank invariants such that the distance between the rank invariant for f and g is at most $\|f - g\|_\infty$.*

An appropriate metric for the rank invariant that satisfies Theorem 3.5.2 is constructed in [6], and is essentially the bottleneck distance with bijections taken between rank invariants. The important thing is that both the bottleneck distance and the metric on rank invariants work by considering bijections to compare the output of the persistence theorems. In particular, these stability theorems imply that small changes in a function do not compound themselves. Conversely, if we detect a significant feature, something that is far from the diagonal, we can infer that the feature represents something inherent to the data being studied, not a false pattern from background noise or small errors in our measurement device. Of course, this presupposes that background noise is of small magnitude relative to the data. By contrast, if we have a poor signal to noise ratio, the stability theorems mean that neither persistence theory can confidently extract information about the data from the overall signal.

Chapter 4. Labeled Linear Graphs

The first step in our generalization of persistent homology to multiple parameters is to provide a bit more structure on our multifiltration than either of the previous persistence theories implement. Furthermore, for the current implementation, we consider only multifiltrations of lines, so we will bypass the simplicial complex description of our space and instead define a richer structure, which we call a *labeled linear graph*. We will then demonstrate how to construct a certain kind of total order on the labeled linear graph and show that our definition and construction are stable in certain senses.

Recall that $\preceq$ is the product order on $\mathbb{R}^p$, namely $(a_1, a_2, \ldots, a_p) \preceq (b_1, b_2, \ldots, b_p)$ if $a_i \leq b_i$ for $1 \leq i \leq p$. We will use $\preceq\!\!\cdot$ to denote a total order on $\mathbb{R}^p$ that extends $\preceq$, and $\curlyvee$ and $\curlywedge$ denote the join and meet operations, respectively. Note that these 'curved' symbols all apply to $\mathbb{R}^p$. In this chapter we will construct various orders and operations on a graph and we will use 'squared' symbols for these as a visual reminder of which orders apply to which space. In particular, $\sqsubseteq$ and $\sqsubseteq\!\!\cdot$ will denote certain orders that depend in part on $\preceq$ and $\preceq\!\!\cdot$, and $\sqcup$ will denote a particular operation that relates to $\curlyvee$.

4.1 Time-Series and Labeled Linear Graphs

A *time-series* is a function $f : T \longrightarrow \mathbb{R}^p$ from a discrete set $T \subset \mathbb{R}$. For the remainder of this discussion, f will denote a given time-series with domain T. Furthermore, we let $f_i : T \longrightarrow \mathbb{R}$ denote the projection of f to the ith coordinate of $\mathbb{R}^p$ and write $f(t) = (f_i(t))$. The assumption that T is discrete reflects the idea that T represents a set of times at which measurements are made for some experiment.

We define a graph Γ_f whose vertex set is T and whose edge set consists of pairs $[t, t']$ where t' is the successor to t in T. We also define the set of *paths* in Γ_f, denoted P_f, to be elements of the form $[r, t]$ where $r, t \in T$ and $r < t$, with the edge set as an obvious subset. We also extend Γ_f to include P_f.

We next define some operations for paths. Given a path $x = [r, t]$, we let $\overleftarrow{x} = r$, $\overrightarrow{x} = t$, $\eth(x) = \{x \in T | r \leq x \leq t\}$, and $\partial(x) = \{r, t\}$. Given $a, b \in \mathrm{P}_f$ with $\overrightarrow{a} = \overleftarrow{b}$, we define

$$a \sqcup b = b \sqcup a = [\overleftarrow{a}, \overrightarrow{b}].$$

Likewise, given $S = \{[t_0, t_1], [t_1, t_2], \ldots, [t_{n-1}, t_n]\}$, we define

$$\bigsqcup S = [t_0, t_n].$$

With this structure, we can now extend f from T to Γ_f, including P_f, by defining

$$f([t, t']) = f(t) \curlyvee f(t') \quad \text{and} \quad f(\bigsqcup S) = \curlyvee(f(S)),$$

where $f(S) = \{f(a) | a \in S\}$. Note that every path in P_f is of the form $\bigsqcup S$ for a unique S, so this is well-defined.

We call the pair (Γ_f, f) a *labeled linear graph structure on T relative to f*, or if f is understood from context, we just call this a labeled linear graph. We will write V_f, E_f, and P_f for the set of vertices, edges, and paths, respectively, with $\mathrm{E}_f \subset \mathrm{P}_f$ and $\mathrm{V}_f = T$.

Our first lemma tells us that the extension of f to Γ_f is *stable*: a small change in the original function f will only produce a small change in the extension.

Lemma 4.1.1. *Suppose Γ_f and Γ_g are labeled linear graph structures on T relative to f and g, respectively. Then Γ_f and Γ_g are equal as graphs and*

$$\left\| f|_{\mathrm{P}_f} - g|_{\mathrm{P}_g} \right\|_\infty \leq \|f|_T - g|_T\|_\infty = \|f - g\|_\infty,$$

where $\|\cdot\|_\infty$ denotes the appropriate uniform norms, i.e., for $h : X \longrightarrow \mathbb{R}^p$,

$$\|h\|_\infty = \sup\{h_i(x) : x \in X, 1 \leq i \leq p\}.$$

Proof. The definition Γ_f and Γ_g depend only on T, so we see immediately that they are equal as graphs. We will use f and g to denote the extended versions of the functions, and $f|_T$ and $g|_T$ to denote the original functions on T. Now, the definition of the uniform norm gives us

$$\|f|_T - g|_T\|_\infty \leq \|f - g\|_\infty \quad \text{and} \quad \left\|f|_{\mathrm{P}_f} - g|_{\mathrm{P}_g}\right\|_\infty \leq \|f - g\|_\infty,$$

since $\|f - g\|_\infty$ is taken over all of Γ_f.

Suppose $a \in \mathrm{P}_f$ with $a = [r, t]$. There exists $k \in \{1, 2, \ldots, p\}$ such that

$$\begin{aligned} \|f(a) - g(a)\|_\infty &= \|f_k(a) - g_k(a)\|_\infty && (4.1.2) \\ &= |\max f_k(\eth(a)) - \max g_k(\eth(a))|. && (4.1.3) \end{aligned}$$

If the maxima in Equation (4.1.2) both occur at some $s \in \eth(a)$, then

$$|\max f_k(\eth(a)) - \max g_k(\eth(a))| = \|f_k(s) - g_k(s)\|_\infty \leq \left\|f|_{\mathrm{V}_f} - g|_{\mathrm{V}_g}\right\|_\infty. \tag{4.1.4}$$

Otherwise, the maxima in Equation (4.1.2) occur at different points of $\eth(a)$, say x and y. Since the equation is symmetric in f and g, we may assume

$$\max f_k(\eth(a)) = f_k(x) \quad \text{and} \quad \max g_k(\eth(a)) = g_k(y). \tag{4.1.5}$$

We now consider the following cases.

4.1.1.1 If $g_k(y) \leq f_k(x)$, then $g_k(x) \leq g_k(y) \leq f_k(x)$ so

$$\begin{aligned} |\max f_k(\eth(a)) - \max g_k(\eth(a))| &= \|f_k(x) - g_k(y)\|_\infty \\ &\leq \|f_k(x) - g_k(x)\|_\infty \\ &\leq \|f|_T - g|_T\|_\infty. \end{aligned}$$

4.1.1.2 If $f_k(x) \leq g_k(y)$, then $f_k(y) \leq f_k(x) \leq g_k(y)$ so

$$\begin{aligned} |\max f_k(\eth(a)) - \max g_k(\eth(a))| &= \|f_k(x) - g_k(y)\|_\infty \\ &\leq \|f_k(y) - g_k(y)\|_\infty \\ &\leq \|f|_T - g|_T\|_\infty. \end{aligned}$$

Thus we see $\|f(a) - g(a)\|_\infty \leq \|f|_T - g|_T\|_\infty$ for all $a \in \Gamma_f$. Since $\|f - g\|_\infty$ is the supremum of the left hand side over a set containing T, we will have

$$\big\|f|_{\mathrm{P}_f} - g|_{\mathrm{P}_g}\big\|_\infty \leq \|f|_T - g|_T\|_\infty = \|f - g\|_\infty. \qquad \square$$

Lemma 4.1.6. *Suppose Γ_f and Γ_g are labeled linear graph structures on T relative to f and g, respectively. Then $g|_T$ is a Scott-adjustment of $f|_T$ if and only if g is a Scott-adjustment of f. Furthermore, the same Scott-adjuster may be used for both adjustments.*

Proof. Suppose $g|_T$ is a Scott-adjustment of $f|_T$ with Scott-adjuster ϕ. Then for any $t \in T$,

$$g(t) = g|_T(t) = \phi \circ f|_T(t) = \phi \circ f(t).$$

For any $[t, t'] \in \mathrm{E}_f$, since ϕ is Scott-continuous, we have

$$\phi \circ f([t, t']) = \phi(f(t) \curlyvee f(t')) = (\phi \circ f(t)) \curlyvee (\phi \circ f(t')) = g(t) \curlyvee g(t') = g([t, t']).$$

Likewise, if $a \in \mathrm{P}_f$, then $a = \bigsqcup S$ for some $S \subset \mathrm{E}_f$, and we have

$$\phi \circ f(a) = \phi(f(\textstyle\bigsqcup S)) = \phi\Big(\bigvee f(S)\Big) = \bigvee \phi(f(S)) = \bigvee g(S) = g(\textstyle\bigsqcup S).$$

The reverse direction follows immediately from the fact that $f|_T$ and $g|_T$ are restrictions of f and g. $\square$

4.2 Partial Order on Labeled Linear Graphs

Now that we have extended f to a labeled linear graph Γ_f, including P_f, we can use it to induce a partial order on Γ_f from $\preceq$ on $\mathbb{R}^p$. This will generally result in a different order on T than the one T inherits as a subset of $\mathbb{R}$, which we will indicate by $\leq$. Intuitively, we will start by defining $a \sqsubseteq b$ if $f(a) \preceq f(b)$, but to make this a partial order we will need to break ties in the cases where $f(a) = f(b)$ and $a \neq b$. Formally, we define $a \sqsubseteq b$ if any of the following hold

(4.2.1) $a = b$,

(4.2.2) $f(a) \prec f(b)$,

(4.2.3) $f(a) = f(b)$, $a \in \mathrm{V}_f$, and $b \in \mathrm{P}_f$, or

(4.2.4) $f(a) = f(b)$, $a, b \in \mathrm{V}_f$, and $a < b$.

If it is known that $a \sqsubseteq b$ and $a \neq b$, then we will write $a \sqsubset b$. Likewise, if it is important to note which function induced the partial order, we will write $a \sqsubseteq_f b$ to indicate that f was the applicable function.

Proposition 4.2.1 (Product Order Pulls Back to Graph)**.** *As defined above, the relation $a \sqsubseteq b$ is a partial order on $\Gamma_f \cup \mathrm{P}_f$. Furthermore, if $S \subset \mathrm{P}_f$, S contains at least two elements, and $\bigsqcup S$ is defined, then exactly one of the following hold:*

4.2.1.1 $f(a) = f(b)$ for all $a, b \in S$, or

4.2.1.2 $a \sqsubseteq \bigsqcup S$ for some $a \in S$.

Proof. Clearly $\sqsubseteq$ is reflexive.

To show that $\sqsubseteq$ is antisymmetric, suppose $a \neq b$ and $a \sqsubseteq b$. Since $a \neq b$, a and b must satisfy one of Condition (4.2.2) to Condition (4.2.4).

If Condition (4.2.2) holds, then $f(a) \prec f(b)$ and the antisymmetry of $\prec$ implies $b \not\sqsubseteq a$.

If Condition (4.2.3) holds, then $f(a) = f(b)$, $a \in \mathrm{V}_f$, and $b \in \mathrm{P}_f$. Thus $b \not\sqsubseteq a$ since $f(a) \not\prec f(b)$ and $b \notin \mathrm{V}_f$.

If Condition (4.2.4) holds, then $f(a) = f(b)$, $a, b \in \mathrm{V}_f$, and $a < b$. Thus $b \not\sqsubseteq a$ since $f(a) \not\prec f(b)$ and $<$ is antisymmetric.

Thus if $a \sqsubset b$ in any case, $b \not\sqsubset a$, so $\sqsubseteq$ is antisymmetric.

To establish transitivity, suppose $a \sqsubseteq b$ and $b \sqsubseteq c$, but $a \not\sqsubseteq c$. Clearly $a = b$, $b = c$, or $a = c$ implies $a \sqsubseteq c$. Therefore, we know a, b, c are all distinct and each pair (a, b) and (b, c) must satisfy one of the other three conditions. In particular, we know that in any case

$$f(a) \preceq f(b) \preceq f(c). \tag{4.2.2}$$

From Equation (4.2.2) and the transitivity of $\prec$, then $f(a) \prec f(b)$ or $f(b) \prec f(c)$ will imply that $f(a) \prec f(c)$ and hence $a \sqsubseteq c$, a contradiction. Therefore Condition (4.2.2) does not apply to either of the pairs (a, b) or (b, c). Thus (a, b) and (b, c) must each satisfy one of the remaining two conditions and so we know that $f(a) = f(b) = f(c)$.

Since (a, b) must satisfy Condition (4.2.3) or Condition (4.2.4), we must have $a \in \mathrm{V}_f$. Likewise, (b, c) must satisfy Condition (4.2.3) or Condition (4.2.4), so $b \in \mathrm{V}_f$. Thus if $c \in \mathrm{P}_f$, we have $a \sqsubseteq c$ by Condition (4.2.3), a contradiction. Hence we know that $c \in \mathrm{V}_f$.

Since $a, b, c \in \mathrm{V}_f$, and $f(a) = f(b) = f(c)$, then (a, b) and (b, c) must satisfy Condition (4.2.4). The transitivity of $<$ then implies that $a \sqsubseteq c$, which is our final contradiction. Thus we see that $\sqsubseteq$ is indeed transitive.

For the other part of our claim, suppose $S \subset \mathrm{P}_f$, $\bigsqcup S$ is defined, and for some $a, b \in S$, $f(a) \neq f(b)$. Then without loss of generality there exists some k for which

$$f_k(a) < f_k(b) \leq f_k(\bigsqcup S).$$

In addition, since $f_i(a) \leq f_i(\bigsqcup S)$ for all i, then $f(a) \prec f(\bigsqcup S)$. Thus, by definition of $\sqsubseteq$, $a \sqsubseteq \bigsqcup S$.

Conversely, if $f(a) = f(b)$ for all $a, b \in S$, then $f(a) = \bigvee f(S) = f(\bigsqcup S)$. Since S contains at least two elements, $a \neq \bigsqcup S$, so $a \not\sqsubseteq \bigsqcup S$. □

Lemma 4.2.3. *If g is a Scott-adjustment of f with injective Scott-adjuster ϕ, then $\forall a, b \in \Gamma_f$, $a \sqsubseteq_f b$ if and only if $a \sqsubseteq_g b$.*

Proof. Clearly, if $a = b$ then both $a \sqsubseteq_f b$ and $a \sqsubseteq_g b$, so Condition (4.2.1) holds under f if and only if it holds under g.

Likewise, since ϕ is injective on the image of f, we know that $f(a) = f(b)$ if and only if $g(a) = g(b)$. This implies that Condition (4.2.4) and Condition (4.2.3) hold for f if and only if they hold for g.

Finally, we must show that Condition (4.2.2) holds for f if and only if it holds for g. That is, we must show $f(a) \prec f(b)$ if and only if $g(a) \prec g(b)$. Because ϕ is Scott-continuous, we know that

$$g(a) \curlyvee g(b) = \phi \circ f(a) \curlyvee \phi \circ f(b) = \phi(f(a) \curlyvee f(b)).$$

Additionally, the definition of the join operator implies

$$f(a) \preceq f(a) \curlyvee f(b) \quad \text{and} \quad g(a) \preceq g(a) \curlyvee g(b).$$

Now, if $f(a) \prec f(b)$, then $f(a) \curlyvee f(b) = f(b)$ and $f(a) \neq f(b)$. Therefore the injectivity of ϕ and the transitivity of $\preceq$ implies $g(a) \prec \phi(f(b)) = g(b)$.

Conversely, suppose $f(a) \not\prec f(b)$, so $f(a) \curlyvee f(b) \neq f(b)$. Since ϕ is Scott-continuous,

$$\phi(f(a) \curlyvee f(b)) = \phi(f(a)) \curlyvee \phi(f(b)) = g(a) \curlyvee g(b),$$
$$\text{and} \quad \phi(f(b)) = g(b).$$

However, ϕ is injective on the image of f, so $g(a) \curlyvee g(b) \neq g(b)$, which means $g(a) \not\prec g(b)$. □

4.3 Total Order on Labeled Linear Graphs

The last construction we need to complete our labeled linear graph is to pull back a total order from $\mathbb{R}^p$ to Γ_f. More precisely, given a total order $\preceq\cdot$ on $\mathbb{R}^p$ that extends $\preceq$, we will show that there exists a total order $\sqsubseteq\cdot$ on Γ_f induced by $\preceq\cdot$ and that extends $\sqsubseteq$. We note our convention for order symbols means that any order on $\mathbb{R}^p$ is curvy, while any order on Γ_f is linear. We hope that this convention will aid the reader in determining what elements are being considered for a given order statement. For example, $a \sqsubseteq b$ presupposes that $a, b \in \Gamma_f$.

We also adopt another convention regarding orders: for any order relation $\sim$ on a space X and $A \subset X$, $\min_\sim A$ is the minimum of A with respect to $\sim$, if it exists. We define $\max_\sim$, $\inf_\sim$ and $\sup_\sim$ similarly.

Now, given Γ_f a labeled linear graph structure on T and $\preceq\cdot$ an extension of $\prec$, define the following subsets of $\Gamma_f \times \Gamma_f$.

$$
\begin{aligned}
S_0\,\Gamma_f &= \{(a,a) \in \Gamma_f \times \Gamma_f : a \in \Gamma_f\}.\\
S_1\,\Gamma_f &= \{(a,b) \in \Gamma_f \times \Gamma_f : f(a) \prec\cdot f(b)\}.\\
S_2\,\Gamma_f &= \{(a,b) \in \mathrm{V}_f \times \mathrm{P}_f : f(a) = f(b)\}\\
S_3\,\Gamma_f &= \{(a,b) \in \mathrm{V}_f \times \mathrm{V}_f : f(a) = f(b), a < b\}.
\end{aligned}
$$

For $a, b \in \mathrm{V}_f$ we define $a \sqsubseteq\cdot\, b$ if $(a,b) \in \bigsqcup_{i=0}^{3} S_i\,\Gamma_f$.

$$
\begin{aligned}
P_\diamond &= \{(a,b) \in \mathrm{P}_f \times \mathrm{P}_f : f(a) = f(b), a \neq b\}\\
X\,\Gamma_f &= \left\{(a,b) \in P_\diamond : f\left(\max_{\sqsubseteq\cdot}(\partial(a))\right) = f\left(\max_{\sqsubseteq\cdot}(\partial(b))\right)\right\}\\
S_4\,\Gamma_f &= \left\{(a,b) \in P_\diamond \setminus X\,\Gamma_f : f\left(\max_{\sqsubseteq\cdot}(\partial(a))\right) \prec\cdot f\left(\max_{\sqsubseteq\cdot}(\partial(b))\right)\right\}.\\
S_5\,\Gamma_f &= \left\{(a,b) \in X\,\Gamma_f : f\left(\min_{\sqsubseteq\cdot}(\partial(a))\right) \prec\cdot f\left(\min_{\sqsubseteq\cdot}(\partial(b))\right)\right\}.\\
S_6\,\Gamma_f &= \left\{(a,b) \in X\,\Gamma_f : f\left(\min_{\sqsubseteq\cdot}(\partial(a))\right) = f\left(\min_{\sqsubseteq\cdot}(\partial(b))\right), \overleftarrow{a} \leq \overrightarrow{b}\right\}.
\end{aligned}
$$

Now we finish defining $a \sqsubseteq\!\!\cdot\ b$ for $a, b \in \Gamma_f$: $a \sqsubseteq\!\!\cdot\ b$ if $(a, b) \in \bigsqcup_{i=0}^{6} S_i\,\Gamma_f$. Note that the $S_i\,\Gamma_f$ sets are mutually exclusive and that $a \sqsubseteq\!\!\cdot\ b$ implies $f(a) \preceq\!\!\cdot\ f(b)$. While this relation is not quite a total order on Γ_f, it is a total order on V_f and $\mathrm{V}_f \cup \mathrm{E}_f$ (see Proposition 4.3.2) and provides a sufficient ordering on some subsets of P_f for our purposes (see Theorem 5.1.1). As usual, if we need to specify which function was used to define $\sqsubseteq\!\!\cdot$, we will write $\sqsubseteq\!\!\cdot_f$; furthermore, $a \sqsubset\!\!\cdot\ b$ means $a \sqsubseteq\!\!\cdot\ b$ and $a \neq b$.

Proposition 4.3.1. *$\sqsubseteq\!\!\cdot$ is a total order on V_f.*

Proof. First, $\sqsubseteq\!\!\cdot$ is clearly connex on V_f from $S_0\,\Gamma_f$, $S_1\,\Gamma_f$, $S_3\,\Gamma_f$, and the fact that $\prec\!\!\cdot$ and $<$ are connex.

Next, suppose $a, b \in \mathrm{V}_f$ with $a \sqsubset\!\!\cdot\ b$. Since $a \neq b$, then $(a, b) \in S_1\,\Gamma_f \cup S_3\,\Gamma_f$. In either case, since $\prec\!\!\cdot$ and $<$ are antisymmetric, then $b \not\sqsubset\!\!\cdot\ a$, so $\sqsubseteq\!\!\cdot$ is antisymmetric on V_f.

Finally, suppose $a, b, c \in \mathrm{V}_f$ with $a \sqsubseteq\!\!\cdot\ b$ and $b \sqsubseteq\!\!\cdot\ c$. Clearly $a = b$ or $b = c$ immediately imply $a = c$ and thus $a \sqsubseteq\!\!\cdot\ c$, thus we may assume that a, b, c are all distinct and $f(a) \preceq\!\!\cdot\ f(b) \preceq\!\!\cdot\ f(c)$. Then, if $f(a) \prec\!\!\cdot\ f(b)$ or $f(b) \prec\!\!\cdot\ f(c)$, we have $f(a) \prec\!\!\cdot\ f(c)$ and thus $a \sqsubseteq\!\!\cdot\ c$ again. Thus we may now assume that $f(a) = f(b) = f(c)$, so $a < b$ and $b < c$. Since $<$ is transitive, this implies $a < c$ and thus $a \sqsubseteq\!\!\cdot\ c$. Therefore $\sqsubseteq\!\!\cdot$ is transitive on V_f. □

Proposition 4.3.2. *$\sqsubseteq\!\!\cdot$ is a total order on $\mathrm{V}_f \cup \mathrm{E}_f$.*

Proof. We will follow the same basic structure of the definition of $\sqsubseteq\!\!\cdot$ to show connexity and antisymmetry first, and then we will give a similar proof of transitivity. Let $a, b \in \mathrm{V}_f \cup \mathrm{E}_f$ be given, then consider each of the following cases.

Case 0: Suppose $a = b$. Clearly we have both $a \sqsubseteq\!\!\cdot\ b$ and $b \sqsubseteq\!\!\cdot\ a$. Thus we may now assume $a \neq b$.

Case 1: Suppose $f(a) \neq f(b)$. Since $\preceq\!\!\cdot$ is a total order, we must have exactly one of $f(a) \prec\!\!\cdot\ f(b)$ or $f(b) \prec\!\!\cdot\ f(a)$ and therefore exactly one of $a \sqsubseteq\!\!\cdot\ b$ or $b \sqsubseteq\!\!\cdot\ a$. Thus we may now assume $f(a) = f(b)$.

Case 2: Suppose $f(a) = f(b)$ and either $(a, b) \in \mathrm{V}_f \times \mathrm{E}_f$ or $(b, a) \in \mathrm{V}_f \times \mathrm{E}_f$. Depending on which element is the vertex, we have $a \sqsubseteq\!\!\cdot\ b$ or $b \sqsubseteq\!\!\cdot\ a$, but not both. Thus we may now assume $(a, b) \in \mathrm{V}_f \times \mathrm{V}_f \cup \mathrm{E}_f \times \mathrm{E}_f$.

Case 3: Suppose $f(a) = f(b)$, $(a, b) \in \mathrm{V}_f \times \mathrm{V}_f$, and $a \neq b$. Proposition 4.3.1 covers this case. Thus we may now assume $(a, b) \in \mathrm{E}_f \times \mathrm{E}_f$.

For the remaining cases, we define the following:

$$a_1 = \min_{\sqsubseteq\!\!\cdot}(\partial(a)), \qquad a_2 = \max_{\sqsubseteq\!\!\cdot}(\partial(a)),$$
$$b_1 = \min_{\sqsubseteq\!\!\cdot}(\partial(b)), \qquad b_2 = \max_{\sqsubseteq\!\!\cdot}(\partial(b)).$$

Recall that we have reduced to the cases where $a \neq b$, $f(a) = f(b)$, and $(a, b) \in \mathrm{E}_f \times \mathrm{E}_f$.

Case 4: Suppose $f(a_2) \neq f(b_2)$. $\preceq\!\!\cdot$ is a total order, so exactly one of $f(a_2) \prec\!\!\cdot\ f(b_2)$ or $f(b_2) \prec\!\!\cdot\ f(a_2)$ holds. This means that exactly one of $a \sqsubseteq\!\!\cdot\ b$ or $b \sqsubseteq\!\!\cdot\ a$ holds. Thus we may now assume $f(a_2) = f(b_2)$.

Case 5: Suppose $f(a_1) \neq f(b_1)$. $\preceq\!\!\cdot$ being a total order again implies exactly one of $f(a_1) \prec\!\!\cdot\ f(b_1)$ or $f(b_1) \prec\!\!\cdot\ f(a_1)$ holds and therefore exactly one of $a \sqsubseteq\!\!\cdot\ b$ or $b \sqsubseteq\!\!\cdot\ a$ holds. Thus we may now assume $f(a_1) = f(b_1)$.

Case 6: Now we have $f(a_1) = f(b_1)$ and $f(a_2) = f(b_2)$. Since we know $\leq$ is a total order on V_f, we may define the following

$$x_1 = \overrightarrow{a}, \qquad x_2 = \overleftarrow{a},$$
$$y_1 = \overrightarrow{b}, \qquad y_2 = \overleftarrow{b}.$$

Suppose that $y_1 < x_2$. Since $\leq$ is a total order, we have

$$x_1 \leq x_2, \quad y_1 \leq y_2, \quad y_1 < x_2.$$

Recall that $[t, t'] \in \mathrm{E}_f$ only if t' is the $\leq$-successor of t. In particular, $a \in \mathrm{E}_f$ and $y_1 < x_2$ implies $y_1 \leq x_1$. On the other hand, $b \in \mathrm{E}_f$ and $a \neq b$, so $y_1 < x_1$ and so $y_2 \leq x_1$, which means $b \sqsubseteq\!\!\cdot\ a$. Likewise $x_1 < y_2$ implies $a \sqsubseteq\!\!\cdot\ b$.

Furthermore, at least one of $y_1 < x_2$ or $x_1 < y_2$ holds, since otherwise $x_2 \leq y_1$ and $y_2 \leq x_1$ implies $x_1 = x_2$, which is a contradiction. Conversely, if we have both $y_1 < x_2$ and $x_1 < y_2$, then $a = b$, which is also a contradiction. Therefore exactly one of $a \sqsubseteq\!\!\cdot\ b$ or $b \sqsubseteq\!\!\cdot\ a$ holds.

We have thus established that $\sqsubseteq\!\!\cdot$ is connex and antisymmetric, since only the first case allowed $a \sqsubseteq\!\!\cdot\ b$ and $b \sqsubseteq\!\!\cdot\ a$ simultaneously.

Now let us suppose $a \sqsubseteq\!\!\cdot\ b$ and $b \sqsubseteq\!\!\cdot\ c$, but $a \not\sqsubseteq\!\!\cdot\ c$. We may assume a, b, c are all distinct, since otherwise we have an immediate contradiction from $S_0\,\Gamma_f$.

Since a, b, c are distinct, we must have $f(a) \preceq\!\!\cdot\ f(b) \preceq\!\!\cdot\ f(c)$ from the definition of the various $S_i\,\Gamma_f$ groups. In particular, if either of these inequalities is strict, then by transitivity of $\preceq\!\!\cdot$ we have $f(a) \prec\!\!\cdot\ f(c)$ and hence $a \sqsubseteq\!\!\cdot\ c$ from $S_1\,\Gamma_f$, which is a contradiction. Thus we may assume $f(a) = f(b) = f(c)$.

If $b \in \mathrm{E}_f$, then $b \sqsubseteq\!\!\cdot\ c$ implies $c \in \mathrm{E}_f$. Thus $b \in \mathrm{E}_f$ implies $a \in \mathrm{E}_f$ as well, since otherwise $a \sqsubseteq\!\!\cdot\ c$ from $S_2\,\Gamma_f$. Similarly, if $b \in \mathrm{V}_f$, then $a \in \mathrm{V}_f$ from $a \sqsubseteq\!\!\cdot\ b$ and so $c \in \mathrm{V}_f$ since otherwise $S_2\,\Gamma_f$ would imply $a \sqsubseteq\!\!\cdot\ c$. Thus we may now assume either $a, b, c \in \mathrm{V}_f$ or $a, b, c \in \mathrm{E}_f$.

If $a, b, c \in \mathrm{V}_f$, then Proposition 4.3.1 gives us $a \sqsubseteq\!\!\cdot\ c$. Therefore, we may now assume $a, b, c \in \mathrm{E}_f$. As we did for proving connexity and antisymmetry, we will now define some values for our convenience is analyzing the last cases, namely

$$
\begin{aligned}
a_1 &= \min_{\sqsubseteq\!\!\cdot}(\partial(a)), & b_1 &= \min_{\sqsubseteq\!\!\cdot}(\partial(b)), & c_1 &= \min_{\sqsubseteq\!\!\cdot}(\partial(c)),\\
a_2 &= \max_{\sqsubseteq\!\!\cdot}(\partial(a)), & b_2 &= \max_{\sqsubseteq\!\!\cdot}(\partial(b)), & c_2 &= \max_{\sqsubseteq\!\!\cdot}(\partial(c)),\\
x_1 &= \overrightarrow{a}, & y_1 &= \overrightarrow{b}, & z_1 &= \overrightarrow{c},\\
x_2 &= \overleftarrow{a}, & y_2 &= \overleftarrow{b}, & z_2 &= \overleftarrow{c}.
\end{aligned}
$$

If $a \sqsubseteq\cdot\ b$ and $b \sqsubseteq\cdot\ c$, then $f(a_2) \preceq\cdot\ f(b_2)$ and $f(b_2) \preceq\cdot\ f(c_2)$ from the definitions of $S_4\,\Gamma_f$, $S_5\,\Gamma_f$, and $S_6\,\Gamma_f$. Thus, by transitivity of $\preceq\cdot$, if either of these inequalities is strict, $f(a_2) \prec\cdot\ f(c_2)$ and so $a \sqsubseteq\cdot\ c$ from $S_4\,\Gamma_f$. Thus we see $f(a_2) = f(b_2) = f(c_2)$.

From $S_5\,\Gamma_f$ and $S_6\,\Gamma_f$, we must have $f(a_1) \preceq\cdot\ f(b_1) \preceq\cdot\ f(c_1)$. Again, the transitivity of $\preceq\cdot$ and the definition of $S_5\,\Gamma_f$ imply $f(a_1) = f(b_1) = f(c_1)$ since $a \not\sqsubseteq\cdot\ c$.

This leaves us with $x_2 \leq y_1$ and $y_2 \leq z_1$, but the transitivity of $\leq$ and the fact that $y_1 \leq y_2$ implies $x_2 \leq z_1$ and thus $a \sqsubseteq\cdot\ c$ from $S_6\,\Gamma_f$. Therefore $\sqsubseteq\cdot$ must be transitive. □

While $\sqsubseteq\cdot$ does not provide a total order on all of Γ_f, we will later show that $\sqsubseteq\cdot$ is a total order for certain larger subsets of Γ_f than just $\mathrm{V}_f \cup \mathrm{E}_f$. Before we expand our view, though, we will first look back at the partial order $\sqsubset$ we previously defined, and show that $\sqsubseteq\cdot$ also extends $\sqsubset$.

Proposition 4.3.3. *$\sqsubseteq\cdot$ extends $\sqsubseteq$ to a total order on $\mathrm{V}_f \cup \mathrm{E}_f$, and more generally $a \sqsubseteq b$ implies $a \sqsubseteq\cdot\ b$ for all $a, b \in \Gamma_f$.*

Proof. This follows almost immediately from comparing the definition of $a \sqsubseteq b$ and $a \sqsubseteq\cdot\ b$, particularly the sets $S_0\,\Gamma_f$, $S_1\,\Gamma_f$, $S_2\,\Gamma_f$, and $S_3\,\Gamma_f$. However, we must remember that if $f(a) \prec f(b)$ then $f(a) \prec\cdot\ f(b)$ since $\prec\cdot$ is an extension of $\prec$. □

We close this section with another stability result.

Lemma 4.3.4. *If g is a Scott-adjustment of f with injective Scott-adjuster ϕ, then for all $a, b \in \Gamma_f$, the following hold.*

4.3.4.1 $a \sqsubseteq_f b$ implies $a \sqsubseteq\cdot_g b$. That is, $\sqsubseteq\cdot_g$ extends $\sqsubseteq_f$.

4.3.4.2 $a \sqsubseteq_g b$ implies $a \sqsubseteq\cdot_f b$. That is, $\sqsubseteq\cdot_f$ extends $\sqsubseteq_g$.

4.3.4.3 If $f(x) \preceq\cdot\ f(y)$ implies $g(x) \preceq\cdot\ g(y)$ for all $x, y \in \mathrm{V}_f$, or if $g(x) \preceq\cdot\ g(y)$ implies $f(x) \preceq\cdot\ f(y)$ for all $x, y \in \mathrm{V}_g$, then $\sqsubseteq\cdot_f$ and $\sqsubseteq\cdot_g$ are equivalent, that is $a \sqsubseteq\cdot_f b$ if and only if $a \sqsubseteq\cdot_g b$.

Proof. Conclusions 4.3.4.1 and 4.3.4.2 follow directly from Proposition 4.3.3 and Lemma 4.2.3.

For Conclusion 4.3.4.3, first suppose by way of contradiction that there exists $a, b \in \Gamma_f$ for which the claim is false. Clearly if $a = b$, then $a \sqsubseteq_f b$ and $a \sqsubseteq_g b$, so 4.3.4.3 holds trivially. Thus $a \neq b$.

For a moment, we restrict ourselves to the case where $a, b \in \mathrm{V}_f$. Then since $\sqsubset_f$ and $\sqsubset_g$ are total orders on V_f, we may assume without loss of generality that $a \sqsubset_f b$ and $b \sqsubset_g a$. If $f(a) \neq f(b)$, then we have $f(a) \prec f(b)$; and since ϕ is injective on the image of f, $g(a) \neq g(b)$ so $g(b) \prec g(a)$. However, this contradicts the hypothesis of 4.3.4.3, so we know $f(a) = f(b)$ and, by the injectivity of ϕ, $g(a) = g(b)$.

Now with $a, b \in \mathrm{V}_f$, then $a \sqsubset_f b$ implies $a < b$, from $S_3\,\Gamma_f$, and hence $a \sqsubset_g b$, a contradiction. Thus we see that the claim holds for vertices.

Now we may assume that one of a or b is in P_f. Suppose $f(a) \neq f(b)$, and so $g(a) \neq g(b)$ by the injectivity of ϕ. Since $\prec$ is a total order on $\mathbb{R}^p$, then without loss of generality we must have $f(a) \prec f(b)$ and $g(b) \prec g(a)$ since we assumed our conclusion was false. However, this contradicts our hypothesis that $f(a) \prec f(b)$ implies $g(a) \prec g(b)$ or $g(a) \prec g(b)$ implies $f(a) \prec f(b)$. Therefore we see that $f(a) = f(b)$ and $g(a) = g(b)$.

Now we easily see that if one of a or b is in V_f, then we get a contradiction from $S_2\,\Gamma_f$, and hence $a, b \in \mathrm{P}_f$, with $f(a) = f(b)$ and $g(a) = g(b)$. However, the remaining sets $S_4\,\Gamma_f$, $S_5\,\Gamma_f$, and $S_6\,\Gamma_f$ only depend on the behavior of $\sqsubset$ for vertices, which we have already established in equivalent for f and g. □

Chapter 5. Coordinated Persistent Homology

As with multi-dimensional persistent homology, we wish to describe time-series using the homology of a multi-filtration. In particular, we wish to identify how the homology changes between different levels of the multi-filtration. In Chapter 4 we first encoded a multi-filtration as a labeled linear graph and then endowed it with a total order. Now we must identify when new homology elements are created and when old homology elements become trivial. The encoding as a labeled linear graph structure enables us to identify new homology elements from V_f. The FindDeaths algorithm described in this chapter identifies when old homology elements become trivial.

As in previous chapters, Γ_f is a given labeled linear graph structure on T, though here we will assume that T is in fact a finite subset of $\mathbb{R}$ so that the algorithm can actually terminate. We will continue to use $\preceq$ to denote the product order on $\mathbb{R}^p$ described in Section 2.2, $\preceq\cdot$ is a given total order on $\mathbb{R}^p$ that extends $\preceq$, $\sqsubseteq$ is a partial order on Γ_f induced by $\preceq$ and f as constructed in Section 4.2, and $\sqsubseteq\cdot$ is a relation on Γ_f induced by $\preceq\cdot$ and which extends $\sqsubseteq$ to a total order on $\mathrm{V}_f \cup \mathrm{E}_f$, as described in Section 4.3.

Given a vertex v, we will determine the parameters values at which the component of Γ_f generated by v first merges with a component generated by some other vertex w with $w \sqsubset\cdot v$, recorded by the path from w to v. This pairing of vertices and sets of paths is the *CPH* of the given multifiltration and total order. By way of comparison, persistent homology is the pairing of vertices with a single path of this type.

5.1 The FindDeaths Algorithm

Let **OrderedEdgeList** be the totally-ordered list of edges using the order $\sqsubseteq\cdot$. We will recursively define a new function $\mathcal{D}_f$ that sends elements of V_f to lists of paths. In describing the algorithm that defines $\mathcal{D}_f$, we will use $a \leftarrow b$ to mean that the variable a is assigned the value of b. We will use OrderedEdgeList$[i]$ to denote the ith element of OrderedEdgeList,

where indexing begins at 0. The actual algorithm is provided in Algorithm 2.

Algorithm 2 FindDeaths Algorithm

while OrderedEdgeList $\neq \emptyset$ **do**
  curEdge $\leftarrow$ OrderedEdgeList[0]
  Delete OrderedEdgeList[0] from OrderedEdgeList.
  target $\leftarrow \max_{\sqsubseteq \cdot}$(curEdge).
  while $\mathcal{D}_f$[target] $\neq \emptyset$ **do**
    preEdge $\leftarrow$ the 0th entry of $\mathcal{D}_f$[target].
    chord $\leftarrow$ curEdge $\sqcup$ preEdge.
    if preEdge $\sqsubseteq$ curEdge or f(preEdge) = f(curEdge) **then**
      curEdge $\leftarrow$ chord and target $\leftarrow \max_{\sqsubseteq \cdot}$(curEdge).
    else
      $\mathcal{D}_f$[target] $\leftarrow$ [preEdge, curEdge].
      OrderedEdgeList $\leftarrow$ OrderedEdgeList $\cup$ {chord}.
      curEdge $\leftarrow$ OrderedEdgeList[0]
      Delete OrderedEdgeList[0] from OrderedEdgeList.
      target $\leftarrow \max_{\sqsubseteq \cdot}$(curEdge).
    end if
  end while
  $\mathcal{D}_f$[target] $\leftarrow$ [curEdge].
end while

Let us study the algorithm by considering it at various points of execution, starting at the beginning. If E_f is empty, then of course the algorithm does nothing. Otherwise, we select the smallest element of E_f. Since we have not yet assigned any values for $\mathcal{D}_f$, we get $\mathcal{D}_f$[target] = [curEdge]. Note that OrderedEdgeList has decreased in size by 1 since nothing has been added to it.

We continue in the same fashion until we hit an edge $a = [r, r']$ that has the same target vertex as a previous edge $b = [s, s']$. In particular, we know that $b \sqsubseteq \cdot\ a$ and either $r = s'$ or $s = r'$. Let us consider from a homological point of view what is happening at that point.

Suppose $f(b) \preceq f(a)$. Since b has already entered the multi-filtration, $s' - s$ is already a trivial element of the homology and we have now made $r' - r$ trivial. However, if $r = s'$, then we see that $r' - s$ has also been made trivial; while $r' = s$ implies that $s' - r$ is now trivial. In either case, this is the boundary of the chord $a \sqcup b$. Therefore, in listing the edges

or paths that make certain homology elements trivial, we may use b and $a \sqcup b$ instead of b and a. This is carried out by replacing the first curEdge with the chord.

On the other hand, suppose $f(b) \not\preceq f(a)$. Since $f(a) \preceq f(b)$ implies $a \sqsubseteq\cdot\ b$, we see that $f(a)$ and $f(b)$ are actually incomparable by $\preceq$. This means there are levels of the multi-filtration at which a alone is present, levels at which b alone is present, and levels at which both a and b are present. Therefore, our list of which paths trivialize homology elements must include a, b and $a \sqcup b$ separately. This is carried out by adding the chord as a distinct element.

Observe that once a chord is formed, the vertex common to the edges involved can never be visited again since each vertex belongs to at most two edges. Therefore $\mathcal{D}_f[\text{target}]$ will never consist of more than two paths. On the other hand, because the algorithm runs through all edges and the underlying complex is connected, there will be exactly one vertex for which $\mathcal{D}_f[\text{target}]$ is empty. This represents the $\sqsubseteq\cdot$-minimum vertex, which of necessity will also be a $\sqsubseteq$-minimal vertex.

Theorem 5.1.1. *At each stage of the FindDeaths algorithm, $\sqsubseteq\cdot$ restricted to OrderedEdgeList is a total order on OrderedEdgeList.*

Proof. By Proposition 4.3.2, we know the statement is true when the algorithm initiates. Furthermore, removing an element from OrderedEdgeList cannot change $\mathcal{T}$ and neither can lines that do not affect OrderedEdgeList. Therefore $\mathcal{T}$ holds at least until Line 12 is first executed. Let n be the number of times that Line 12 has been executed and let $\mathcal{T}(n)$ be the statement that $\sqsubseteq\cdot$ restricted to OrderedEdgeList is a total order on OrderedEdgeList when Line 12 has been executed n times.

If $n = 1$, then chord $= a \sqcup b$ for some pairs of edges $a = [r, r']$ and $b = [s, s']$ such that $r' = s$, $r \sqsubseteq\cdot\ r'$, and $s' \sqsubseteq\cdot\ s$. Furthermore, because of the conditional on Line 8, $f(a) \neq f(b)$, $a \not\sqsubseteq b$ and $b \not\sqsubseteq a$. Therefore, $f(a) \prec f(a \sqcup b)$ and $f(b) \prec f(a \sqcup b)$, which also implies that $a \sqsubseteq\cdot\ a \sqcup b$ and $b \sqsubseteq\cdot\ a \sqcup b$. This assures us that $a \sqcup b$ did not need to be considered before a or b.

Since a and b were both previously removed from OrderedEdgeList, and this is the first time a path has been added, there are no paths in OrderedEdgeList with r' as an endpoint. Therefore, given $c = [t, t'] \in$ OrderedEdgeList, we know that exactly one of $c \sqsubset\cdot\ a \sqcup b$ or $a \sqcup b \sqsubset\cdot\ c$ holds, since otherwise we obtain a contradiction as follows.

Suppose $c \not\sqsubset\cdot\ a \sqcup b$ and $a \sqcup b \not\sqsubset\cdot\ c$. Since $c, a \sqcup b \in \mathrm{P}_f$, the definition of $S_1\,\Gamma_f$ implies $f(a \sqcup b) = f(c)$. Likewise, the definition of $S_4\,\Gamma_f$ implies $f\big(\max_{\sqsubseteq\cdot_f}(a)\big) = f\big(\max_{\sqsubseteq\cdot_f}(b)\big)$. In turn, this means that the definition of $S_5\,\Gamma_f$ implies $f\big(\min_{\sqsubseteq\cdot_f}(a)\big) = f\big(\min_{\sqsubseteq\cdot_f}(b)\big)$. However, we also know that

$$\overrightarrow{a \sqcup b} = r \quad \overleftarrow{a \sqcup b} = s' \quad \overrightarrow{c} = t \quad \overleftarrow{c} = t'.$$

From $S_6\,\Gamma_f$, we must conclude that $r < t'$ and $t < s'$. However, t and t' are consecutive, so $t' \leq s'$. Furthermore, s and s' were consecutive, so $t' \leq s$. Finally, r and $r' = s$ were consecutive, so $t' \leq r$, a contradiction.

Thus we see that the insertion of $a \sqcup b$ into OrderedEdgeList preserves the total order by $\sqsubseteq\cdot$, so $\mathcal{T}(1)$ is true.

Note further that each time Line 12 is executed, the paths making up chord were removed from OrderedEdgeList before chord is added. Thus OrderedEdgeList always consists of paths such that no two paths overlap at more than one vertex, when two paths do overlap at a vertex it is always an endpoint of the paths, and any given vertex is an endpoint of at most two paths in OrderedEdgeList.

Now suppose that $\mathcal{T}(k-1)$ holds and that Line 12 will be executed at least k times. Consider the value of chord at $n = k$. Now chord $= a \sqcup b$ where $a = [u, v]$ and $b = [v, w]$ are paths and not necessarily edges. Nevertheless, because of Line 8, we still conclude that $f(a) \prec f(a \sqcup b)$ and $f(b) \prec f(a \sqcup b)$. We also know that a and b were both previously removed from OrderedEdgeList and that no other paths contain any of the vertices strictly between u and w. Therefore, the same argument used for $n = 1$ shows that given any element $c = [t, t'] \in$ OrderedEdgeList, exactly one of $c \sqsubseteq\cdot\ a \sqcup b$ or $a \sqcup b \sqsubseteq\cdot\ c$ holds, so $\mathcal{T}(k)$ is true.

The theorem follows by induction. □

5.2 Properties of $\mathcal{D}_f$

A comparison of FindDeaths algorithm with those given for standard persistent homology, as in [1] and [4], yields an obvious connection: if $\preceq$ is also a total order on the image of f, the FindDeaths algorithm calculates the persistent homology birth-death pairs in each parameter simultaneously. This may happen, for instance, if the parameters are correlated.

On the other hand, if $\preceq$ is not a total order on the image of f, the output of FindDeaths may depend on which extension of $\preceq$ is chosen. For example, consider $f : \{0, 1, 2\} \to \mathbb{R}^3$ given by $f(0) = (1, 0, 0)$, $f(1) = (1, 1, 1)$ and $f(2) = (0, 1, 0)$. If we use the usual lexicographic order for $\preceq_\cdot$, then $\sqsubseteq_\cdot$ gives us $[1, 2] \sqsubset_{0,1}$ and $2 \sqsubset_\cdot 0 \sqsubset_\cdot 1$, so the FindDeaths algorithm proceeds as in Algorithm 3. On the other hand, using a different lexicographic order where we consider the coordinates in reverse, we have $[0, 1] \sqsubset_{\cdot 1,2}$ and $0 \sqsubset_\cdot 2 \sqsubset_\cdot 1$, so the FindDeaths algorithm proceeds as in Algorithm 4.

Algorithm 3 FindDeaths Example: Standard Lexicographical Order

1: OrderedEdgeList $\leftarrow [[1, 2], [0, 1]]$.
2: curEdge $\leftarrow [1, 2]$;
3: OrderedEdgeList $\leftarrow [[0, 1]]$.
4: target $\leftarrow 1$.
5: $\mathcal{D}_f[1] \leftarrow [[1, 2]]$.
6: curEdge $\leftarrow [0, 1]$.
7: OrderedEdgeList $\leftarrow []$.
8: target $\leftarrow 1$.
9: preEdge $\leftarrow [1, 2]$.
10: chord $\leftarrow [0, 1] \sqcup [1, 2] = [0, 2]$.
11: curEdge $\leftarrow [0, 2]$ and target $\leftarrow 0$.
12: $\mathcal{D}_f[0] \leftarrow [[0, 2]]$.

We now present a stability-like theorem for CPH, but we note that it differs from the stability theorems in Section 3.5 in two significant ways. First, we can compare CPH results directly, without the need for a bijection. Second, we use the idea of Scott adjustments to compare f and g instead of the difference $\|f - g\|_\infty$. This does not mean CPH does not admit a classical form of stability, but we do not give an argument for or against such a theorem at this time.

Algorithm 4 FindDeaths Example: Reverse Lexicographical Order

1: OrderedEdgeList $\leftarrow [[0,1],[1,2]]$.
2: curEdge $\leftarrow [0,1]$;
3: OrderedEdgeList $\leftarrow [[1,2]]$.
4: target $\leftarrow 1$.
5: $\mathcal{D}_f[1] \leftarrow [[0,1]]$.
6: curEdge $\leftarrow [1,2]$.
7: OrderedEdgeList $\leftarrow []$.
8: target $\leftarrow 1$.
9: preEdge $\leftarrow [0,1]$.
10: chord $\leftarrow [0,1] \sqcup [1,2] = [0,2]$.
11: curEdge $\leftarrow [0,2]$ and target $\leftarrow 2$.
12: $\mathcal{D}_f[2] \leftarrow [[0,2]]$.

Theorem 5.2.1 (Coordinated Persistent Homology is Stable with Respect to Scott Adjustment)**.** *Suppose g is a Scott-adjustment of f with injective Scott-adjuster ϕ. Furthermore, suppose that either $f(x) \preceq f(y)$ implies $g(x) \preceq g(y)$ for all $x, y \in \Gamma_f$ or that $g(x) \preceq g(y)$ implies $f(x) \preceq f(y)$ for all $x, y \in \Gamma_f$. Then $\mathcal{D}_f(a) = \mathcal{D}_g(a)$ for all $a \in \mathrm{V}_f$, and hence the coordinated persistent homology of Γ_f and Γ_g are equivalent.*

Proof. By Lemma 4.3.4, the hypotheses are sufficient to ensure $\sqsubseteq_f$ and $\sqsubseteq_g$ are equivalent. Furthermore, since ϕ is injective on the image of f we know $f(\text{preEdge}) = f(\text{curEdge})$ if and only if $g(\text{preEdge}) = g(\text{curEdge})$. Since the execution of the algorithm depends only on the labeled linear graph structure Γ_f and these preceding facts, we see that it will produce the same output whether f or g is used. □

This alternate kind of stability offers a certain advantage: because we are concerned with the orders induced by f and g rather than the actual difference between the values, we can still compare functions that are very different under the uniform metric. Furthermore, we obtain an exact correspondence between our lists, without the need for considering all possible bijections between the lists. Nevertheless, we are interested in determining whether CPH satisfies a more classic kind of stability as well, though we do not have any such result at this time.

It may help to consider a few examples of Scott-continuous functions to see how our stability result differs from the classical result. First, consider a simple rescaling of each dimension. This is in fact order preserving, but the difference between f and $\phi \circ f$ can be made arbitrarily large. A more interesting would be rounding coordinates up to the next integer, or more generally to any precision.

Chapter 6. Applying CPH to Seismic Data

We shall now seek to address the age-old question posed by countless mathematics students, "When will I ever use this?" In particular, we present a current problem in seismology and the results of a joint project with Sandia National Laboratories that used coordinated persistent homology (CPH) to address the problem. We shall focus specifically on the application of CPH to this problem and leave a detailed description of the data gathering and generation processes to the joint paper by Callor et. al. [7].

We wish to acknowledge here again the contributions of those who worked on the project. Erik Webb, Steven Vigil, and Jason Heath presented the problem to me and provided invaluable leadership throughout the project. The real world data we considered was compiled by Brian Young and Katherine Aur, while the synthetic data was produced by Christian Poppeliers. Greg Conner was a crucial asset in developing the actual computer code that implemented the CPH algorithm and in refining the algorithm to its current elegance.

6.1 The Seismic Classification Problem

We shall first define some basic terms to ensure consistency and then state the seismic classification problem. A *seismograph* is an instrument used to record the motion of the ground, and a *seismogram* is the record of this motion for a particular period of time. Because the ground may move in 3-dimensional space, we may record the motion along a set of three orthogonal directions and thus obtain a full description of the motion in a *3-component seismogram*. If the primary source of energy in a seismogram is a natural phenomenon, such as a sudden slip on a fault, then we call that an *earthquake*. This is meant to distinguish such events from similar ground motion caused by the denotation of a man-made device, and we will call these latter instances *explosions*. The *seismic classification problem* is to determine whether a given seismogram represents an earthquake or an explosion.

We have hope to solve the seismic classification problem due to the differences in how energy is generated by different sources. In particular, earthquakes are thought to generate energy over an area, while explosions behave more like point-sources, as in [14, 15]. In particular, an ideal explosion would result in motion only along the radial and vertical axes, while an earthquake should produce motion along all three axes. Recall that the radial axis is the direction from the seismograph to the actual event source, and the transverse axis is the direction perpendicular to both the radial and vertical axes.

Unfortunately, matters are complicated by the fact that energy is not produced uniformly in all directions from an earthquake source. In [14, 15], a certain kind of seismic source, called a double-couple source, is shown to have notably different patterns of energy dispersal than a point-source explosion. In particular, given a good spread of seismographs around an explosion, the seismograms should look quite similar; but the seismograms for an earthquake will vary considerably as the direction from the event changes.

6.2 Approach

Our goal is to correctly distinguish a given set of three-component seismograms that are known to be explosions from a larger set that includes known earthquakes. Because of time constraints, we were limited to seven real world and 240 synthetically generated seismograms. Because we are interested at present only with the application of CPH, we will refer the reader to [7] for a description of these datasets, though we again credit Katherine Aur and Brian Young with obtaining the real world data and Christian Poppeliers with generating the synthetic seismograms. For our current purpose, it suffices to consider our seismic data as a set of functions into $\mathbb{R}^3$.

In order to choose an appropriate total order on $\mathbb{R}^3$, we first apply a rotation to the data so that the first coordinate is motion along the radial axis, the second coordinate is along the transverse axis, and the third component is along the vertical axis. Now let $\preceq_\cdot$ be the lexicographical order on $\mathbb{R}^3$ and define Γ_f, $\sqsubseteq$, and $\sqsubseteq_\cdot$ as in Chapter 4. In particular, recall

that $\sqsubseteq$ is a partial order on the graph Γ_f induced by $\preceq$, and $\sqsubseteq\cdot$ is a total order on Γ_f that extends $\sqsubseteq$ and is induced by $\preceq\cdot$.

We can then calculate the coordinated persistent homology of f, which is a pairing of vertices of Γ_f with lists of paths in Γ_f. For this application, however, we further simplify our invariant. First, for any $v \in \mathrm{V}_f$ such that $\mathcal{D}_f(v)$ is an empty list, we redefine $\mathcal{D}_f(v)$ to be $\bigsqcup V$. Next, we define the *persistence vector* at v to be

$$\mathcal{P}(v) = \left(\bigwedge f(\mathcal{D}_f(v)) \right) - f(v).$$

Finally, we define the *characteristic vector* of a set of labeled linear graphs as the average of the persistence vectors of those graphs.

6.3 Motivation

Each of these additional steps taken to simplify our description of a seismogram serves a specific purpose, which we explain here before discussing our results.

First, the convention that empty lists are filled in as $\bigsqcup V$ is analogous to assigning a death value of ∞ to unpaired objects in persistent homology, but because our signals are always bounded, it should suffice to use the greatest observed values for a given signal, which come from $f(\bigsqcup V) = \bigvee f(V)$. The need to choose a finite value for this assignment becomes apparent later. However, we make no claim that this particular choice is optimal, merely that it is convenient and is the choice we made in obtaining our results.

The idea of the second step, defining the persistence vector, is to obtain a kind of multidimensional barcode, but with only the change in values being preserved. This was motivated by the idea that earthquakes and explosions would produce different ratios of energy along the three axes. Accordingly, we only needed to preserve the ratio of displacement, not actual position. The most surprising part of this step is probably the use of $\curlywedge$ to handle the case where $\mathcal{D}_f$ consists of multiple paths. However, this is actually motivated by mathematics, not

our application. By comparing the PH and CPH algorithms, one sees that $\curlywedge f(\mathcal{D}_f(v))$ gives the corresponding death values of v obtained by running PH on each dimension separately.

The third step of combining persistence vectors from multiple seismograms allows us to leverage the differences in energy distribution. If the energy source was a point source, we would expect multiple, evenly space seismographs to experience similar ground motion. However, for an earthquake, if we have a good distribution of instruments, we expect some seismograms to actually look quite similar to an explosion, while most of the seismograms should be different [14, 15]. Thus averaging our results allows us to offset the seismograms that were poor discriminators.

6.4 Results

Since the final output of our analysis of the seismograms is a vector in $\mathbb{R}^3$ with non-negative entries, we can visualize these using spherical coordinates. The accompanying figures have been reproduced and edited from [7]. In order to account for differences in magnitude between events and to allow for a 2-dimensional figure, we plot the azimuth and elevation angles of the characteristic vectors, where the azimuth angle is measured from the radial axis and the elevation angle is measured from the radial-transverse plane.

We shall first consider the results of our analysis on the synthetic data generated by Christian Poppeliers. This set consists of 240 seismograms that represent 12 different seismographs recording 10 simulated earthquakes and 10 simulated explosions. In Figure 6.1, we group the seismograms according to which event they recorded and then calculate the corresponding characteristic vector for each group. From the figure we see that explosions strongly clustered away from the earthquakes. As expected, the explosions typically have a lower ratio of transverse to radial energy than the earthquakes and this ratio is fairly consistent between explosions. This is seen in the horizontal clustering of explosions in the figure. We also see a clear vertical division between the explosions and the earthquakes, which corresponds to a greater ratio of vertical motion vs horizontal motion. In fact the

vertical distance between the closest explosion-earthquake pair is nearly the same as the vertical distance between the furthest pair of earthquakes.

We must also note from Figure 6.2 the importance of considering multiple seismograms for a given event. Though the majority of explosions continue to plot some ways above $\pi/4$, and the majority of earthquakes plot at or below $\pi/4$, there are many exceptions. However, by comparing both plots we conclude that for a given event, most seismograms for the event will correctly discriminate.

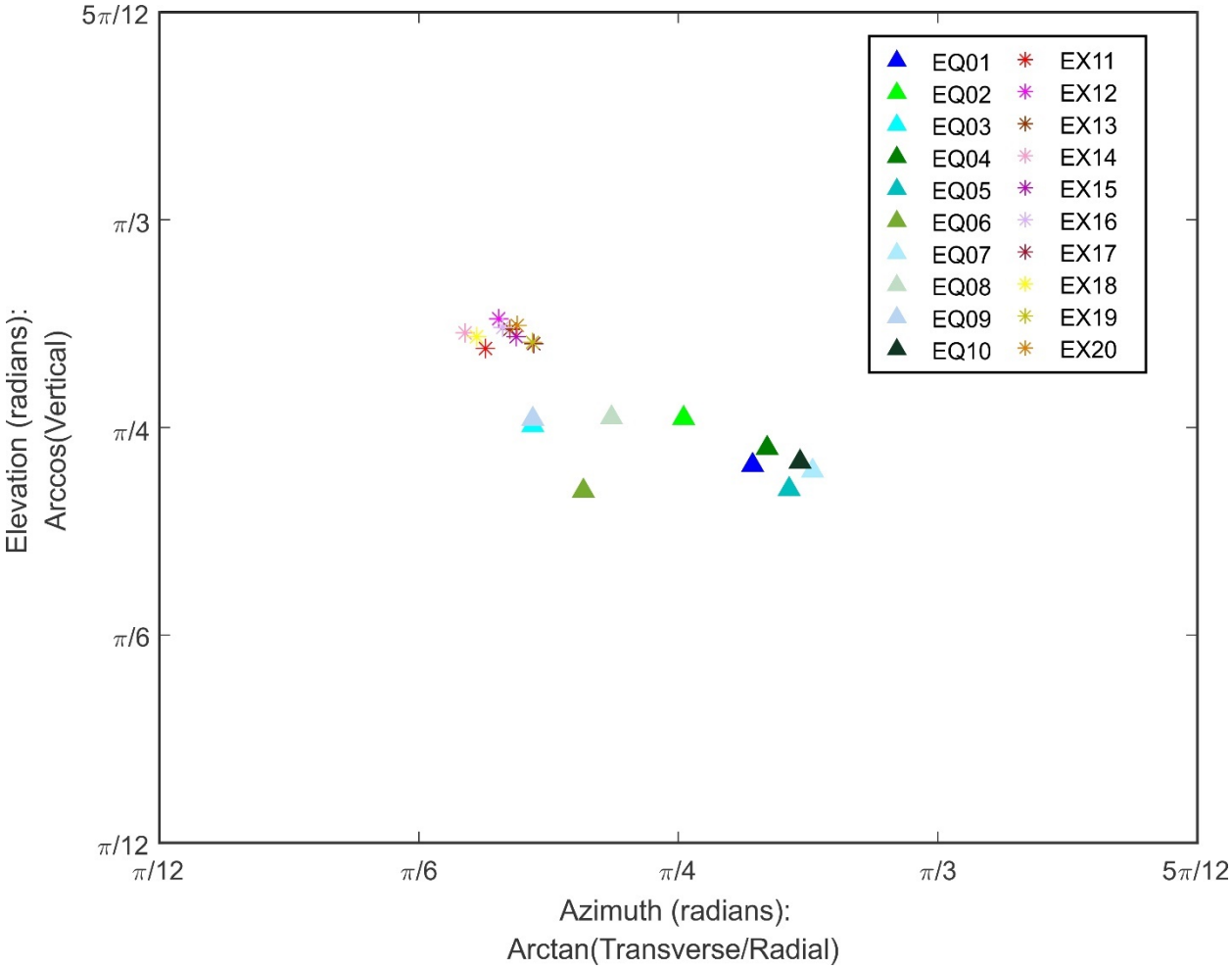

Figure 6.1: Azimuth and elevation plot of the grouped characteristic vectors for the synthetic seismograms

We now consider the few real world datasets that could be released to us for study, which were compiled by Katherine Aur and Brian Young. In order to keep the current manuscript

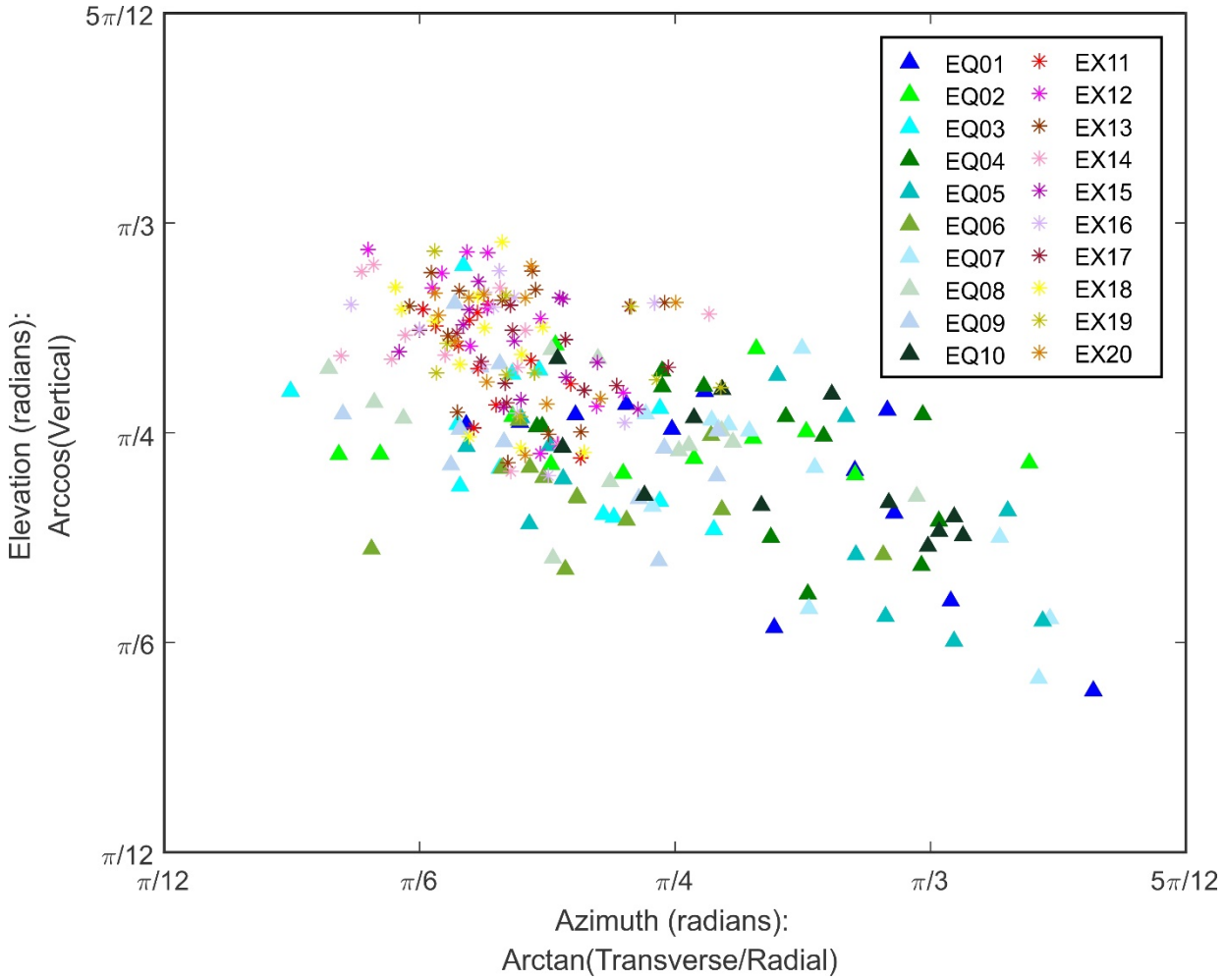

Figure 6.2: Azimuth and elevation plot of the characteristic vectors for the synthetic seismograms

clear of any possible sensitive information, however, we have sanitized the actual details about these datasets, though they can be found in [7] when it is published. It suffices here to state the these events were chosen because of their similarities to one another. Though we have already established the importance of considering multiple stations, technical limitations forced us to consider only one seismogram per event for this dataset. Nevertheless, as we see in Figure 6.3, the characteristic vector for the individual seismograms was sufficient to separate the explosions from the earthquakes by a horizontal band. Though the dividing band is lower for this dataset than for the synthetics, we suspect the difference is simply the presence of real-world background noise lowering all of the signals.

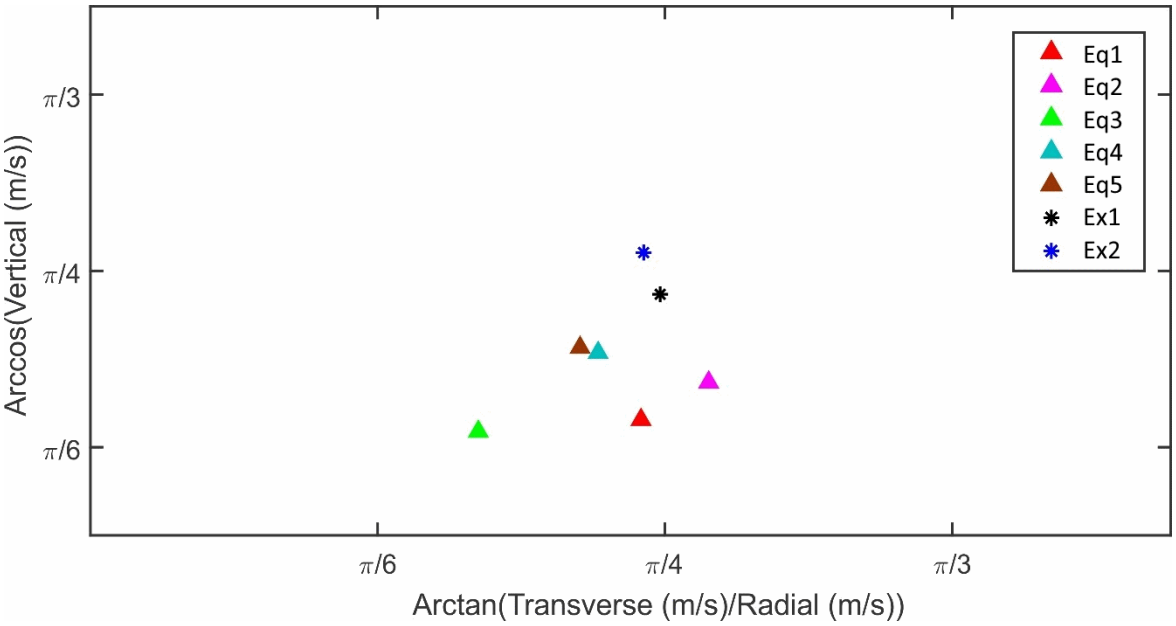

Figure 6.3: Azimuth and elevation plot of the characteristic vectors for the real seismograms

6.5 Additional Observations

We close this chapter with some observations about the characteristic vectors for our synthetic dataset and a corresponding conjecture for the overall theory of coordinated persistent homology. After obtaining the positive results discussed in the the previous section, a question arose about how much noise would affect the characteristic vector. At the time we had

no version of a stability theorem. A few people, and in particular Jason Heath and Greg Conner, suggested running simulations to see what kind of stability was present experimentally.

Due to time constraints, we were only able to consider simple noise models. We ultimately chose to look at the effect of adding pink noise because it was suggested that this would be a good approximation of typical seismic noise and still be simple enough to work with. We then used a custom Matlab script to add pink noise to our synthetic data at various signal-to-noise ratios. In particular, we ran simulations where the energy present in the original signal and the energy present in the added noise were at ratios of 16:1, 8:1, 4:1, 2:1, 1:1, 1:2, 1:4, 1:10, 1:100, 1:1000, 1:10000, 1:100000, and 1:1000000. That is to say, in our last simulation, the energy present in the added noise was 10^6 times the energy originally present in the signal. At each simulation level, we generated a random pink noise signal for each of the original 240 synthetic seismograms and ran a full analysis on the new noisy signals. We then repeated this process 10 times for each simulation level.

In order to compare different analyses, we computed the difference in elevation angle between the lowest characteristic vector for an explosion event and the greatest characteristic vector for an earthquake event. We plot the results of this inquiry in Figure 6.4 as a box-and-whisker plot. As we can see, in every instance CPH was able to discriminate explosions from earthquakes with a gap of about $3\pi/64$.

Surprisingly, the variation within a single signal-to-noise ratio level was greater than the variation between different levels! This suggests that CPH is resistant to the effect of adding pink noise, at least for discriminating earthquakes and explosions. By contrast, the stability theorems for PH and MPH suggest that they would be unlikely to resist noise at ratios below 1:1. At levels like 1:1000, the noise is numerically far more important than the original signal, and so those classic theories should start to extract information about the so-called noise and override the signal as a numerical recording error.

Unfortunately, we have yet to prove that CPH resists pink noise, or perhaps even other kinds of noise. However, this question and the question of classical stability provide a good course forward to expand the theory.

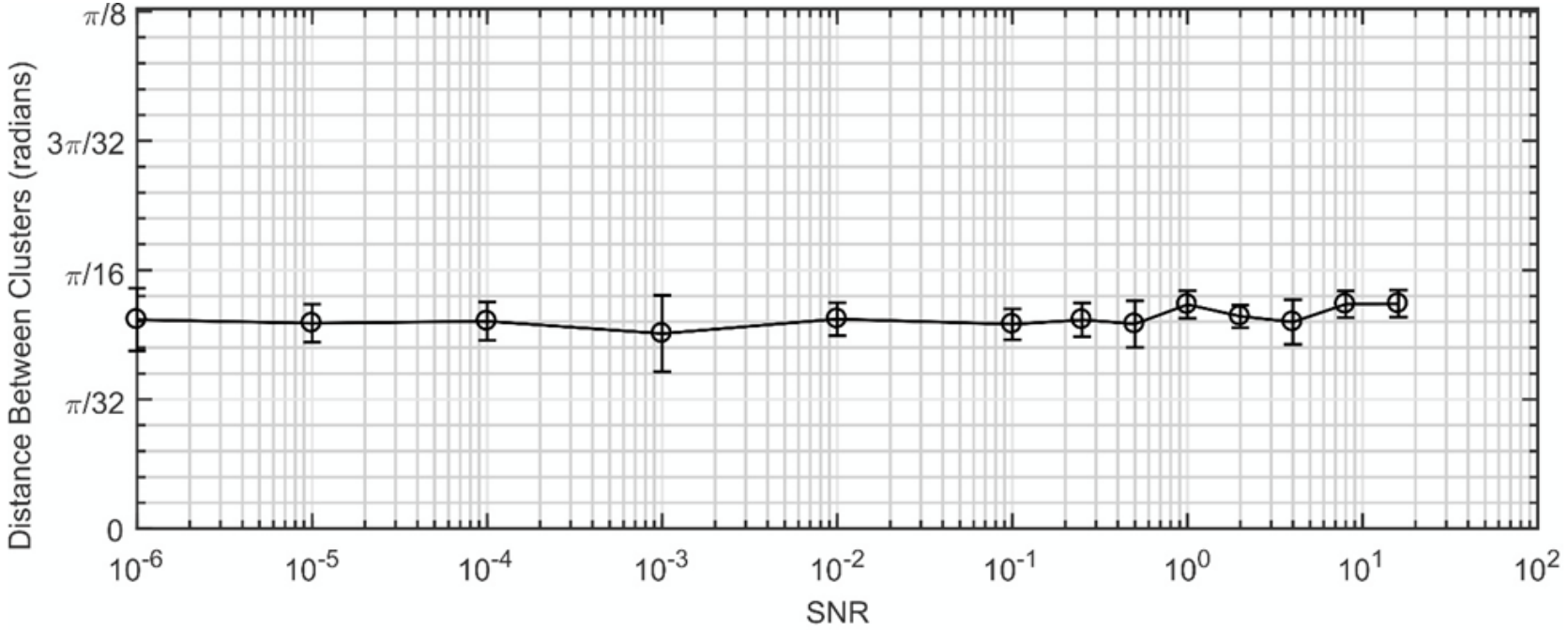

Figure 6.4: Comparison of the effect of pink noise on CPH

Bibliography

[1] Herbert Edelsbrunner, David Letscher, Afra Zomorodian. Topological Persistence and Simplification *Discrete & Computational Geometry.* 2002; 28:511-533.

[2] Gunnar Carlsson, Afra Zomorodian. The Theory of Multidimensional Persistence *Discrete & Computational Geometry.* 2009; 42:71-93.

[3] Vanessa Robins. *Computational topology at multiple resolutions: foundations and applications to fractals and dynamics.* PhD thesis University of Colorado 2002.

[4] Gunnar Carlsson, Afra Zomorodian, Anne Collins, Leonidas J. Guibas. Persistence Barcodes For Shapes *International Journal of Shape Modeling.* 2005; 11:149-187.

[5] Gunnar Carlsson, Gurjeet Singh, Afra Zomorodian. Computing Multidimensional Persistence *Journal of Computational Geometry.* 2009; 1:72-100.

[6] Andrea Cerri, Barbara Di Fabio, Massimo Ferri, Patrizio Frosini, Claudia Landi. Multidimensional persistent homology is stable 2009.

[7] Nickolas B. Callor, Jason E. Heath, Gregory R. Conner, Brian Young, Katherine A. Aur, Christian Poppeliers. Classifying Seismic Events via Characteristic Vectors of Concurrent Persistent Homology preprint Sandia National Labs 2018.

[8] Dana S Scott. Continuous lattices in *Toposes, Algebraic Geometry and Logic* (F. W. Lawvere. , ed.)(Berlin, Heidelberg):97-136Springer Berlin Heidelberg 1972.

[9] Robert Ghrist. Barcodes: The persistent topology of data *Bulletin of the American Mathematical Society.* 2007; 45:61-76.

[10] Erik Carlsson, Gunnar Carlsson, Vin De Silva. An Algebraic Topological Method For Feature Identification *International Journal of Computational Geometry & Applications.* 2006; 16:291–314.

[11] Herbert Edelsbrunner, John Harer. Persistent homology—a survey *Surveys on Discrete and Computational Geometry Contemporary Mathematics.* 2008:257–282.

[12] Dmitriy Morozov. Persistence algorithm takes cubic time in worst case *BioGeometry News, Dept. Comput. Sci., Duke Univ.* 2005.

[13] David Cohen-Steiner, Herbert Edelsbrunner, John Harer. Stability of Persistence Diagrams *Discrete & Computational Geometry.* 2006; 37:103–120.

[14] Keiiti Aki, Paul G. Richards. *Quantitative seismology theory and methods*; 1. W. H. Freeman and Company1 ed. 1980.

[15] Peter M. Shearer. *Introduction to seismology.* Cambridge University Press2 ed. 2011.

Chapter 7. Appendix: Stock Market Example

7.1 The Original Data

Suppose we have collected stock market values for a particular stock over an 11-day period, as listed in Figure 7.1. The *Open* value for a date is the value of the stock when trading opened in a particular market, while the *Close* value is the value when trading closed for the same market. Likewise, the *High* and *Low* values are the maximum and minimum values for the stock while trading was open for that market. Because the value of a stock can change while a particular market is closed, the Close value for day t is not necessarily the Open value for day $t + 1$.

Date	**Open**	**High**	**Low**	**Close**
0	42.2	42.2	39.4	39.4
1	40.2	41.8	40.1	40.6
2	39.5	42	39.2	42
3	41	41.7	41	41.7
4	42	42	38.7	39.4
5	38.9	40.6	38.9	40.6
6	40.3	41.3	40.3	41.3
7	41.6	41.6	39.2	40
8	40.6	41.7	39.3	39.3
9	39.5	40.9	39.5	40.9
10	39.8	41.4	39.8	39.8

Figure 7.1: Sample Stock Market Data

7.2 Multifiltrations

We can visualize one multifiltration on the data by letting $X_{(\alpha,\beta)}$ consist of stocks whose High value was at least α and whose Low value was at most β and for pairs $\alpha \geq \gamma$, $\beta \leq \delta$, we have the inclusion map $\phi^{(\gamma,\delta)}_{(\alpha,\beta)}$. Figure 7.2 provides a visualization of this multifiltration, where lines are drawn between consecutive dates when both are present in a given $X_{(\alpha,\beta)}$, and we have inclusion maps going to the right or up from a given space. Note that the

top-right space $X_{(40.6,41)}$ contains the entire space since 40.6 is the minimum High value for these dates and 41 is the maximum Low value.

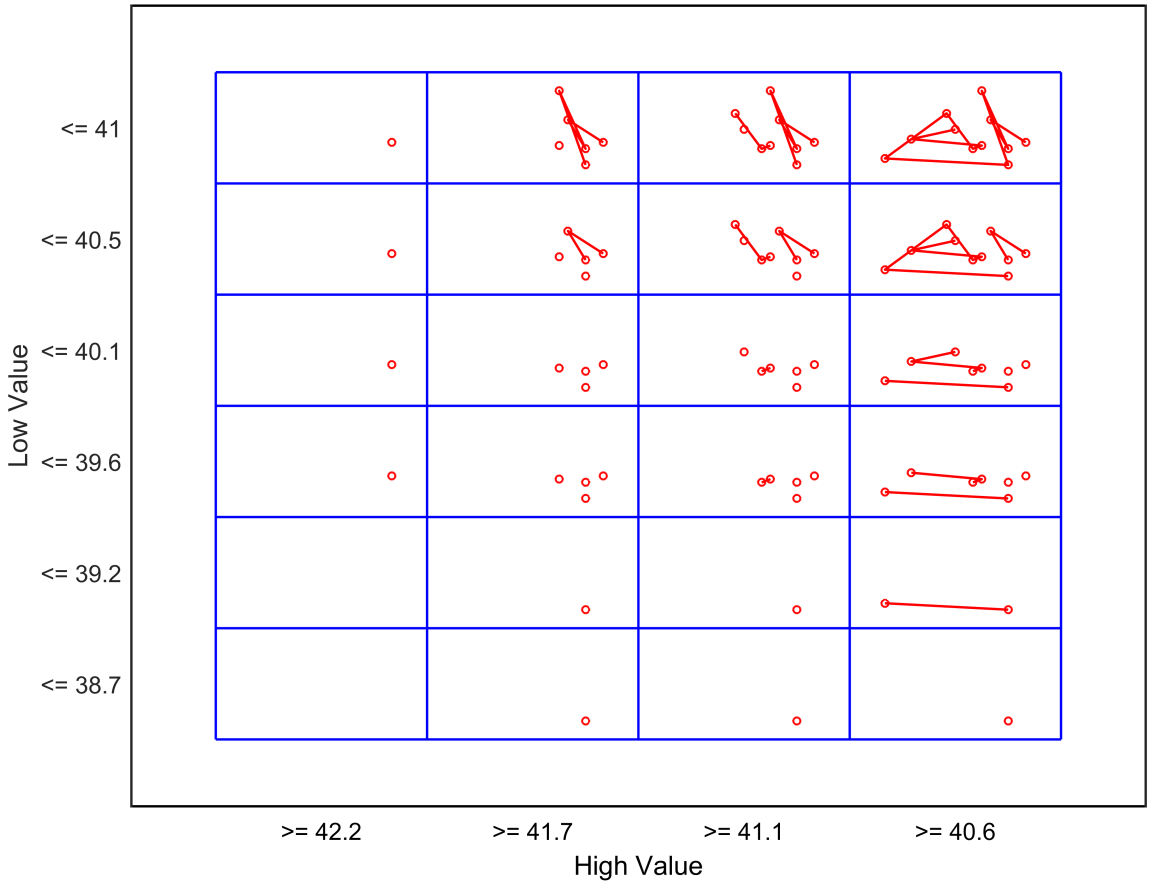

Figure 7.2: Multifiltration using High and Low Values

While the preceding multifiltration may be interesting, it may cause confusion to use a partial order on $\mathbb{R}^2$ that comes from a mix of $\geq$ on one coordinate and $\leq$ on the other. While this is not a problem for CPH, we will continue our example using the more natural multifiltration given by letting $X_{(\alpha,\beta,\gamma,\delta)}$ be the dates for which the Open value was at most α, the High value was at most β, the Low value was at most γ, and the Close value was at most δ. Then we have inclusion maps ϕ_u^v when $u \preceq v$ with the standard product order formed by considering $\leq$ on each coordinate. We will let $f(a) = (u, v, w, x)$, where u, v, w, and x are the Open, High, Low, and Close values of the stock on day a.

7.3 Orders

With the product order $\preceq$, we obtain the partial order $\sqsubseteq$ on our labelled linear graph as described in Section 4.2. In particular, $a \sqsubseteq b$ if $f(a) \preceq f(b)$. The Hasse diagram for this partial order is given in Figure 7.3, where an arrow from a to b means $a \sqsubseteq b$.

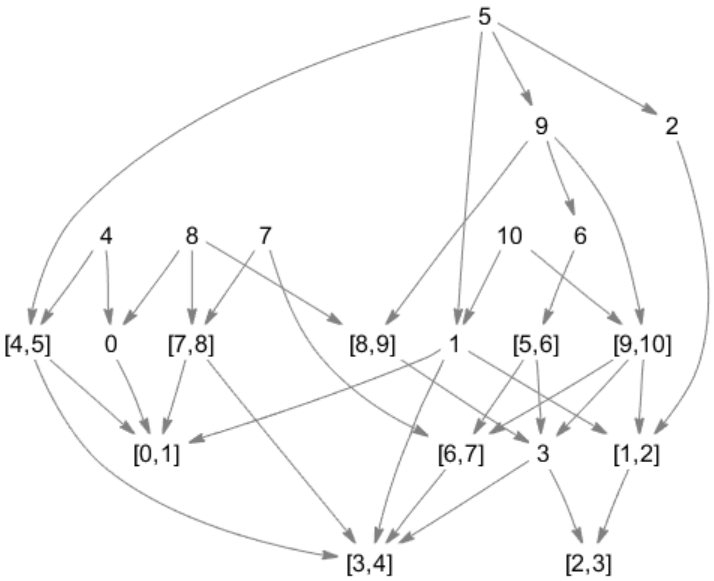

Figure 7.3: Hasse diagram for $\sqsubseteq$

We will use the lexicographic order on $\mathbb{R}^4$ for our total order $\preceq\cdot$ and we will follow the construction in Section 4.3 to obtain a total order $\sqsubseteq\cdot$ on our labeled linear graph. This gives us the following ordered lists of vertices and edges.

$$5 \sqsubset\cdot 9 \sqsubset\cdot 2 \sqsubset\cdot 10 \sqsubset\cdot 1 \sqsubset\cdot 6 \sqsubset\cdot 8 \sqsubset\cdot 3 \sqsubset\cdot 7 \sqsubset\cdot 4 \sqsubset\cdot 0$$

$$[9,10] \sqsubset\cdot [1,2] \sqsubset\cdot [5,6] \sqsubset\cdot [8,9] \sqsubset\cdot [2,3] \sqsubset\cdot [6,7] \sqsubset\cdot [7,8] \sqsubset\cdot [4,5] \sqsubset\cdot [3,4] \sqsubset\cdot [0,1]$$

7.4 Find Deaths

In order to illustrate the algorithm from Section 5.1, we represent **OrderedEdgeList** and the information regarding $\max_{\sqsubseteq\cdot}(\text{curEdge})$ by forming the table in Table 7.1a. As we see

from Table 7.1a, the first time we see an edge whose maximum endpoint has already been considered is at $[7, 8]$ and the comparison edge preEdge is $[6, 7]$. These two edges are incomparable with $\sqsubseteq$, so we add $[7, 8]$ to the list of death values for 7 and add the chord $[6, 7] \sqcup [7, 8] = [6, 8]$ to **OrderedEdgeList**. $6 \sqsubseteq\cdot\ 8$, so the maximum endpoint of $[6, 8]$ is 8. $f([6, 8]) = (41.6, 41.7, 40.3, 41.3)$, so $[6, 8]$ is inserted between $[7, 8]$ and $[4, 5]$, which means $[6, 8]$ is the next path we would consider. The new **OrderedEdgeList** with $\max_{\sqsubseteq\cdot}$ is given in Table 7.1b.

The maximum endpoint of $[6, 8]$ is 8 and the edge $[8, 9]$ also had 8 as its maximum endpoint. Since $f([6, 8]) = (41.6, 41.7, 40.3, 41.3)$ and $f([8, 9]) = (40.6, 41.7, 39.5, 40.9)$, $[8, 9] \sqsubseteq_{6,8}$ and so we form the chord $[6, 8] \sqcup [8, 9] = [6, 9]$ and replace $[6, 8]$ in **OrderedEdgeList**. However, the maximum endpoint of $[6, 9]$ is 6, which was also the maximum endpoint of $[5, 6]$. Since $f([6, 9]) = (41.6, 41.7, 40.3, 41.3)$ and $f([5, 6]) = (40.3, 41.3, 40.3, 41.3)$, $[5, 6] \sqsubseteq_{6,9}$ and so we form the chord $[5, 9]$ and replace $[6, 9]$ in **OrderedEdgeList**. The maximum endpoint of $[5, 9]$ is 9, which is not the maximum endpoint of any previous edge or path. We give **OrderedEdgeList** at this point in Table 7.1c.

The next edge in our list, $[4, 5]$, has a maximum endpoint that has not yet had a death value assigned, so we make the assignment and move to $[3, 4]$, which also has 4 as its maximum endpoint. $[4, 5] \sqsubseteq_{3,4}$, so we replace $[3, 4]$ with the chord $[3, 5]$. This new path has 3 as its maximum endpoint, which is also the maximum endpoint of $[2, 3]$. These edges are not $\sqsubseteq$-comparable, so insert the chord $[2, 5]$ in **OrderedEdgeList** and add $[3, 5]$ to the list of death values for 3. $f([2, 5]) = (42, 42, 41, 42)$ so $[2, 5]$ is inserted between $[3, 5]$ and $[0, 1]$. The maximum endpoint of $[2, 5]$ is 2, which is not the maximum endpoint of any previous edge or path, so we assign $[2, 5]$ as the death value for 2 and move on to our final edge $[0, 1]$. Table 7.1d gives **OrderedEdgeList** at this stage of the process. In fact, since the maximum endpoint of $[0, 1]$ is 0, which is not the maximum endpoint of any other edge or path, then we finish the algorithm by assigning $[0, 1]$ as the death value for 0 and so Table 7.1d is the final state of OrderedEdgeList.

Path	[9,10]	[1,2]	[5,6]	[8,9]	[2,3]	[6,7]	[7,8]	[4,5]	[3,4]	[0,1]
$\max_{\sqsubseteq \cdot}$	10	1	6	8	3	7	7	4	4	0

(a) The initial **OrderedEdgeList** together with $\max_{\sqsubseteq \cdot}$ for each path.

Path	[9,10]	[1,2]	[5,6]	[8,9]	[2,3]	[6,7]	[7,8]	[6,8]	[4,5]	[3,4]	[0,1]
$\max_{\sqsubseteq \cdot}$	10	1	6	8	3	7	7	8	4	4	0

(b) **OrderedEdgeList** and $\max_{\sqsubseteq \cdot}$ after considering the edge $[7, 8]$.

Path	[9,10]	[1,2]	[5,6]	[8,9]	[2,3]	[6,7]	[7,8]	[5,9]	[4,5]	[3,4]	[0,1]
$\max_{\sqsubseteq \cdot}$	10	1	6	8	3	7	7	9	4	4	0

(c) **OrderedEdgeList** and $\max_{\sqsubseteq \cdot}$ after considering the paths $[6, 8]$, $[6, 9]$, and $[5, 9]$.

Path	[9,10]	[1,2]	[5,6]	[8,9]	[2,3]	[6,7]	[7,8]	[5,9]	[4,5]	[3,4]	[2,5]	[0,1]
$\max_{\sqsubseteq \cdot}$	10	1	6	8	3	7	7	9	4	4	2	0

(d) **OrderedEdgeList** and $\max_{\sqsubseteq \cdot}$ after considering $[4, 5]$, $[3, 4]$, $[3, 5]$, and $[2, 5]$.

7.5 CPH and Persistence Vectors

Note that the table in Table 7.1d also gives us the list of death values if we look in the second row for our vertex. For instance, we see 10 is associated with $[9, 10]$, while 7 is associated with both $[6, 7]$ and $[7, 8]$. Only vertex 5 is missing from the table, and this is because it was assigned no death value. This follows from the fact that 5 was the $\sqsubseteq \cdot$-minimum vertex and therefore the component it represents is present in each level of the multifiltration once the parameter values are all sufficiently large.