2024

Persistent Relative Homology for Topological Data Analysis

Christian J. Lentz
Macalester College, cjlsoccer23@gmail.com

Persistent Relative Homology for Topological Data Analysis

Christian Lentz

Lori Ziegelmeier, Advisor

Greg Henselman-Petrusek, Reader

Yariana Diaz, Reader

Department of Mathematics, Statistics, and Computer Science

May, 2024

Abstract

A central problem in data-driven scientific inquiry is how to interpret structure in noisy, high-dimensional data. Topological data analysis (TDA) provides a solution via the language of persistent homology, which encodes features of interest as holes within a filtration of the data. The recently presented U-Match Decomposition places the standard persistence computation in a flexible form, allowing for straight-forward extensions of the algorithm to variations of persistent homology. We describe U-Match Decomposition in the context of persistent homology, and extend it to an algorithm for persistent relative homology, providing proofs for the correctness and stability of the presented algorithm.

Contents

1. Introduction

Persistent homology was first introduced as a means of topological simplification, where homology encodes the topological complexity of point cloud data and a scale filtration describes the persistence of features. With this came the introduction of *pair persistence* algorithms for subcomplexes of $\mathbb{S}^3$ over the field $\mathbb{Z}_2$, which was the first means of computing persistence [1], [2]. Following this were generalizations and optimizations of the algorithm. Some of these notable advances include an extension to complexes of arbitrary dimension over arbitrary fields [3], generating persistence pairs via matrix reduction [4], and a clever implementation of the standard matrix reduction algorithm using sparse matrix formats [5] or column-clearing approaches [6]. The persistence computation has cubic worst-case time complexity in the number of simplices, although the worst case is exceedingly rare, especially given optimized implementations.

With the introduction of persistence came the field of Topological Data Analysis (TDA), as well as techniques for understanding and summarizing persistence computations. In particular, these include the *persistence diagram* [1] and the *barcode* [7]. While the former encodes the birth-death interval of a feature with a point on or above the diagonal of the extended plane $\overline{\mathbb{R}}^2$, the latter explicitly encodes these intervals as bars sorted according to dimension. Intuitively, features with larger lifetime will require a larger scale parameter to eliminate the feature. The information contained in a barcode or persistence diagram can also be summarized by the *persistence module*, a sequence of R-modules with associated homomorphisms which induce inclusions on homology across dimension. The persistence module was first introduced in [3] as a means of elucidating the structure of persistent homology and has been a key tool used to provide results assuring the stability of persistence algorithms.

The abundance of literature and applications of persistent homology in recent years suggests a need to relate existing techniques of computation to variants such as *persistent relative homology* (PRH), which describes how the persistent homology of point cloud data depends on a subset of that data. Although there is existing work in this area, it is sparse, and without application. To this end, we provide an algorithm for PRH which is an extension of the U-Match Decomposition [8]. The rich structure of U-Match will allow this work to be straightforward and widely accessible.

2. Background

In this chapter we review the key definitions and computational tools needed for this text. Throughout, we will assume the reader is familiar with persistent homology. In any other case, we recommend [3], [9] and [10] to get started.

1 Persistent Relative Homology

Although the existing literature is sparse, this is not the first study of PRH in the theoretical or algorithmic sense. We start with definitions, notation and previous work relating to PRH.

1.1 Definitions and Notation

Much of this section is motivated by texts from Edelsbrunner and Harer [5] and Hatcher [11]. The definitions in this section will inform how we can make the U-Match Decomposition (see Chapter 2.2) compatible for PRH, and the notation established here will be carried throughout the rest of this text. We start by recalling key ideas and tools of persistent homology.

Let K be an abstract simplicial complex satisfying $dim(K) = n$. For each $p \in \{0, \dots, n\}$, define $C_p(K)$ as the **chain group** that describes the span of all linear combinations of **p-simplices** in K. Equivalently, this is a **chain vector space** with linear combinations using field coefficients. Rather intuitively, we call a linear combination of p-simplices a **p-chain**. Now consider the linear transformation $\partial_p : C_p(K) \to C_{p-1}(K)$ which maps the basis elements of $C_p(K)$ as follows

$$[v_0 v_1 \dots v_p] \mapsto \sum_{j=0}^{p} (-1)^j [v_0 v_1 \dots \hat{v_j} \dots v_p],$$

and can be extended linearly to non-basis elements of the vector space $C_p(K)$. The notation $\hat{v_j}$ denotes a single point of the p-simplex $[v_0 v_1 \dots \hat{v_j} \dots v_p]$ being removed. This is the (alternating) **boundary operator**, which maps a p-simplex in K to a $(p-1)$-chain in K which is its boundary.

Using these definitions, we may define a **chain complex** $\mathcal{C}_\bullet(K)$ to be a sequence of chain vector spaces over some simplicial complex K connected

by boundary operators which satisfy the property that $\partial_{p-1}\partial_p = 0$. This stipulation on the boundary operators of the chain complex give the intuitive idea that the boundary of any p-chain cannot have a boundary itself. The implications of this statement are rather important, as they imply that $\text{Im}(\partial_{p+1}) \subseteq \text{Ker}(\partial_p)$. Thus, any member of $\text{Ker}(\partial_p)$ is a p-chain with no boundary, which we refer to as a **p-cycle**. On the other hand, any member of $\text{Im}(\partial_{p+1})$ is a p-cycle which is also the boundary of some $(p+1)$-chain. We refer to such chains as **p-boundaries**. Figure 2.1 below gives a pictorial description of these ideas.

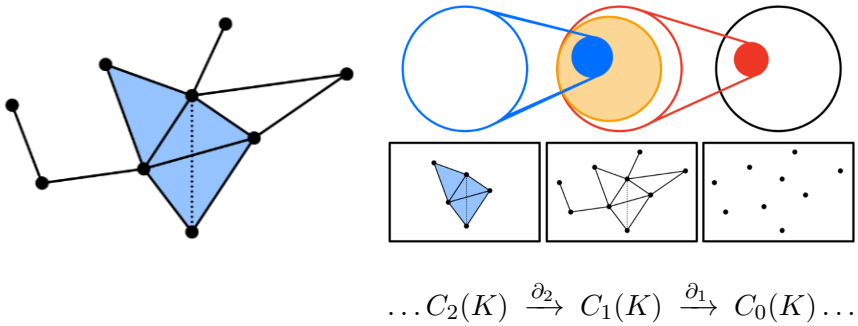


Figure 2.1 A simplicial complex (left) and its chain complex (right) with corresponding chain vector space bases. Note that $\text{Im}(\partial_2) \subseteq \text{Ker}(\partial_1)$.

With these ideas in place, we can define $H_p(K) = \text{Ker}(\partial_p)/\text{Im}(\partial_{p+1})$ as a quotient vector space describing the difference between all p-cycles in K and the p-cycles in K which are not also the boundary of a $(p+1)$-chain. Equivalently, this is a **homology vector space**, whereas the more general **homology group** is a quotient group. From this point on in the text, we will refer to this notion of homology as **absolute homology**.

Traditionally, persistent homology tracks absolute homological features through a scale filtration of a topological space. For a topological space X, a finite, N-level filtration on X is given by $F_1X \subseteq F_2X \subseteq \cdots \subseteq F_NX$, where $F_NX = X$. We denote such a filtration as $F_\bullet X$ and say that any σ born at F_tX has a **filtration value** $b_F(\sigma) = t$. Thus, $F_tX = \{\sigma \in X : b_F(\sigma) \leq t\}$. In words, $b_F(\sigma)$ denotes the **birth** of σ in the filtration $F_\bullet$. There are a number of standard computational methods for persistence, some of which we highlighted in the introduction of this text. For our purpose, we wish to extend these methods to a class of topological spaces known as **quotient spaces**.

Definition 1. *Suppose we have topological spaces X and A such that $A \subseteq X$. Then the quotient space is defined as*

$$X/A = (X \setminus A) \sqcup *$$

*where * is a single point. Any $U \subseteq X/A$ is open in the quotient topology if and only if $\pi^{-1}(U)$ is open in X where $\pi : X \to X/A$ is given by*:

$$\pi(x) = \begin{cases} x \ \textit{if} \ x \in X \setminus A \\ * \ \textit{if} \ x \in A \end{cases}$$

In other words, open sets in X are used to define the open sets in X/A.

At a high level, the quotient topology is similar to other quotient structures across pure mathematics, where by removing certain data or structure from the problem we obtain a new—and perhaps more informative—perspective, as seen in Figure 2.2. An equivalent definition of the quotient topology may define an equivalence relation $\sim$ on X where $Y = X/\sim$ is the set of equivalence classes of X under the the equivalence relation. In this formulation, the map $\pi : X \to X/\sim$ maps some $x \in X$ to its equivalence class $[x] \in X/\sim$. It is this point of view that we wish to bring into the context of persistence. Moving forward, we will consider an abstract simplicial complex K with $dim(K) = n$ and an abstract simplicial complex K_0 such that $K_0 \subseteq K$, rather than more general topological spaces X and A.

Figure 2.2 The infinite bouquet (right) is a canonical example of a quotient space, and quotients $\mathbb{Z}$ from $\mathbb{R}$ (figure from [12]).

Many of the definitions and objects we provided when describing absolute homology have analogous counterparts in the quotient space setting. Define the **relative chain vector space** to be the quotient vector space $C_p(K, K_0) = C_p(K)/C_p(K_0)$ for all $p \in \{0, \ldots, n\}$, which describes all linear combinations of p-simplicies in the quotient space K/K_0. We refer to such linear combinations as **relative p-chains**. In fact, this quotient partitions a basis for $C_p(K)$ into cosets of the form $c+C_p(K_0)$, where c can be any p-chain

in K. This implies that the set of equivalence classes $\{[c]\}$ which correspond to these cosets constitute a basis for K/K_0, which exemplifies the fact that these chains actually do describe the quotient of the two spaces. Further, any two chains of a given coset cannot differ by any of the element(s) in K/K_0. Equivalently, if two relative chains differ by a chain in $K \setminus K_0$, then they, by definition, must be in two different cosets of $C_n(K_0)$ in $C_n(K)$.

When describing absolute homology, we defined a boundary operator $\partial_p : C_p(K) \to C_{p-1}(K)$, where ∂_p maps any basis vector in $C_p(K)$ to a chain which is its boundary in $C_{p-1}(K)$. However, since ∂_p will always map $C_p(K_0)$ to $C_{p-1}(K_0)$, then the **relative boundary operator**, which maps relative chains to their boundaries, is induced by the boundary operator ∂_p. This implies that the chain complex $\mathcal{C}_\bullet(K_0)$ is induced by the chain complex $\mathcal{C}_\bullet(K)$, and therefore we may define a natural inclusion between corresponding chain vector spaces of the chain complexes $\mathcal{C}_\bullet(K)$ and $\mathcal{C}_\bullet(K_0)$. These inclusions, or **chain maps**, given by $f_p : C_p(K_0) \to C_p(K)$ give rise to the commutative diagram seen below, which induces inclusions on the homology groups (resp. vector spaces) of each chain complex. We summarize this idea with the following lemma.

$$\begin{array}{ccccccccccc}
0 & \xrightarrow{\partial_{n+1}} & C_n(K_0) & \xrightarrow{\partial_n} & C_{n-1}(K_0) & \xrightarrow{\partial_{n-1}} & \dots & \xrightarrow{\partial_1} & C_0(K_0) & \xrightarrow{\partial_0} & 0 \\
 & & \downarrow{\scriptstyle f_n} & & \downarrow{\scriptstyle f_{n-1}} & & & & \downarrow{\scriptstyle f_0} & & \\
0 & \xrightarrow[\partial_{n+1}]{} & C_n(K) & \xrightarrow[\partial_n]{} & C_{n-1}(K) & \xrightarrow[\partial_{n-1}]{} & \dots & \xrightarrow[\partial_1]{} & C_0(K) & \xrightarrow[\partial_0]{} & 0
\end{array}$$

Lemma 1. *The chain map f_p induces the homomorhpishm* (*resp. linear transformation*) $f_p^* : H_p(K_0) \to H_p(K)$ *on homology groups* (*resp. vector spaces*).

In order to describe the homology of K/K_0, we need to first understand what it means for a chain to be a cycle or a boundary in K/K_0.

Definition 2. *A Relative p-Cycle is any p-chain $\alpha \in C_p(K)$ such that $\partial_p(\alpha) \in C_{p-1}(K_0)$.*

In absolute homology, we interpret any chain $\alpha \in C_p(K)$ to be a p-cycle if and only if $\partial_p(\alpha)$ is trivial. Similarly, any chain $\sigma \in C_p(K)$ is a relative p-cycle if and only if $\partial_p(\sigma) \in C_{p-1}(K_0)$, where we take the chains in $C_{p-1}(K_0)$ to be trivial. This idea is illustrated in Figure 2.3 below, where the boundary of the 1-chain which spans the width of K are the two 0-simplices in the subspace K_0. An equivalent interpretation of Definition 2 is that any chain in the coset $c + C_p(K_0)$ is a relative cycle if and only if $\partial_p(c) \in C_{p-1}(K_0)$. A

nice consequence of this definition is that any absolute p-chain $\alpha \in \text{Ker}(\partial_p)$ must also be a relative cycle since $\partial_p(\alpha) = \vec{0}$ and $\vec{0} \in C_p(K_0)$ trivially.

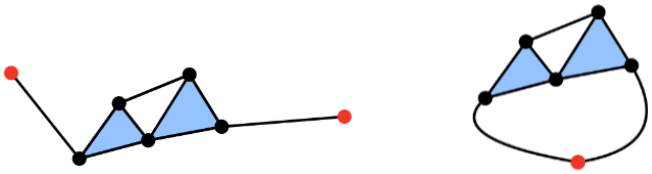


Figure 2.3 Simplicial complexes $K_0 \subseteq K$ (left) and the quotient space K/K_0 (right) forming a relative 1-chain.

Definition 3. *A Relative p-Boundary is any relative p-cycle $\alpha = \partial_{p+1}(\beta) + \gamma$ for some $\beta \in C_{p+1}(K)$ and $\gamma \in C_p(K_0)$.*

In words, if a relative p-cycle α differs from an absolute p-boundary $\partial_{p+1}(\beta)$ by some (possibly trivial) p-chain $\gamma \in C_p(K_0)$, then α must also be a relative p-boundary. This idea is illustrated below in Figure 2.4. To make the example more explicit, call the red 1-simplex γ and the adjacent 2-simplex β. Now let $\alpha \in C_1(K)$ be given by $\alpha = \partial_2(\beta) + \gamma$. Further, we know that $\partial_1(\partial_2(\beta))$ is trivial by the definition of a chain complex. Thus

$$\partial_1(\alpha) = \partial_1(\partial_2(\beta)) + \partial_1(\gamma) = \partial_1(\gamma) \in C_0(K_0).$$

Equivalently, α is a relative 1-cycle that is also a relative 1-boundary by Definition 3. As one might expect, the **relative homology group** $H_p(K, K_0)$ is the quotient group or vector space which describes the difference between all relative p-cycles and relative p-cycles which are also the boundary of a relative $(p+1)$-chain.

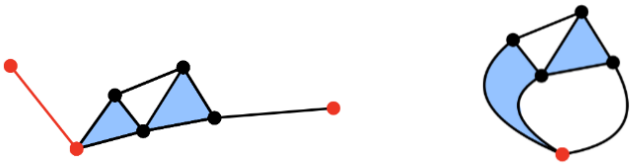


Figure 2.4 Simplicial complexes $K_0 \subseteq K$ (left) and the quotient space K/K_0 (right) forming a relative 1-boundary.

To extend relative homology into persistence, we must place N-level filtrations $F_\bullet$ and $G_\bullet$ respectively on the simplicial complexes K and K_0 such that $G_t K_0 \subseteq F_t K$ for all $t \in \{1, \ldots, N\}$, as seen below in Figure 2.5. In

the case that this condition is satisfied, then the quotient vector spaces given by $H_p(F_tK, G_tK_0)$ for any $p \in \{0, \ldots, n\}$ and any $t \in \{1, \ldots, N\}$ describe the persistent homology of K modulo K_0. This is **persistent relative homology**.

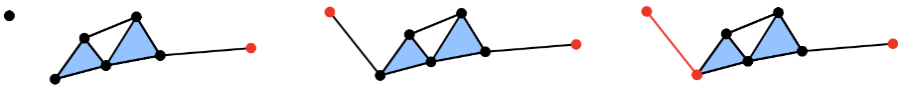


Figure 2.5 A filtered simplicial complex K and a filtered simplcial complex K_0 (in red) satisfying that $G_tK_0 \subset F_tK$ for any t.

Suppose we write K and K_0 as the pair (K, K_0) and introduce the pair of simplicial complexes (L, L_0) such that $L_0 \subseteq L$, $K_0 \subseteq L_0$ and $K \subseteq L$. In this setting, there is a natural inclusion of spaces $K \hookrightarrow L$ which also maps $K_0 \hookrightarrow L_0$. In fact, this gives rise to the chain map $g_p : C_p(K, K_0) \to C_p(L, L_0)$. Therefore, Lemma 1 has an equivalent formulation and accompanying commutative diagram in terms of relative homology [11].

$$\begin{array}{ccccccccc} 0 & \xrightarrow{\partial_{n+1}} & C_n(K, K_0) & \xrightarrow{\partial_n} & \ldots & \xrightarrow{\partial_1} & C_0(K, K_0) & \xrightarrow{\partial_0} & 0 \\ & & \downarrow{\scriptstyle g_n} & & & & \downarrow{\scriptstyle g_0} & & \\ 0 & \xrightarrow[\partial_{n+1}]{} & C_n(L, L_0) & \xrightarrow[\partial_n]{} & \ldots & \xrightarrow[\partial_1]{} & C_0(L, L_0) & \xrightarrow[\partial_0]{} & 0 \end{array}$$

Lemma 2. *The chain map g_p induces the homomorphism (resp. linear transformation) $g_p^* : H_p(K, K_0) \to H_p(L, L_0)$ on relative homology groups (resp. vector spaces).*

By this fact, a filtration of the quotient space K/K_0 will give rise to induced homomorphisms of the form

$$g_p^* : H_p(F_t(K), G_t(K_0)) \to H_p(F_{t+\varepsilon}(K), G_{t+\varepsilon}(K_0)),$$

where $\varepsilon \geq 0$. Lemma 2 allows us to better understand the structure of PRH, and will later be useful in guaranteeing the stability of algorithms used for its computation.

1.2 Previous Work

In [13], Morozov presents an algorithm for persistent relative homology based on the standard matrix reduction $R = DV$. This algorithm is cubic in

the number of simplices, and assumes there are two simplicial complexes $K_0 \subseteq K$ with an injective function $f : K \to \mathbb{R}$ whose sublevel sets are subcomplexes of K along with an injective function g given by a restriction of f to K_0 whose sublevel sets are subcomplexes of K_0. The algorithm proceeds with two reductions to give

$$R_f = D_f V_f \qquad\qquad R_g = D_g V_g$$

where D_f is the boundary matrix of K whose rows and columns are ordered by f, and D_g is the boundary matrix of K_0 whose rows and columns are ordered by g. At a high level, the algorithm continues by constructing matrices D_{im}, D_{ker} and D_{cok} by permuting rows and/or replacing columns of D_f. Reductions on each of these three matrices are performed, allowing for the construction of persistence diagrams corresponding to nested sequences of images, kernels and cokernels.

The algorithm we present shares similar features, but a few key differences. First, the assumption that g is a restriction of f to K_0 implicitly assumes that any simplex which is a member of K_0 at the terminating step of the filtration must be a member of K_0 as soon as it is born in K. Our algorithm will not assume this is the case. Second, our algorithm will only require two reduction steps rather than five. Third, while we do prioritize information in the subcomplex K_0 by reordering rows of the boundary matrix, this step is only performed once.

A related and important computation is (persistent) local homology, which is a special case of relative homology where the subspace that we quotient is one single point or simplex. Morozov's work in [13] also describes this problem. More recently, Kerber and Söls introduced the *localized bifiltration* [14] where data is filtered by scale and a second filtration parameter describes the locality of the scale filtration about a single point in the data. In [15], the authors describe the local homology of abstract simplicial complexes with applications in the analysis of graphs and hypergraphs. In [16], the authors present a theoretical framework for the identification of critical nodes or bottlenecks in a network which replaces computational expensive graph algorithms with a local homology approach. This previous work in persistent relative and local homology is largely theoretical with a few instances where implementations or applications are explored, suggesting the need for an efficient and flexible method of computation.

2 The U-Match Decomposition

The algorithm which we present and prove in the following chapter is an extension of the U-Match Decomposition introduced in [8]. It is therefore essential to understand how and why the U-Match works. This section provides a concise description of the standard U-Match algorithm and the statement of a few lemmas which will be essential for ensuring the correctness of the algorithm we present in the following chapter. We close this section with two brief discussions about the U-Match which are useful for our purpose. First, a description of the U-Match as it relates to persistent homology. Second, a point of view in which the U-Match acts as a change of basis from the standard simplex basis of a simplicial complex to certain bases of chains.

Definition 4. *A U-Match Decomposition is a tuple of matrices* $(\mathcal{T}, M, D, \mathcal{S})$ *which satisfy the following three conditions*:

- $\mathcal{T}M = D\mathcal{S}$.
- M *is a matching matrix.*
- $\mathcal{T}$ *and* $\mathcal{S}$ *are both upper-triangular and invertible.*

By abuse of notation, we will refer to a U-Match decomposition simply as $\mathcal{T}M = D\mathcal{S}$.

In our case, we always consider the matrix D to be the **block boundary matrix** of a chain complex given by

$$D = \begin{pmatrix} 0 & \partial_1 & & & \\ & 0 & \partial_2 & & \\ & & \ddots & \ddots & \\ & & & 0 & \partial_n \\ & & & & 0 \end{pmatrix}.$$

The **matching matrix** M is the reduced form of D. It has coefficients from the field $\mathbb{Z}_2$ and contains at most one nonzero entry per row and column. The matrix $\mathcal{T}$, or the **target matrix**, contains information about the image of D. The matrix $\mathcal{S}$, or the **source matrix**, contains information about the kernel of D. There are a number of methods to compute a U-Match, but one method is as follows. The reduction of D to M proceeds bottom to top and left to right, using $\mathcal{T}^{-1}$ to record row operations and $\mathcal{S}$ to record column

operations as seen below. We call $\mathcal{T}$ and $\mathcal{S}$ **Columnar Ordered Matching Bases**, or COMBs.

$$\begin{pmatrix} D & I_n \\ I_m & 0 \end{pmatrix} \mapsto \begin{pmatrix} M & \mathcal{T}^{-1} \\ \mathcal{S} & 0 \end{pmatrix}$$

The notation $\mathbb{K}^{n\times m}$ denotes the set of $n \times m$ matrices with coefficients in the field $\mathbb{K}$. Let $COL_I(D)$ denote columns of $D \in \mathbb{K}^{n\times m}$ indexed by the set $I \subseteq \{1, \ldots, m\}$ and similarly define $ROW_J(D)$ for $J \subseteq \{1, \ldots, n\}$. Using this notation, the following from [8] describes an efficient implementation of the U-Match Decomposition.

Algorithm 1 U-Match Matrix Decomposition

Require: Matrix $D \in \mathbb{K}^{n\times m}$.

Ensure: Upper uni-triangular matrices $\mathcal{T}^{-1} \in \mathbb{K}^{m\times m}, \mathcal{S}^{-1} \in \mathbb{K}^{n\times n}$ and matching matrix $M \in \mathbb{Z}_2^{m\times n}$ such that $\mathcal{T}^{-1}D = M\mathcal{S}^{-1}$. The corresponding U-match decomposition is $\mathcal{T}M = D\mathcal{S}$.

1: $\mathcal{T}^{-1} \leftarrow I^{m\times m}$, $\mathcal{S}^{-1} \leftarrow I^{n\times n}$, $M \leftarrow 0^{m\times n}$
2: **for** $i \leftarrow m$ **to** 1 **do**
3: **while** $\exists j \in \{i+1, \ldots, m\}$ and $k \in \{1, \ldots, n\}$ such that $ROW_i(D)$ and $ROW_j(D)$ both have leading nonzero entries in column k **do**
4: $ROW_i(D) \leftarrow ROW_i(D) - \frac{D[i,k]}{D[j,k]} ROW_j(D)$
5: $ROW_i(\mathcal{T}^{-1}) \leftarrow ROW_i(\mathcal{T}^{-1}) - \frac{D[i,k]}{D[j,k]} ROW_j(\mathcal{T}^{-1})$
6: **end while**
7: **end for**
8: **for** $i \leftarrow 1$ **to** m **do**
9: **if** for some k, $D[i,k]$ is the leading entry of $ROW_i(D)$ **then**
10: $ROW_k(\mathcal{S}^{-1}) \leftarrow \frac{1}{D[i,k]} ROW_i(D)$
11: $M[i,k] \leftarrow D[i,k]$
12: **end if**
13: **end for**

Algorithm 1 can be applied more generally and not just in the case where D is square or a boundary matrix. The while loop at line 3 reduces the rows of D by locating pivots from the bottom of the matrix. These operations are recorded in $\mathcal{T}^{-1}$, which is upper-triangular and invertible by construction since it is initialized as the identity and constructed with bottom to top linear combinations of its rows. The for loop at line 8 constructs $\mathcal{S}^{-1}$ row-by-row. Each iteration of this loop identifies the column index, k, which

contains the leading entry in the i^{th} row of M. This entry must be the pivot which is recorded in the matching matrix M since the column reduction of D proceeds left to right. Thus, $M[i,k] \neq 0$. Therefore $ROW_i(M\mathcal{S}^{-1}) = ROW_i(M)\mathcal{S}^{-1} = M[i,k]ROW_k(\mathcal{S}^{-1}) = D[i,k](\mathcal{S}^{-1})$. So $\mathcal{T}^{-1}D = M\mathcal{S}^{-1}$ giving $\mathcal{T}M = D\mathcal{S}$ as required. A more detailed proof of correctness and a memory efficient **clear and compress** implementation of Algorithm 1 is contained in [8].

In the case that D is the boundary matrix of a chain complex, then D is square and $D^2 = 0$, which is equivalent to the statement that $\partial_{p-1}\partial_p = 0$ for any $p \in \{1, \ldots, n\}$. Let $r_\bullet$ and $c_\bullet$ be sets which denote, respectively, the indices of nonzero rows and indices of nonzero columns in the matching matrix M of the U-Match $\mathcal{T}M = D\mathcal{S}$. With this notation and the assumption that $D^2 = 0$, we have the following lemma regarding the structure of M.

Lemma 3. *The sets $r_\bullet$ and $c_\bullet$ are disjoint. Thus, $r_\bullet \subseteq \overline{c_\bullet}$, where $\overline{c_\bullet}$ are the indices of zero columns in M.*

Proof. Given the U-Match $\mathcal{T}M = D\mathcal{S}$, we see that $\mathcal{S}^{-1}\mathcal{T}M = \mathcal{S}^{-1}D\mathcal{S}$. But $D^2 = 0 \Rightarrow (\mathcal{S}^{-1}D\mathcal{S})^2 = 0 \Rightarrow (\mathcal{S}^{-1}\mathcal{T}M)^2 = 0$. But $\mathcal{S}^{-1}\mathcal{T}$ is invertible and upper-triangular and M is a matching matrix, therefore implying that the nonzero columns of $\mathcal{S}^{-1}\mathcal{T}M$ form a linearly independent set. Furthermore, since we know $(\mathcal{S}^{-1}\mathcal{T}M)^2 = 0$, then it follows that for any row r_i and any column c_j in $\mathcal{S}^{-1}\mathcal{T}M$, the dot product $r_i \cdot c_j = 0$. This holds for the case $i = j$, therefore implying that the following sets are disjoint:

$$R := \{i : \text{ row i of } \mathcal{S}^{-1}\mathcal{T}M \text{ is nonzero}\}$$
$$C := \{j : \text{ column j of } \mathcal{S}^{-1}\mathcal{T}M \text{ is nonzero}\}$$

Since $\mathcal{S}^{-1}\mathcal{T}$ is invertible and M is a matching matrix, it follows that the set C is precisely the set $c_\bullet$, and the set R is precisely the set $r_\bullet$. This concludes the proof. □

It is important to note that the assumption that $D^2 = 0$ is necessary, and thus Lemma 3 holds only for this particular class of U-Match Decompositions. We include the statement of this lemma as it gives the following corollaries which imply that the U-Match Decomposition may be used to compute homology.

Corollary 3.1. *$COL_{r_\bullet}(\mathcal{T})$ is a basis for Im(D).*

Proof. Suppose that $COL_j(\mathcal{T}M) \neq \vec{0}$, then we must have $COL_j(D\mathcal{S}) \neq \vec{0}$. Additionally, by properties of matrix multiplication it must be true that

$$COL_j(\mathcal{T}M) = COL_j(D\mathcal{S})$$
$$\mathcal{T} \cdot COL_j(M) = D \cdot COL_j(\mathcal{S}).$$

Consider the right-hand side (RHS) first. Since D is a boundary matrix and the columns of $\mathcal{S}$ contain a chain basis for some simplicial complex K, then the RHS asserts that $COL_j(\mathcal{T}M)$ must be the boundary of some chain in K. Equivalently, $COL_j(\mathcal{T}M) \in \text{Im}(D)$ when $COL_j(\mathcal{T}M) \neq \vec{0}$. Now consider the left-hand side (LHS). Since M is a matching matrix and $\mathcal{T} \cdot COL_j(M) \neq \vec{0}$, then $COL_j(M)$ must contain exactly one nonzero entry m_{ij} occurring in (wlog) row i. In other words,

$$COL_j(\mathcal{T}M) = \mathcal{T} \cdot COL_j(M) = COL_i(\mathcal{T}) \cdot M = m_{ij} COL_i(\mathcal{T}).$$

Since $i \in r_\bullet$ and $COL_j(\mathcal{T}M) \in \text{Im}(D)$, this concludes the proof. □

Corollary 3.2. *$COL_{\overline{c_\bullet}}(\mathcal{S})$ is a basis for Ker(D).*

Proof. Consider the column vector $COL_j(D\mathcal{S})$. Again, by properties of matrix multiplication we must have $COL_j(D\mathcal{S}) = D \cdot COL_j(\mathcal{S})$. Since D is a boundary matrix and $COL_j(\mathcal{S})$ is just some chain in a simplicial complex K, then we want to show that $j \in \overline{c_\bullet} \Rightarrow D \cdot COL_j(\mathcal{S}) = 0$. Again, we note that $\mathcal{T} \cdot COL_j(M) = D \cdot COL_j(\mathcal{S})$. But since $j \in \overline{c_\bullet}$ then

$$\mathcal{T} \cdot COL_j(M) = \mathcal{T} \cdot \vec{0} = \vec{0},$$

which concludes the proof. □

These corollaries along with the fact that $r_\bullet \subseteq \overline{c_\bullet}$ imply that $rank(D) \leq nullity(D)$, which we should expect given that D is the boundary matrix of a chain complex. In fact, we may actually strengthen this statement further to show that the U-Match Decomposition can indeed be used to compute homology.

Theorem 4. *Given a U-Match $\mathcal{T}M = D\mathcal{S}$ satisfying that $D^2 = 0$, Im$(D) \subseteq$ Ker(D).*

The U-Match Decomposition can also compute persistent homology. Given a filtered simplicial complex K, index rows and columns of D by simplices in K with increasing filtration value in $F_\bullet K$ as column and row

indices increase. After performing a U-Match, this ordering is carried over to $\mathcal{T}^{-1}$ and $\mathcal{S}$ since they are initialized as identity matrices and respectively constructed with bottom-to-top linear combinations of rows and left-to-right linear combinations of columns. Thus, the rows of $\mathcal{T}$ and $\mathcal{S}$ are indexed by simplices in K according to increasing filtration value in $F_\bullet K$, whereas their columns contain chains recorded during the reduction of D which are ordered by increasing filtration value in $F_\bullet K$. This is illustrated below, and discussed further in the following chapter.

$$
\begin{array}{cc}
\begin{array}{c} \\ s \in F_\bullet K \end{array}
\begin{array}{c} c \in F_\bullet K \\ \begin{pmatrix} \mathcal{T} \end{pmatrix} \end{array}
&
\begin{array}{c} \\ c \in F_\bullet K \end{array}
\begin{array}{c} c \in F_\bullet K \\ \begin{pmatrix} M \end{pmatrix} \end{array}
\\[2em]
\begin{array}{c} \\ s \in F_\bullet K \end{array}
\begin{array}{c} s \in F_\bullet K \\ \begin{pmatrix} D \end{pmatrix} \end{array}
&
\begin{array}{c} \\ s \in F_\bullet K \end{array}
\begin{array}{c} c \in F_\bullet K \\ \begin{pmatrix} \mathcal{S} \end{pmatrix} \end{array}
\end{array}
$$

Figure 2.6 Matrices of a U-Match $\mathcal{T}M = D\mathcal{S}$ where $s \in F_\bullet K$ and $c \in F_\bullet K$ respectively denote simplices and chains ordered by $F_\bullet K$. Filtration value increases with increasing row and column index.

Let $\mathcal{V}_K$ (resp. $\mathcal{V}_{K_0}$) be the vector space which spans all chains in the simplicial complex K (resp. K_0). Then D is a linear transformation of the form $D : \mathcal{V}_K \to \mathcal{V}_K$ (resp. $D_0 : \mathcal{V}_{K_0} \to \mathcal{V}_{K_0}$) mapping chains expressed in a **simplex basis** to their boundary expressed in a simplex basis. The simplex basis is analogous to the idea of the standard basis, and we denote it as $\mathcal{B}_S$. Further, since the reduction of D records row operations in $\mathcal{T}^{-1}$, then the chains which index the columns of $\mathcal{T}$ must span $\mathcal{V}_K$. We refer to this chain basis as the **row operation basis** denoted $\mathcal{B}_R$. Thus, $\mathcal{T} : \mathcal{V}_K \to \mathcal{V}_K$ may be considered as a change of basis matrix, taking chains expressed in terms of $\mathcal{B}_R$ to chains expressed in terms of $\mathcal{B}_S$. Similarly, the columns of $\mathcal{S}$ must contain a **column operation basis**, which we denote $\mathcal{B}_C$. We use the notation $\mathcal{V}_K$ and $\mathcal{V}_{K_0}$ as well as this change of basis point of view extensively in the following chapter.

3. An Algorithm for PRH

In this chapter, we provide pseudocode and proofs of correctness for an algorithm for persistent relative homology, which consists of a few general steps. First, a **relative boundary matrix** $\mathcal{D}$ is constructed via a permutation of rows of the block boundary matrix D. Second, $\mathcal{D}$ is reduced to give the U-Match $\mathcal{T}M = \mathcal{D}\mathcal{S}$, and columns of the COMBs are permuted to give a filtration of relative cycles and relative boundaries. Third, a second U-Match is performed to construct a single matrix whose columns contain a filtered basis for the relative cycles and relative boundaries. We refer to such bases as being **matched**. We state a number of lemmas throughout this chapter which will be used to prove the correctness of this algorithm.

1 Constructing the Boundary Matrix

Our starting point for the algorithm is the block boundary matrix of a simplicial complex K equipped with an N-level filtration $F_\bullet K$. As mentioned in Section 2.2, the rows of D are indexed by the simplices in K and ordered top-to-bottom by increasing filtration value in $F_\bullet K$. The columns of D are ordered similarly left-to-right. Importantly, we also assume there exists some subcomplex $K_0 \subseteq K$ identically equipped with an N-level filtration such that $G_t K_0 \subseteq F_t K$ for any $t \in \{1, \dots, N\}$. To construct the relative boundary matrix $\mathcal{D}$, permute rows of D to be indexed from top-to-bottom by simplices in K_0 by increasing filtration value in $G_\bullet K_0$ followed by simplices in $K \setminus K_0$ ordered by increasing filtration value in $F_\bullet K$. We say that the rows of $\mathcal{D}$ **respect** the filtration $G_\bullet K_0$, while the columns of $\mathcal{D}$ respect the filtration $F_\bullet K$. This step is illustrated below in Figure 3.1.

To motivate why this step is necessary, recall $C_p(K, K_0)$ partitions $C_p(K)$ into cosets of the form $c + C_p(K_0)$ where c is any p-chain in K. Since the U-Match reduces $\mathcal{D}$ with bottom-to-top linear combinations of rows, then these linear combinations will record relative chains in the rows of $\mathcal{T}^{-1}$. This idea is exemplified in Figure 3.1, where the rows in blue are indexed by the simplices corresponding to the equivalence classes $\{[c]\}$, and the rows in red are indexed by a simplex basis for $\mathcal{V}_{K_0}$. This step will be rather useful when proving the correctness of subsequent and less-trivial steps of the algorithm.

For the sake of implementation, it is important to describe how the

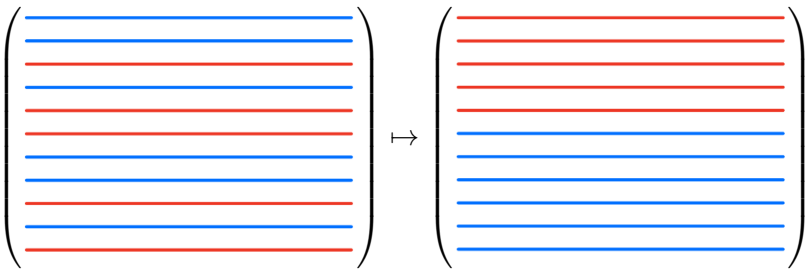


Figure 3.1 The rows of a block boundary matrix before (left) and after (right) the permutation to construct the relative boundary matrix. The rows highlighted in red are simplices in K_0, and the rows in blue are simplices in $K \setminus K_0$.

matrix D may be constructed from point cloud data, as this step is necessary to then construct $\mathcal{D}$. Assume that the point cloud data is stored in a distance matrix M_1, and some subset of this data is also stored in a distance matrix M_2. Note that M_2 should be the same size as M_1, using zero entires for data which is not contained in the subspace. In fact, these matrices implicitly contain information about how to construct a **Vietoris-Rips (VR) complex**. For example, suppose that the points $\{x_1, x_2, \ldots, x_{p+1}\}$ which index some subset of the rows and columns of M_1 satisfy that $d(x_i, x_j) < \varepsilon_t$ for any $i, j \in \{1, \ldots, p+1\}$. In words, we say that these points are pairwise-close within ε_t. Now, use these points to form a p-simplex σ with **diameter** ε_t and filtration value t. We write $diam(\sigma)$ to denote the diameter of σ. To construct the filtered simplicial complex $F_\bullet K$, then use a **scale parameter** given by $\varepsilon_1 \leq \varepsilon_2 \leq \cdots \leq \varepsilon_N$. In particular, this means that the simplicial complex $F_t K$ should contain all simplicies constructed from sets of points which are pairwise-close within ε_t. The same process can be used to construct a VR complex from M_2, thus giving the filtered subcomplex $G_\bullet K_0$. However, we must take care to ensure that this method constructs each VR complex such that $G_t K_0 \subseteq F_t K$. Suppose that $\delta_1 \leq \delta_2 \leq \cdots \leq \delta_N$ is the scale parameter used to construct a VR complex from M_2. Then the required property will be satisfied if and only if $\delta_t \leq \varepsilon_t$ for each $t \in \{1, \ldots, N\}$, where we assume that both are N-level filtrations. In addition, since it may be true that $\delta_t < \varepsilon_t$, then we do not assume that any simplex in K_0 enters the filtration $G_\bullet K_0$ immediately upon birth in $F_\bullet K$.

Importantly, each simplex constructed by the above method will be

assigned a filtration value (birth) and a diameter. In general, if σ and τ are simplices in a VR complex where $b(\sigma) = t_1$ and $b(\tau) = t_2$, then $diam(\sigma) < diam(\tau)$ if and only if $t_1 < t_2$. In other words, either of these measures can be used to permute the rows of D as needed. A block boundary matrix may be stored explicitly or as a sparse matrix, although most applications will require sparse matrix formats due to the tendency of boundary matrices to be exceedingly large for any practical application. In either case, all that is needed to construct D is a filtered list of simplices in K, which we denote $\mathcal{F}$. Similarly, all that is needed to construct $\mathcal{D}$ is the list $\mathcal{F}$ and a filtered list of simplices in K_0, which we denote $\mathcal{G}$. Therefore, the list of simplices which index the rows of the relative boundary matrix $\mathcal{D}$ may then be constructed with a simple sorting algorithm which uses an order operator to compare any two simplices in $\mathcal{F}$, as seen below.

Algorithm 2 Relative Boundary Matrix Row Order Operator

Require: Two simplices $\sigma, \tau \in \mathcal{F}$.
Ensure: True if σ before τ, and false otherwise.
1: **if** $\sigma \in \mathcal{G}$ and $\tau \notin \mathcal{G}$ **then**
2: Return True
3: **end if**
4: **if** $\sigma \notin \mathcal{G}$ and $\tau \in \mathcal{G}$ **then**
5: Return False
6: **end if**
7: **if** $\sigma, \tau \in \mathcal{G}$ **then**
8: Return $b_G(\sigma) < b_G(\tau)$
9: **end if**
10: **if** $\sigma, \tau \notin \mathcal{G}$ **then**
11: Return $b_F(\sigma) < b_F(\tau)$
12: **end if**

Lemma 5 (Relative Boundary Matrix)**.** *Algorithm* **2** *correctly orders any two simplices $\sigma, \tau \in \mathcal{F}$ given a filtered list $\mathcal{G}$.*

Proof. We require that any simplex in K_0 appear before any simplex in $K \setminus K_0$, regardless of diameter or filtration value. Lines 1 and 4 above handle these cases. In the case that $\sigma, \tau \in K_0$ then ordering must be determined by comparing $b_G(\sigma)$ and $b_G(\tau)$. This case is handled at line 7 above. In the case that $\sigma, \tau \in K \setminus K_0$ then ordering must be determined by comparing $b_F(\sigma)$

and $b_F(\tau)$. This is handled similarly to the previous case, as seen above at line 10. □

2 Decomposition and Permuting COMBs

Now that we have a relative boundary matrix, we can perform a U-Match Decomposition using Algorithm 1 to obtain $\mathcal{T}M = \mathcal{D}\mathcal{S}$.

$$\begin{array}{cc} & c \in G_\bullet K_0 \\ s \in G_\bullet K_0 & \left(\begin{array}{c} \mathcal{T} \end{array} \right) \end{array} \qquad \begin{array}{cc} & c \in F_\bullet K \\ c \in G_\bullet K_0 & \left(\begin{array}{c} M \end{array} \right) \end{array}$$

$$\begin{array}{cc} & s \in F_\bullet K \\ s \in G_\bullet K_0 & \left(\begin{array}{c} \mathcal{D} \end{array} \right) \end{array} \qquad \begin{array}{cc} & c \in F_\bullet K \\ s \in F_\bullet K & \left(\begin{array}{c} \mathcal{S} \end{array} \right) \end{array}$$

Figure 3.2 The matrices of a U-Match $\mathcal{T}M = \mathcal{D}\mathcal{S}$. The notation $s \in F_\bullet K$ and $c \in F_\bullet K$ have the same meaning as in Figure 2.6, while $s \in G_\bullet K_0$ and $c \in G_\bullet K_0$ respectively denote simplices and chains which respect the filtration $G_\bullet K_0$.

Since we are using the matrix $\mathcal{D}$ for this U-Match Decomposition, there are a few key differences when compared to the U-Match $\mathcal{T}M = D\mathcal{S}$, as Figure 3.2 suggests. Most importantly, since the rows of $\mathcal{D}$ respect the filtration $G_\bullet K_0$, this ordering is carried over to the rows and columns of $\mathcal{T}$ and the rows of M. This idea is formalized with the following proposition.

Proposition 1. *Suppose that $\mathcal{V}_{K_0}$ is a vector space with dimension i spanning all chains in K_0. Then, the first i columns of $\mathcal{T}$ from the U-Match $\mathcal{T}M = \mathcal{D}\mathcal{S}$ are a basis which spans $\mathcal{V}_{K_0}$.*

Proof. Let I be the set $\{1, ..., i\}$. By the construction of $\mathcal{D}$ and the properties of Algorithm 1, the row operation matrix $\mathcal{T}^{-1}$ is a linear transformation $\mathcal{V}_K \to \mathcal{V}_K$ taking chains in the basis $\mathcal{B}_S$ to chains in the basis $\mathcal{B}_R$. Equivalently, the construction of $\mathcal{D}$ necessarily forces the bases which index the rows and columns of $\mathcal{T}^{-1}$ to inherit the ordering placed on the rows of $\mathcal{D}$. This implies that $\mathcal{T} : \mathcal{V}_K \to \mathcal{V}_K$ maps chains in the basis $\mathcal{B}_R$ to chains in the

basis $\mathcal{B}_S$. In particular, the columns of $\mathcal{T}$ inherit an ordering given by the filtered basis $\mathcal{B}_R$. Furthermore, since $ROW_I(\mathcal{D})$ spans $\mathcal{V}_{K_0}$ by construction and $\mathcal{T}^{-1}$ is constructed with bottom to top linear combinations of its rows, then $ROW_I(\mathcal{T}^{-1})$ must span $\mathcal{V}_{K_0}$. Therefore, $COL_I(\mathcal{T})$ must span $\mathcal{V}_{K_0}$ as well, which completes the proof. □

With this in mind, we are now ready to prove that this modified U-Match can compute relative homology. For the proof, we use the previously introduced notation $\mathcal{V}_K$ (resp. $\mathcal{V}_{K_0}$) to denote the vector space which spans all chains in the simplicial complex K (resp. K_0).

Lemma 6 (Containment)**.** *Suppose a U-Match $\mathcal{T}M = \mathcal{D}\mathcal{S}$. Then*

(*i.*) *the columns of $\mathcal{S}$ contain a basis for the relative cycles denoted RelKer($\mathcal{D}$).*

(*ii.*) *the columns of $\mathcal{T}$ contain a basis for the relative boundaries denoted RelIm($\mathcal{D}$).*

Proof. (i.) By Definition 2, a relative p-cycle is any $\alpha \in C_p(K)$ such that $\partial_p(\alpha) \in C_{p-1}(K_0)$. Equivalently, we may also write this in terms of the matrix $\mathcal{D}$ as $\alpha \in \mathcal{V}_K$ such that $\mathcal{D}\alpha \in \mathcal{V}_{K_0}$. Suppose that $\alpha \in \mathcal{V}_K$ satisfies this property. There are two cases to consider:

> **Case 1:** Since $\vec{0} \in \mathcal{V}_{K_0}$ trivially, then any α which is an absolute cycle is also a relative cycle. By Corollary 3.2, the columns of $\mathcal{S}$ given by $COL_{\overline{c_\bullet}}(\mathcal{S})$ give a basis for Ker($\mathcal{D}$).
>
> **Case 2:** We now show that columns of $\mathcal{S}$ contain a basis for any α which are relative cycles but not also absolute cycles. Equivalently, any α which satisfy $\mathcal{D}\alpha \in \mathcal{V}_{K_0}$ is nontrivial. Let $I = \{1, \ldots, i\}$. Recall by Proposition 1 that if $\mathcal{V}_{K_0}$ has dimension i, then $COL_I(\mathcal{T})$ spans $\mathcal{V}_{K_0}$. Thus, $COL_I(\mathcal{T})$ contains all boundaries of relative cycles in its span.
>
> > **Case 2a:** Let $k \in r_\bullet$ and $k \in I$. Then, by Corollary 3.1, the chain $COL_k(\mathcal{T}) \in \mathcal{V}_{K_0}$ is also an absolute boundary and there exits $c \in c_\bullet$ such that
> >
> > $$COL_k(\mathcal{T}) = COL_c(\mathcal{T}M) = COL_c(\mathcal{D}\mathcal{S}) = \mathcal{D} \cdot COL_c(S) = \mathcal{D}\alpha.$$
> >
> > In words, $\alpha = COL_c(\mathcal{S})$ is a chain which is not an absolute cycle and has a boundary in $\mathcal{V}_{K_0}$.

Case 2b: Let $k \in \overline{r_\bullet}$ and $k \in I$ and suppose for contradiction that $\alpha = COL_j(S)$ satisfies that $\mathcal{D}\alpha = COL_k(\mathcal{T})$ for some j. Therefore, we must have

$$\mathcal{D}\alpha = COL_j(\mathcal{D}\mathcal{S}) = COL_j(\mathcal{T}M) = COL_k(\mathcal{T}).$$

But for this to hold, we must have $k \in r_\bullet$, which is a contradiction.

Cases 2a and 2b establish that any $\beta \in \mathcal{V}_{K_0}$ with $\beta \neq \vec{0}$ which satisfies $\mathcal{D}\alpha = \beta$ for some $\alpha \in \mathcal{V}_K$ must also satisfy that α is contained within the span of the columns of $\mathcal{S}$.

□

Proof. (ii.) By Definition 3, any relative p-boundary may be written as $\alpha = \partial_{p+1}(\beta) + \gamma$ where $\beta \in C_{p+1}(K)$ and $\gamma \in C_p(K_0)$. When considering any $p \leq dim(K)$, we see that a basis for RelIm($\mathcal{D}$) can be given by a basis for Im($\mathcal{D}$) together with a basis for $\mathcal{V}_{K_0}$. If $\mathcal{V}_{K_0}$ has dimension i, then it is immediate by Corollary 3.1 and Proposition 1 that

$$\text{RelIm}(\mathcal{D}) = COL_{r_\bullet}(\mathcal{T}) \cup COL_I(\mathcal{T}).$$

□

Lemma 6 establishes that our method is capable of computing relative homology in a few simple steps, but is not always useful for identifying the exact columns which constitute RelKer($\mathcal{D}$) and RelIm($\mathcal{D}$). Furthermore, it does not describe if or how we may assign a filtration value to these relative features. The rest of this section addresses these issues by extending our algorithm with methods that determine if and when any column of $\mathcal{S}$ (resp. $\mathcal{T}$) is born as a relative cycle (resp. boundary) throughout a filtration. Before proceeding, recall that the notation $b_F(\alpha)$ is used to denote the birth of a chain α in the filtration $F_\bullet$. On the other hand, when we use the notation $b(\alpha)$ we are referring explicitly to the birth of either a relative cycle or relative boundary. In this case, we do not distinguish birth according to filtration as we assume that the filtrations $F_\bullet K$ and $G_\bullet K_0$ are both N-level, as described in the previous section. We consider the case of relative cycles first.

Algorithm 3 Test Relative Cycle Birth

Require: A positive integer c which is a column index in M that corresponds to the column of $\mathcal{S}$ given by $\alpha = COL_c(\mathcal{S})$.

Ensure: Some $a \in [0, \infty)$ describing the birth of α as a relative cycle.

1: $m \leftarrow COL_c(M)$
2: $x \leftarrow b_G(m)$
3: $y \leftarrow b_F(\alpha)$
4: $a \leftarrow max(x, y)$

Algorithm 3 is simple and is based on the fact that any chain α which is a relative cycle must satisfy that α has been born in the filtration $F_\bullet K$, and $\mathcal{D}\alpha$ has been born in the filtration $G_\bullet K_0$. This follows directly from Definition 2. The former is checked on line 3, while the latter is checked on line 2. This pseudocode mirrors what an implementation of the algorithm might look like, where data structures describing the filtrations $F_\bullet K$ and $G_\bullet K_0$ have been defined. We now provide a proof of correctness for Algorithm 3 which elucidates some of the finer details.

Lemma 7 (Relative Cycle Filtration)**.** *Given a column vector $\alpha = COL_c(\mathcal{S})$ where c is a column index in M, then Algorithm 3 assigns a filtration value at which α is born as a relative cycle.*

Proof. Let $\alpha = COL_c(\mathcal{S})$, where c is some column index of M. Note that the birth of $\mathcal{D}\alpha$ corresponds to the birth of $m = COL_c(M)$ as follows

$$\mathcal{D}\alpha = \mathcal{D} \cdot COL_c(\mathcal{S}) = COL_c(\mathcal{DS}) = COL_c(\mathcal{T}M) = \mathcal{T} \cdot COL_c(M).$$

With this, there are a few simple cases to check.

> **Case 1:** Suppose α is not a relative cycle. It is trivial that $m \notin \mathcal{V}_{K_0}$ and thus $x = \infty$. However, since $b(\alpha) = max(x, y)$, then we must have $b(\alpha) = \infty$, implying that Algorithm 3 does not assign a birth to α as required.
>
> **Case 2:** Suppose α is a relative cycle but not an absolute cycle. Therefore we must have $m \neq \vec{0}$ and $m \in \mathcal{V}_{K_0}$, implying that $x > 0$ is finite. But since $y > 0$ must also be finite then $b(\alpha) = max(x, y)$ is finite as required.
>
> **Case 3:** Suppose α is an absolute cycle. Then $c \in \overline{c_\bullet}$ by Corollary 3.2 and $m = \vec{0} \Rightarrow x = 0$. But since $y > 0$ then $b(\alpha) = y$. Finally, since y is

finite then Algorithm 3 assigns $b(\alpha)$ to be the time step at which α is born as required.

□

Now we turn our attention to the case of relative boundaries. The pesudocode provided for this case is slightly less trivial than the previous, but the proof is no more difficult. Once again, we rely on a simple observation regarding the definition of a relative boundary. In particular, that any relative chain $\alpha \in K/K_0$ will be born as a relative boundary once $\alpha \in \text{Im}(\mathcal{D})$ or once $\alpha \in \mathcal{V}_{K_0}$.

Algorithm 4 Test Relative Boundary Birth

Require: A positive integer r which is a row index in M that corresponds to the column of $\mathcal{T}$ given by $\alpha = COL_r(\mathcal{T})$.
Ensure: Some $a \in [0, \infty)$ describing the birth of α as a relative boundary.

$x \leftarrow b_G(\alpha)$
$m_r \leftarrow ROW_r(M)$
if $r \in r_\bullet$ **then**
    $c \leftarrow$ index of nonzero entry in m_r
    $m_c \leftarrow COL_c(M)$
    $y \leftarrow b_F(m_c)$
end if
if $r \in \overline{r_\bullet}$ **then**
    $y \leftarrow \infty$
end if
$a \leftarrow min(x, y)$

Lemma 8 (Relative Boundary Filtration)**.** *Given a column vector $\alpha = COL_r(\mathcal{T})$ where r corresponds to a row index in M, then Algorithm 4 assigns a filtration value at which α is born as a relative boundary.*

Proof. Algorithm 4 relies on the simple observation that

$$\text{RelIm}(\mathcal{D}) = \text{Im}(\mathcal{D}) \cup \mathcal{V}_{K_0}.$$

Suppose that $\alpha = COL_r(\mathcal{T})$, where r corresponds to the index of a row in the matching matrix M. There are a few simple cases to check.

Case 1: Suppose α is not a relative boundary. By Definition 3, we must have $\alpha \notin \mathcal{V}_{K_0} \Rightarrow x = \infty$. Furthermore, Corollary 3.1 implies that $r \in \overline{r_\bullet} \Rightarrow y = \infty$. Therefore $b(\alpha) = min(x, y) = \infty$ as required.

Case 2: Suppose α is a relative boundary but not an absolute boundary. By Definition 3, we must have $\alpha \in \mathcal{V}_{K_0}$ where $\alpha \neq \vec{0}$. This implies $x > 0$ is finite. Once again, Corollary 3.1 implies $y = \infty$. Therefore, we have $b(\alpha) = x$, as required.

Case 3: Suppose that α is an absolute boundary. In this case, Corollary 3.1 implies $r \in r_\bullet$. Consider the nonzero row of M which corresponds to α, given by $m_r = ROW_r(M)$. Suppose, WLOG, that the nonzero entry in m_r occurs at column index c. Now consider $m_c = COL_c(M)$. Since $m_c \neq \vec{0}$ we have

$$COL_r(\mathcal{T}) = COL_c(\mathcal{T}M) = T \cdot m_c = \mathcal{D} \cdot COL_c(\mathcal{S}).$$

Let $\beta = COL_c(\mathcal{S})$. In fact, we have just shown that α is the boundary of β and that $b_F(\beta)$ corresponds to $b_F(m_c)$. Since the boundary of a chain must be born when that chain is born, then we must have $b(\alpha) = b_F(\beta)$. This equality implies $x = y$ when we have $\beta \in \mathcal{V}_{K_0}$. In the case that $\beta \notin \mathcal{V}_{K_0}$, then $\alpha \notin \mathcal{V}_{K_0} \Rightarrow x = \infty$. In either case, we see that when α is an absolute boundary, we have $b(\alpha) = y$ as required.

□

When applying Algorithm 1 to a relative boundary matrix $\mathcal{D}$ which has been constructed from the filtered quotient space K/K_0, Lemma 6 implies that the resulting U-Match $\mathcal{T}M = \mathcal{D}\mathcal{S}$ contains information about the homology of K/K_0. Combining this result with Algorithms 3 and 4, we may translate this into PRH by simply permuting the columns of the COMBs. Specifically, permute the columns of $\mathcal{T}$ such that relative boundaries appear in increasing birth order as column index increases. Append any column which is not a relative boundary to the end, and call this permuted matrix $\mathcal{A}$. Permute the columns of $\mathcal{S}$ similarly to obtain a matrix $\mathcal{B}$. Similar to the construction of $\mathcal{D}$, this becomes a matter of writing a simple sorting algorithm using Algorithm 3 or Algorithm 4 as an order operator. There is only one issue left, namely that the columns of $\mathcal{T}$ are in the basis $\mathcal{B}_R$ and the columns of $\mathcal{S}$ are in the basis $\mathcal{B}_C$. It would be computationally simple to express the columns of $\mathcal{T}$ and $\mathcal{S}$ such that a basis for RelIm($\mathcal{D}$) is a subset of a basis for RelKer($\mathcal{D}$), allowing for simple extraction of homological generators. As noted earlier, we call such bases in this form **matched**. This final step is addressed in the following section.

3 Matching Bases

We are nearly ready to state the U-Match PRH algorithm in full. To this end, we present one final lemma regarding the properties of the U-Match Decomposition. In fact, the correctness of the final step of the algorithm is an immediate consequence of this lemma. Throughout this section, we use I to denote the set $\{1, \dots, i\}$ and similarly use J to denote the set $\{1, \dots, j\}$. Also, note that the filtrations $F_\bullet$ and $G_\bullet$ which are introduced in Lemma 9 do not correspond to our filtered simplicial complexes $F_\bullet K$ and $G_\bullet K_0$.

Lemma 9 (Basis Matching)**.** *Let $A \in \mathbb{K}^{n\times n}$ be invertible with B being a not necessarily square or invertible matrix. Thus $B \in \mathbb{K}^{n\times m}$ and we allow that $m = n$. In both cases, the matrix multiplication $A^{-1}B$ is compatible. Further, let $F_\bullet$ be a filtration of the vector space $\mathbb{K}^m$ such that $F_i\mathbb{K}^m$ describes the span of the first i columns of A. Similarly define $G_\bullet$ for the columns of B. In the case that the columns of B do not span $\mathbb{K}^m$, then let $G_{m+1} = \mathbb{K}^m$ to ensure that the filtration on the columns of B terminates. Let $D = A^{-1}B$ for a U-Match $\mathscr{T}\mathcal{M} = D\mathscr{S}$. Then the columns of $A\mathscr{T}$ contain a basis for each F_i and G_j where $i, j \in \{1, \dots, m\}$.*

Proof. This rather bulky theorem has a concise proof. First, note that $\text{Im}(\mathcal{A}\mathscr{T}) = \text{Im}(\mathcal{A})$ since $\mathscr{T}$ is invertible by definition. In fact, since $\mathscr{T}$ is also upper-triangular, then each F_i is spanned by $COL_I(\mathcal{A}\mathscr{T})$. On the other hand, consider the U-Match $\mathscr{T}\mathcal{M} = D\mathscr{S}$. It follows that

$$\begin{aligned} \mathscr{T}\mathcal{M} &= D\mathscr{S} \\ \mathscr{T}\mathcal{M} &= (A^{-1}B)\mathscr{S} \\ A\mathscr{T}\mathcal{M} &= B\mathscr{S} \\ \Rightarrow \text{Im}(A\mathscr{T}\mathcal{M}) &= \text{Im}(B\mathscr{S}). \end{aligned}$$

Since $\mathscr{S}$ is upper-triangular and invertible, then each G_j is spanned by $COL_J(\mathcal{A}\mathscr{T}\mathcal{M})$. Now, since M is a matching matrix, then the multiplication $\mathcal{A}\mathscr{T}\mathcal{M}$ simply acts to permute columns of $\mathcal{A}\mathscr{T}$. Thus, each F_i is spanned by some combination of columns in $\mathcal{A}\mathscr{T}$, which concludes the proof. □

Lemma 9 is a particularly nice property of the U-Match for our purpose, and implies the final step needed in order to express the permuted COMBs $\mathcal{A}$ and $\mathcal{B}$ in a matched form. In fact, it is immediate by Lemma 9 that if given the U-Match $\mathscr{T}\mathcal{M} = (\mathcal{A}^{-1}\mathcal{B})\mathscr{S}$, then the columns of the matrix $\mathcal{A}\mathscr{T}$ must contain a basis for both RelKer($\mathcal{D}$) and RelIm($\mathcal{D}$). With this in mind, we summarize the U-Match PRH algorithm with the following theorem.

Theorem 10 (U-Match PRH)**.** *Let K and K_0 be simplicial complexes equipped with finite filtrations $F_\bullet K$ and $G_\bullet K_0$, and suppose that for any filtration value t we have $G_t K_0 \subseteq F_t K$. Apply the following steps*:

1. *Construct relative boundary matrix $\mathcal{D}$ using Algorithm* 2*.*
2. *Apply Algorithm* 1 *to obtain a U-Match decomposition $\mathcal{T}M = \mathcal{D}\mathcal{S}$.*
3. *Use Algorithms* 3 *and* 4 *to obtain the permuted COMBs $\mathcal{A}$ and $\mathcal{B}$ respectively from $\mathcal{T}$ and $\mathcal{S}$.*
4. *Apply Algorithm* 1 *to obtain a U-Match Decomposition $\mathscr{T}\mathcal{M} = (\mathcal{A}^{-1}\mathcal{B})\mathscr{S}$.*

Suppose RelIm($\mathcal{D}$) has dimension i and RelKer($\mathcal{D}$) has dimension j at filtration value t. If the above steps are applied, then the set

$$COL_J(\mathcal{A}\mathscr{T}\mathcal{M}) \setminus COL_I(\mathcal{A}\mathscr{T})$$

contains a basis for $H_p(F_t K, G_t K_0)$ for each $p < dim(K)$.

We have already done all of the work necessary to prove Theorem 10. By construction of $\mathcal{D}$ and Lemmas 6, 7 and 8, matrices $\mathcal{A}$ and $\mathcal{B}$ must contain filtered bases for RelIm($\mathcal{D}$) and RelKer($\mathcal{D}$) at any filtration value. As previously noted, Lemma 9 implies that step 4 will give one single matrix $\mathcal{A}\mathscr{T}$ whose columns give a matched basis for RelKer($\mathcal{D}$) and RelIm($\mathcal{D}$). Finally, note that $\mathcal{M}$ must be a permutation matrix since $\mathcal{A}^{-1}\mathcal{B}$ is full-rank. Thus, the right multiplication of $\mathcal{A}\mathscr{T}$ by $\mathcal{M}$ can be seen as a permutation between versions of $\mathcal{A}\mathscr{T}$ which is either ideal for extracting relative cycles or ideal for extracting relative boundaries.

4. Stability

In this chapter, we provide a theorem guaranteeing the stability of the U-Match PRH algorithm. In fact, this result is derived from a more general notion of the stability of persistence modules arising from filtered quotient spaces. We start by recalling a few key definitions and previous results.

1 Persistence Modules and the Isometry Theorem

Persistence modules describe the theoretical and structural foundation of persistent homology and are often essential when providing results which guarantee the stability of persistence algorithms. It is beyond the scope of this text to give a detailed account of the literature, and thus we provide the details essential to this work, namely a definition of the persistence module and a pseudometric on persistence modules known as the interleaving distance. For the reader who is not familiar with these topics we recommend [17] and [9] as an introduction.

Definition 5. *A persistence module $\mathcal{M}$ defined over some indexing poset $(A, \leq) \subset \mathbb{R}$ is a collection of R-modules $\{M_a\}_{a\in A}$ along with a collection of homomorphisms $m_{a,a'} : M_a \to M_{a'}$ such that $m_{a,a}$ is the identity and $m_{a',a''} \circ m_{a,a'} = m_{a,a''}$ for $a \leq a' \leq a''$.*

It will be convenient to take the R-modules $\mathcal{M}_a$ to be vector spaces V_a with coefficients from a field R, and the homomorphisms to be linear transformations $v_{a,a'} : V_a \to V_{a'}$. Although originally defined in [3] as a graded module over a polynomial ring, this definition from [9] allows us to more explicitly associate the sequence of R-modules $\{M_a\}_{a\in A}$ with a sequence of homology vector spaces related by a collection of induced homomorphisms on homology. Persistence modules provide a means of understanding the algebraic structure of persistent homology and are a key theoretical tool for computational topology and TDA.

The first result ensuring the stability of persistence algorithms is due to [17], in which the authors provide a metric on persistence diagrams known as the **bottleneck distance**, d_B. Let X be a triangulable topological space and define functions $f, g : X \to \mathbb{R}$ which give rise to the persistence

diagrams $D(f)$ and $D(g)$. The bottleneck distance is given by

$$d_B(D(f), D(g)) = \inf_{\gamma} \sup_{x} ||x - \gamma(x)||_\infty,$$

where x ranges over all $x \in D(f)$ and likewise γ ranges over all bijections of the form $\gamma : D(f) \to D(g)$. In words, we choose the bijection between the persistence diagrams which minimizes the maximal distance between corresponding points under the bijection. The key stability result in [17] is that

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

where d_H is the Hausdorff distance. Thus, the bottleneck distance ensures that small changes in the filtration functions imply small changes in the persistence diagram.

For ensuring the stability of our algorithm, we will use the **interleaving distance** d_I, which is a pseudometric on the category of persistence modules arising from n-dimensional filtrations. To define the interleaving distance we must first describe an ε**-interleaving** on persistence modules, introduced in [18]. Suppose that $\mathcal{M}$ and $\mathcal{N}$ are persistence modules over $\mathbb{R}$. Call $\mathcal{M}$ and $\mathcal{N}$ ε-interleaved if there exists two families of homomorphisms $\varphi_a : M_a \to N_{a+\varepsilon}$ and $\psi_a : N_a \to M_{a+\varepsilon}$ which give rise to a commutative diagram satisfying rectangular and triangular commutability. If $\mathcal{M}$ and $\mathcal{N}$ are given by

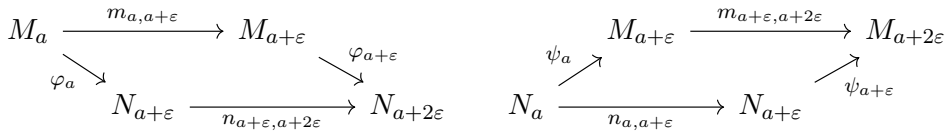
$$\mathcal{M} := \ldots \xrightarrow{m_{a-\varepsilon,a}} M_a \xrightarrow{m_{a,a+\varepsilon}} M_{a+\varepsilon} \xrightarrow{m_{a+\varepsilon,a+2\varepsilon}} M_{a+2\varepsilon} \xrightarrow{m_{a+2\varepsilon,a+3\varepsilon}} \ldots$$

$$\mathcal{N} := \ldots \xrightarrow{n_{a-\varepsilon,a}} N_a \xrightarrow{n_{a,a+\varepsilon}} N_{a+\varepsilon} \xrightarrow{n_{a+\varepsilon,a+2\varepsilon}} N_{a+2\varepsilon} \xrightarrow{n_{a+2\varepsilon,a+3\varepsilon}} \ldots$$

then rectangular commutability is satisfied when

$$n_{a+\varepsilon,a'+\varepsilon} \circ \varphi_a = \varphi_{a'} \circ m_{a,a'} \text{ and } m_{a+\varepsilon,a'+\varepsilon} \circ \psi_a = \psi_{a'} \circ n_{a,a'}$$

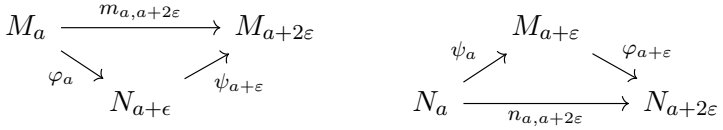
for $a \leq a'$ and $\varepsilon \geq 0$. If such maps exist, then we have the following commutative diagrams on $\mathcal{M}$ and $\mathcal{N}$:



On the other hand, triangular commutability is satisfied when

$$\psi_{a+\varepsilon} \circ \varphi_a = m_{a,a+2\varepsilon} \text{ and } \varphi_{a+\varepsilon} \circ \psi_a = n_{a,a+2\varepsilon}$$

for $a \leq a'$ and $\varepsilon \geq 0$, giving the commutative diagrams:



In the case that such maps exist, then the interleaving distance is the infimal ε such that $\mathcal{M}$ and $\mathcal{N}$ are ε-interleaved. Equivalently, we have

$$d_I = \inf\{\varepsilon : \mathcal{M} \text{ and } \mathcal{N} \text{ are } \varepsilon\text{-interleaved}\}.$$

The interleaving distance will be of use in our case by the *Isometry Theorem* due to [19] and [20], which we state here as a lemma.

Lemma 11. *Given two q-tame persistence modules $\mathcal{M}$ and $\mathcal{N}$ defined over the totally ordered indexing set $\mathbb{R}$,*

$$d_I(\mathcal{M}, \mathcal{N}) = d_B(D(\mathcal{M}), D(\mathcal{N})),$$

where $D(\mathcal{M})$ and $D(\mathcal{N})$ are the persistence diagrams for $\mathcal{M}$ and $\mathcal{N}$.

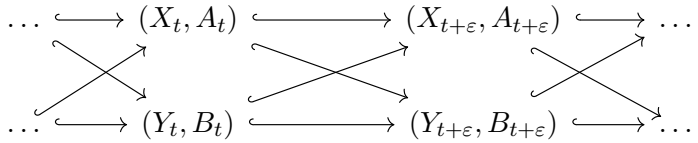
By the *Isometry Theorem*, it is enough to establish an ε-interleaving on persistence modules to imply bottleneck stability. It is worth noting that in our case, although the persistence modules are q-tame, they are not defined over $\mathbb{R}$ but rather some poset $(A, \leq) \subset \mathbb{R}$. This is not an issue, as any persistence module, $\mathcal{M}$, defined over A can be extended over $\mathbb{R}$ as follows: Suppose $a < a'$ where (a, a') is not in A. For any $a \leq b < b' < a'$, assume that $m_{b,b'}$ is an isomorphism and that $\lim_{a \to -\infty} M_a = 0$.

2 Stability of the U-Match PRH Algorithm

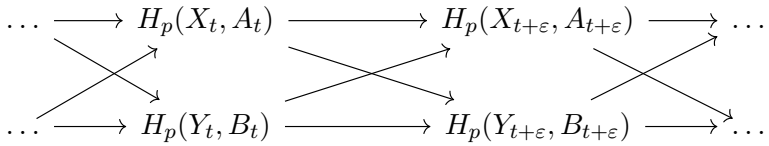
We conclude this chapter with a theorem which guarantees stability of the U-Match PRH algorithm. For this argument, we consider general topological spaces, and do not assume that they are simplicial complexes.

Theorem 12. *Persistence modules arising from filtered quotient spaces satisfy bottleneck stability.*

Proof. Consider two pairs of topological spaces (X, A) and (Y, B) such that $A \subseteq X, B \subseteq Y, X \subseteq Y$ and $A \subseteq B$. Furthermore, suppose that we define filtration functions for X, Y, A, B which preserve the inclusion relationships. Thus, we have the following inclusions of pairs for any $t \in \mathbb{R}$ and some $\varepsilon \geq 0$:



Applying relative homology, then by Lemma 2 we have the following commutative diagram on persistent relative homology groups:



This relationship holds for any p^{th} homology. The top row of the diagram is the persistence module $\mathcal{M}_{XA}$ for the persistent homology of the filtered quotient space X/A. Similarly we have $\mathcal{M}_{YB}$ on the bottom row. This commutative diagrams establishes an ε-interleaving on $\mathcal{M}_{XA}$ and $\mathcal{M}_{YB}$. The required result follows by Lemma 11, which concludes the proof. □

A proof of this fact can also follow by establishing an ε-interleaving at the filtration level, which implies an ε-interleaving on the associated persistence modules. In this case, the proof is slightly less trivial, as an ε-interleaving on a space (resp. simplicial) filtration must be established via a family of homotopic (resp. contiguous) maps which satisfy rectangular and triangular commutability. In our proof, we do assume that the persistence modules are indexed over $\mathbb{R}$. As previously mentioned, any persistence module indexed over some poset $(A, \leq) \subset \mathbb{R}$ can be extended over $\mathbb{R}$ as described in the previous section.

5. Conclusion

We have provided an algorithm for computing persistent relative homology with an eye toward applications in TDA. In fact, since the U-Match PRH algorithm uses two U-Match Decompositions and two sorting steps, then its asymptotic complexity is no different than that of the standard persistence computation. We have demonstrated the correctness of this algorithm using only elementary linear algebra, properties of the U-Match Decomposition and basic definitions of algebraic topology. Furthermore, by appealing to the theory of ε-interleavings of persistence modules, we have argued that persistence modules arising from filtered quotient spaces satisfy bottleneck stability, thus implying the stability of the U-Match PRH algorithm. Finally, it is important to address that this text does not include an implementation. Future work would be primarily concerned with applying this method to real data.

Bibliography

[1] H. Edelsbrunner, D. Letscher, and A. Zomorodian, "Topological persistence and simplification," *Discrete and Computational Geometry*, pp. 511–533, 2002.

[2] A. J. Zomorodian, *Computing and comprehending topology: Peresistence and hierarchical morse complexes*. University of Illinois at Urbana-Champaign, 2001.

[3] G. Carlsson and A. Zomorodian, "Computing persistent homology," *In Proceedings of the twentieth annual symposium on Computational geometry*, pp. 347–356, 2004.

[4] D. Cohen-Steiner, H. Edelsbrunner, and D. Morozov, "Vines and vineyards by updating persistence in linear time," in *Proceedings of the Twenty-Second Annual Symposium on Computational Geometry*, p. 119–126, Association for Computing Machinery, 2006.

[5] H. Edelsbrunner and J. L. Harer, *Computational topology: an introduction*. American Mathematical Society, 2022.

[6] C. Chen and M. Kerber, "An output sensitive algorithm for persistent homology," *In Proceedings of the twenty-seventh annual symposium on Computational geometry*, pp. 207–216, 2011.

[7] G. Carlsson, A. Zomorodian, A. Collins, and L. Guibas, "Persistence barcodes for shapes," *In Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium of Geometry processing*, pp. 124–135, 2004.

[8] H. Hang, C. Giusti, L. Ziegelmeier, and G. Henselman-Petrusek, "U-match factorization: sparse homological algebra, lazy cycle representatives, and dualities in persistent (co)homology," 2021.

[9] T. K. Dey and Y. Wang, *Computational topology for data analysis*, ch. 3-4. Cambridge University Press, 2022.

[10] R. Ghrist, "Barcodes: the persistent topology of data," *Bulletin of the American Mathematical Society*, pp. 61–75, 2008.

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

[12] M. D. Crossley, *Essential Topology*. Springer Science and Business Media, 2006.

[13] D. Morozov, *Homological illusions of persistence and stability*, ch. 6. Duke University, 2008.

[14] M. Kerber and M. Söls, "The localized union-of-balls bifiltration," 2023.

[15] M. Robinson, C. Capraro, C. Joslyn, E. Purvine, B. Praggastis, S. Ranshous, and A. Sathanur, "Local homology of abstract simplicial complexes," 2018.

[16] M. Robinson and J. Palladino, "Protocol-independent critical node detection," *arXiv preprint arXiv:1607.06022*, 2016.

[17] D. Cohen-Steiner, H. Edelsbrunner, and J. H. Harer, "Stability of persistence diagrams," *In Proceedings of the twenty-first annual symposium on Computational Geoemtry*, pp. 263–271, 2005.

[18] F. Chazal, D. Cohen-Steiner, M. Glisse, L. J. Guibas, and S. Y. Oudot, "Proximity of persistence modules and their diagrams," in *Proceedings of the Twenty-Fifth Annual Symposium on Computational Geometry*, p. 237–246, Association for Computing Machinery, 2009.

[19] M. Lesnick, "The theory of the interleaving distance on multidimensional persistence modules," *Foundations of Computational Mathematics*, pp. 613–650, 2015.

[20] U. Bauer and M. Lesnick, "Induced matchings of barcodes and the algebraic stability theorem," *In Proceedings of the thirtieth annual symposium on Computational geometry*, pp. 355–364, 2014.