Computing Persistent Homology [*]

Afra Zomorodian[†] and Gunnar Carlsson[‡]

(Discrete and Computational Geometry)

Abstract

We show that the persistent homology of a filtered d-dimensional simplicial complex is simply the standard homology of a particular graded module over a polynomial ring. Our analysis establishes the existence of a simple description of persistent homology groups over arbitrary fields. It also enables us to derive a natural algorithm for computing persistent homology of spaces in arbitrary dimension over any field. This result generalizes and extends the previously known algorithm that was restricted to subcomplexes of $\mathbb{S}^3$ and $\mathbb{Z}_2$ coefficients. Finally, our study implies the lack of a simple classification over non-fields. Instead, we give an algorithm for computing individual persistent homology groups over an arbitrary principal ideal domains in any dimension.

1 Introduction

In this paper, we study the homology of a filtered d-dimensional simplicial complex K, allowing an arbitrary principal ideal domain D as the ground ring of coefficients. A filtered complex is an increasing sequence of simplicial complexes, as shown in Figure 1. It determines an *inductive system* of homology groups, i.e., a family of Abelian groups $\{G_i\}_{i\geq 0}$ together with homomorphisms $G_i \rightarrow G_{i+1}$. If the homology is computed with field coefficients, we obtain an inductive system of vector spaces over the field. Each vector space is determined up to isomorphism by its dimension. In this paper we obtain a simple classification of an inductive system of vector spaces. Our classification is in terms of a set of

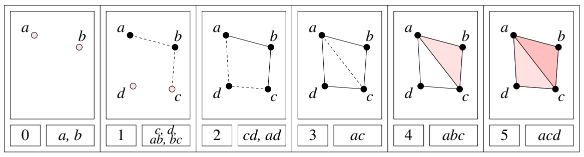


Figure 1. A filtered complex with newly added simplices highlighted.

intervals. We also derive a natural algorithm for computing this family of intervals. Using this family, we may identify homological features that persist within the filtration, the *persistent homology* of the filtered complex.

Furthermore, our interpretation makes it clear that if the ground ring is not a field, there exists no similarly simple classification of persistent homology. Rather, the structures are very complicated, and although we may assign interesting invariants to them, no simple classification is, or is likely ever to be, available. In this case we provide an algorithm for computing a single persistent group for the filtration.

In the rest of this section we first motivate our study through three examples in which filtered complexes arise whose persistent homology is of interest. We then discuss prior work and its relationship to our work. We conclude this section with an outline of the paper.

1.1 Motivation

We call a filtered simplicial complex, along with its associated chain and boundary maps, a *persistence complex*. We formalize this concept in Section 3. Persistence complexes arise naturally whenever one is attempting to study topological invariants of a space computationally. Often, our knowledge of this space is limited and imprecise. Consequently, we must utilize a multiscale approach to capture the connectivity of the space, giving us a persistence complex.

[*]Research by the first author is partially supported by NSF under grants CCR-00-86013 and ITR-0086013. Research by the second author is partially supported by NSF under grant DMS-0101364. Research by both authors is partially supported by NSF under grant DMS-0138456.

[†]Department of Computer Science, Stanford University, Stanford, California.

[‡]Department of Mathematics, Stanford University, Stanford, California.

Example 1.1 (point cloud data) Suppose we are given a finite set of points X from a subspace $\mathbb{X} \in \mathbb{R}^n$. We call X *point cloud data* or *PCD* for short. It is reasonable to believe that if the sampling is dense enough, we should be able to compute the topological invariants of $\mathbb{X}$ directly from the PCD. To do so, we may either compute the *Čech* complex, or approximate it via a *Rips* complex [15]. The latter complex $R_\epsilon(X)$ has X as its vertex set. We declare a set of vertices $\sigma = \{x_0, x_1, \ldots, x_k\}$ to span a k-simplex of $R_\epsilon(X)$ iff the vertices are pairwise close, that is, $d(x_i, x_j) \leq \epsilon$ for all pairs $x_i, x_j \in \sigma$. There is an obvious inclusion $R_\epsilon(X) \to R_{\epsilon'}(X)$ whenever $\epsilon < \epsilon'$. In other words, for any increasing sequence of non-negative real numbers, we obtain a persistence complex.

Example 1.2 (density) Often, our samples are not from a geometric object, but are heavily concentrated on it. It is important, therefore, to compute a measure of density of the data around each sample. For instance, we may count the number of samples $\rho(x)$ contained in a ball of size ϵ around each sample x. We may then define $R^r_\epsilon(X) \subseteq R_\epsilon$ to be the Rips subcomplex on the vertices for which $\rho(x) \leq r$. Again, for any increasing sequence of non-negative real numbers r, we obtain a persistence complex. We must analyze this complex to compute topological invariants attached to the geometric object around which our data is concentrated.

Example 1.3 (Morse functions) Given a manifold M equipped with a Morse function f, we may filter M via the *excursion sets* $M_r = \{m \in M \mid f(m) \leq r\}$. We again choose an increasing sequence of non-negative numbers to get a persistence complex. If the Morse function is a height function attached to some embedding of M in $\mathbb{R}^n$, persistent homology can now give information about the shape of the submanifolds, as well the homological invariants of the total manifold.

1.2 Prior Work

We assume familiarity with basic group theory and refer the reader to Dummit and Foote [10] for an introduction. We make extensive use of Munkres [16] in our description of algebraic homology and recommend it as an accessible resource to non-specialists. There is a large body of work on the efficient computation of homology groups and their ranks [1, 8, 9, 13].

Persistent homology groups are initially defined in [11, 18]. The authors also provide an algorithm that worked only for spaces that were subcomplexes of $\mathbb{S}^3$ over $\mathbb{Z}_2$ coefficients. The algorithm generates a set of intervals for a filtered complex. Surprisingly, the authors show that these intervals allowed the correct computation of the rank of persistent homology groups. In other words, the authors prove constructively that persistent homology groups of subcomplexes of $\mathbb{S}^3$, if computed over $\mathbb{Z}_2$ coefficients, have a simple description in terms of a set of intervals. To build these intervals, the algorithm pairs *positive* cycle-creating simplices with *negative* cycle-destroying simplices. During the computation, the algorithm ignores negative simplices and always looks for the *youngest* simplex. While the authors prove the correctness of the results of the algorithm, the underlying structure remains hidden.

1.3 Our Work

We are motivated primarily by the unexplained results of the previous work. We wish to answer the following questions:

1. Why does a simple description exist for persistent homology of subcomplexes of $\mathbb{S}^3$ over $\mathbb{Z}_2$?

2. Does this description also exist over other rings of coefficients and arbitrary-dimensional simplicial complexes?

3. Why can we ignore negative simplices during computation?

4. Why do we always look for the youngest simplex?

5. What is the relationship between the persistence algorithm and the standard reduction scheme?

In this paper we resolve all these questions by uncovering and elucidating the structure of persistent homology. Specifically, we show that the persistent homology of a filtered d-dimensional simplicial complex is simply the standard homology of a particular graded module over a polynomial ring. Our analysis places persistent homology within the classical framework of algebraic topology. This placement allows us to utilize a standard structure theorem to establish the existence of a simple description of persistent homology groups as a set of intervals, answering the first question above. This description exists over arbitrary fields, not just $\mathbb{Z}_2$ as in the previous result, resolving the second question.

Our analysis also enables us to derive a persistence algorithm from the standard reduction scheme in algebra, resolving the next three questions using two main lemmas. Our algorithm generalizes and extends the previously known algorithm to complexes in arbitrary dimensions over arbitrary fields of coefficients. We also show that if we consider integer coefficients or coefficients in some non-field R, there is no similar simple classification. This negative result suggests the possibility of interesting yet incomplete invariants of inductive systems. For now, we give an algorithm for classifying a single homology group over an arbitrary principal ideal domain.

1.4 Spectral Sequences

Any filtered complex gives rise to a *spectral sequence*, so it is natural to wonder about the relationship between this sequence and persistence. A full discussion on spectral sequences is outside the scope of this paper. However, we include a few remarks here for the reader who is familiar with the subject. We may easily show that the persistence intervals for a filtration correspond to nontrivial differentials in the spectral sequence that arises from the filtration. Specifically, an interval of length r corresponds to some differential d_{r+1}. Given this correspondence, we realize that the method of spectral sequences computes persistence intervals in order of length, finding all intervals of length r during the computation of the E^{r+1} term. In principle, we may use this method to compute the result of our algorithm. However, the method does not provide an *algorithm*, but a *scheme* that must be tailored for each problem independently. The practitioner must decide on an appropriate basis, find the zero terms in the sequence, and deduce the nature of the differentials. Our analysis of persistent homology, on the other hand, provides a complete, effective, and implementable algorithm for any filtered complex.

1.5 Outline

We begin by reviewing concepts from algebra and simplicial homology in Section 2. We also re-introduce persistent homology over integers and arbitrary dimensions. In Section 3 we define and study the persistence module, a structure that represents the homology of a filtered complex. In addition, we establish a relationship between our results and prior work. Using our analysis, we derive an algorithm for computation over fields in Section 4. For non-fields, we describe an algorithm in Section 5 that computes individual persistent groups. Section 6 describes our implementation and some experiments. We conclude the paper in Section 7 with a discussion of current and future work.

2 Background

In this section we review the mathematical and algorithmic background necessary for our work. We begin by reviewing the structure of finitely generated modules over principal ideal domains. We then discuss simplicial complexes and their associated chain complexes. Putting these concepts together, we define simplicial homology and outline the standard algorithm for its computation. We conclude this section by describing persistent homology.

2.1 Algebra

Throughout this paper we assume a ring R to be commutative with unity. A *polynomial $f(t)$ with coefficients in R* is a formal sum $\sum_{i=0}^{\infty} a_i t^i$, where $a_i \in R$ and t is the indeterminate. For example, $2t + 3$ and $t^7 - 5t^2$ are both polynomials with integer coefficients. The set of all polynomials $f(t)$ over R forms a commutative ring $R[t]$ with unity. If R has no divisors of zero, and all its ideals are principal, it is a *principal ideal domain (PID)*. For our purposes, a PID is simply a ring in which we may compute the *greatest common divisor* or *gcd* of a pair of elements. This is the key operation needed by the structure theorem that we discuss below. PIDs include the familiar rings $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$. Finite fields $\mathbb{Z}_p$ for p a prime, as well as $F[t]$, polynomials with coefficients from a field F, are also PIDs and have effective algorithms for computing the gcd [6].

A *graded ring* is a ring $\langle R, +, \cdot \rangle$ equipped with a direct sum decomposition of Abelian groups $R \cong \bigoplus_i R_i$, $i \in \mathbb{Z}$, so that multiplication is defined by bilinear pairings $R_n \otimes R_m \to R_{n+m}$. Elements in a single R_i are called *homogeneous* and have *degree* i, $\deg e = i$ for all $e \in R_i$. We may grade the polynomial ring $R[t]$ non-negatively with the *standard grading* $(t^n) = t^n \cdot R[t], n \geq 0$. In this grading, $2t^6$ and $7t^3$ are both homogeneous of degree 6 and 3, respectively, but their sum $2t^6 + 7t^3$ is not homogeneous. The product of the two terms, $14t^9$, has degree 9 as required by the definition. A *graded module* M over a graded ring R is a module equipped with a direct sum decomposition, $M \cong \bigoplus_i M_i$, $i \in \mathbb{Z}$, so that the action of R on M is defined by bilinear pairings $R_n \otimes M_m \to M_{n+m}$. The main structure in our paper is a graded module and we include concrete examples that clarify this concept later on. A graded ring (module) is *non-negatively graded* if $R_i = 0$ ($M_i = 0$) for all $i < 0$.

The standard structure theorem describes finitely generated modules and graded modules over PIDs.

Theorem 2.1 (structure) If D is a PID, then every finitely generated D-module is isomorphic to a direct sum of cyclic D-modules. That is, it decomposes uniquely into the form

$$D^\beta \oplus \left(\bigoplus_{i=1}^{m} D/d_i D \right), \qquad (1)$$

for $d_i \in D, \beta \in \mathbb{Z}$, such that $d_i | d_{i+1}$. Similarly, every graded module M over a graded PID D decomposes uniquely into the form

$$\left(\bigoplus_{i=1}^{n} \Sigma^{\alpha_i} D \right) \oplus \left(\bigoplus_{j=1}^{m} \Sigma^{\gamma_j} D/d_j D \right), \qquad (2)$$

where $d_j \in D$ are homogeneous elements so that $d_j | d_{j+1}$, $\alpha_i, \gamma_j \in \mathbb{Z}$, and Σ^α denotes an α-shift upward in grading.

In both cases, the theorem decomposes the structures into two parts. The *free* portion on the left includes generators that may generate an infinite number of elements. This portion is a vector space and should be familiar to most readers. Decomposition (1) has a vector space of dimension β. The *torsional* portion on the right includes generators that may generate a finite number of elements. For example, if PID D is $\mathbb{Z}$ in the theorem, $\mathbb{Z}/3\mathbb{Z} = \mathbb{Z}_3$ would represent a generator capable of only creating three elements. These torsional elements are also homogeneous. Intuitively then, the theorem describes finitely generated modules and graded modules as structures that look like vector spaces but also have some dimensions that are "finite" in size.

2.2 Simplicial Complexes

A *simplicial complex* is a set K, together with a collection $\mathcal{S}$ of subsets of K called *simplices* (singular *simplex*) such that for all $v \in K, \{v\} \in \mathcal{S}$, and if $\tau \subseteq \sigma \in \mathcal{S}$, then $\tau \in \mathcal{S}$. We call the sets $\{v\}$ the *vertices* of K. When it is clear from context what $\mathcal{S}$ is, we refer to set K as a complex. We say $\sigma \in \mathcal{S}$ is a k-*simplex* of *dimension* k if $|\sigma| = k + 1$. If $\tau \subseteq \sigma$, τ is a *face* of σ, and σ is a *coface* of τ. An *orientation* of a k-simplex σ, $\sigma = \{v_0, \ldots, v_k\}$, is an equivalence class of orderings of the vertices of σ, where $(v_0, \ldots, v_k) \sim (v_{\tau(0)}, \ldots, v_{\tau(k)})$ are equivalent if the sign of τ is 1. We denote an *oriented simplex* by $[\sigma]$. A simplex may be realized geometrically as the convex hull of $k + 1$ affinely independent points in $\mathbb{R}^d, d \geq k$. A realization gives us the familiar low-dimensional k-simplices: *vertices*, *edges*, *triangles*, and *tetrahedra*, for $0 \leq k \leq 3$, shown in Figure 2. Within a realized complex, the simplices must meet along common faces. A *subcomplex* of K is a subset $L \subseteq K$ that is also a simplicial complex. A *filtration* of a complex K is a nested subsequence of complexes $\emptyset = K^0 \subseteq K^1 \subseteq \ldots \subseteq K^m = K$. For generality, we let $K^i = K^m$ for all $i \geq m$. We call K a *filtered complex*. We show a small filtered complex in Figure 1.

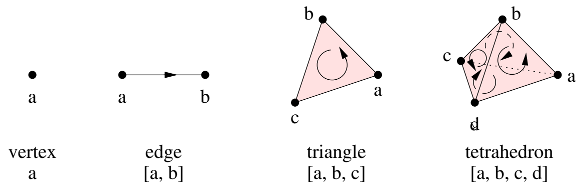


Figure 2. Oriented k-simplices in $\mathbb{R}^3$, $0 \leq k \leq 3$. The orientation on the tetrahedron is shown on its faces.

2.3 Chain Complex

The *kth chain group* C_k *of* K is the free Abelian group on its set of oriented k-simplices, where $[\sigma] = -[\tau]$ if $\sigma = \tau$ and σ and τ are differently oriented.. An element $c \in \mathsf{C}_k$ is a k-chain, $c = \sum_i n_i[\sigma_i]$, $\sigma_i \in K$ with *coefficients* $n_i \in \mathbb{Z}$. The *boundary operator* $\partial_k : \mathsf{C}_k \to \mathsf{C}_{k-1}$ is a homomorphism defined linearly on a chain c by its action on any simplex $\sigma = [v_0, v_1, \ldots, v_k] \in c$,

$$\partial_k \sigma = \sum_i (-1)^i [v_0, v_1, \ldots, \hat{v}_i, \ldots, v_k],$$

where $\hat{v}_i$ indicates that v_i is deleted from the sequence. The boundary operator connects the chain groups into a *chain complex* C_*:

$$\cdots \to \mathsf{C}_{k+1} \xrightarrow{\partial_{k+1}} \mathsf{C}_k \xrightarrow{\partial_k} \mathsf{C}_{k-1} \to \cdots.$$

We may also define subgroups of C_k using the boundary operator: the *cycle group* $\mathsf{Z}_k = \ker \partial_k$ and the *boundary group* $\mathsf{B}_k = \operatorname{im} \partial_{k+1}$. We show examples of cycles in Figure 3. An important property of the boundary operators is that the boundary of a boundary is always empty, $\partial_k \partial_{k+1} = 0$. This fact, along with the definitions, implies that the defined subgroups are nested, $\mathsf{B}_k \subseteq \mathsf{Z}_k \subseteq \mathsf{C}_k$, as in Figure 4. For generality, we often define null boundary operators in dimensions where C_k is empty.

2.4 Homology

The *kth homology group* is $\mathsf{H}_k = \mathsf{Z}_k/\mathsf{B}_k$. Its elements are classes of *homologous* cycles. To describe its structure, we view the Abelian groups we have defined so far

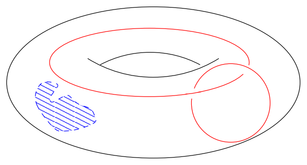


Figure 3. The dashed 1-boundary rests on the surface of a torus. The two solid 1-cycles form a basis for the first homology class of the torus. These cycles are non-bounding: neither is a boundary of a piece of surface.

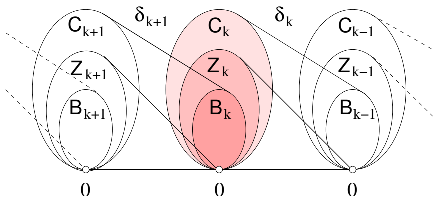


Figure 4. A chain complex with its internals: chain, cycle, and boundary groups, and their images under the boundary operators.

as modules over the integers. This view allows alternate ground rings of coefficients, including fields. If the ring is a PID D, H_k is a D-module and Theorem (2.1) applies: β, the rank of the free submodule, is the *Betti number* of the module, and d_i are its *torsion coefficients*. When the ground ring is $\mathbb{Z}$, the theorem above describes the structure of finitely generated Abelian groups. Over a field, such as $\mathbb{R}$, $\mathbb{Q}$, or $\mathbb{Z}_p$ for p a prime, the torsion submodule disappears. The module is a vector space that is fully described by a single integer, its rank β, which depends on the chosen field.

2.5 Reduction

The standard method for computing homology is the reduction algorithm. We describe this method for integer coefficients as it is the more familiar ring. The method extends to modules over arbitrary PIDs, however.

As C_k is free, the oriented k-simplices form the *standard basis* for it. We represent the boundary operator $\partial_k \colon \mathsf{C}_k \to \mathsf{C}_{k-1}$ relative to the standard bases of the chain groups as an integer matrix M_k with entries in $\{-1, 0, 1\}$. The matrix M_k is called the *standard matrix representation* of ∂_k. It has m_k columns and m_{k-1} rows (the number of k- and $(k-1)$-simplices, respectively). The null-space of M_k corresponds to Z_k and its range-space to B_{k-1}, as manifested in Figure 4. The *reduction algorithm* derives alternate bases for the chain groups, relative to which the matrix for ∂_k is diagonal. The algorithm utilizes the following *elementary row operations* on M_k:

1. exchange row i and row j,
2. multiply row i by -1,
3. replace row i by (row i) + q(row j), where q is an integer and $j \neq i$.

The algorithm also uses *elementary column operations* that are similarly defined. Each column (row) operation corresponds to a change in the basis for C_k (C_{k-1}). For example, if e_i and e_j are the ith and jth basis elements for C_k, respectively, a column operation of type (3) amounts to replacing e_i with $e_i + qe_j$. A similar row operation on basis elements $\hat{e}_i$ and $\hat{e}_j$ for C_{k-1}, however, replaces $\hat{e}_j$ by $\hat{e}_j - q\hat{e}_i$. We shall make use of this fact in Section 4. The algorithm systematically modifies the bases of C_k and C_{k-1} using elementary operations to reduce M_k to its *(Smith) normal form*:

$$\tilde{M}_k = \left[\begin{array}{ccc|c} b_1 & & 0 & \\ & \ddots & & 0 \\ 0 & & b_{l_k} & \\ \hline & 0 & & 0 \end{array}\right],$$

where $l_k = \operatorname{rank} M_k = \operatorname{rank} \tilde{M}_k$, $b^i \geq 1$, and $b_i | b_{i+1}$ for all $1 \leq i < l_k$. The algorithm can also compute corresponding bases $\{e_j\}$ and $\{\hat{e}_i\}$ for C_k and C_{k-1}, respectively, although this is unnecessary if a decomposition is all that is needed. Computing the normal form in all dimensions, we get a full characterization of H_k:

(i) the torsion coefficients of H_{k-1} (d_i in (1)) are precisely the diagonal entries b_i greater than one.

(ii) $\{e_i \mid l_k + 1 \leq i \leq m_k\}$ is a basis for Z_k. Therefore, $\operatorname{rank} \mathsf{Z}_k = m_k - l_k$.

(iii) $\{b_i \hat{e}_i \mid 1 \leq i \leq l_k\}$ is a basis for B_{k-1}. Equivalently, $\operatorname{rank} \mathsf{B}_k = \operatorname{rank} M_{k+1} = l_{k+1}$.

Combining (ii) and (iii), we have

$$\beta_k = \operatorname{rank} \mathsf{Z}_k - \operatorname{rank} \mathsf{B}_k = m_k - l_k - l_{k+1}. \quad (3)$$

Example 2.1 For the complex in Figure 1, the standard matrix representation of ∂_1 is

$$M_1 = \left[\begin{array}{c|ccccc} & ab & bc & cd & ad & ac \\ \hline a & -1 & 0 & 0 & -1 & -1 \\ b & 1 & -1 & 0 & 0 & 0 \\ c & 0 & 1 & -1 & 0 & 1 \\ d & 0 & 0 & 1 & 1 & 0 \end{array}\right],$$

where we show the bases within the matrix. Reducing the matrix, we get the normal form

$$\tilde{M}_1 = \left[\begin{array}{c|ccccc} & cd & bc & ab & z_1 & z_2 \\ \hline d-c & 1 & 0 & 0 & 0 & 0 \\ c-b & 0 & 1 & 0 & 0 & 0 \\ b-a & 0 & 0 & 1 & 0 & 0 \\ a & 0 & 0 & 0 & 0 & 0 \end{array}\right],$$

where $z_1 = ad - bc - cd - ab$ and $z_2 = ac - bc - ab$ form a basis for Z_1 and $\{d - c, c - b, b - a\}$ is a basis for B_0.

We may use a similar procedure to compute homology over graded PIDs. A *homogeneous basis* is a basis of homogeneous elements. We begin by representing ∂_k relative to the standard basis of C_k (which is homogeneous) and a homogeneous basis for Z_{k-1}. Reducing to normal form, we read off the description provided by direct sum (2) using the new basis $\{\hat{e}_j\}$ for Z_{k-1}:

(i) zero row i contributes a free term with shift $\alpha_i = \deg \hat{e}_i$,

(ii) row with diagonal term b_i contributes a torsional term with homogeneous $d_j = b_j$ and shift $\gamma_j = \deg \hat{e}_j$.

The reduction algorithm requires $O(m^3)$ elementary operations, where m is the number of simplices in K. The operations, however, must be performed in exact integer arithmetic. This is problematic in practice, as the entries of the intermediate matrices may become extremely large.

2.6 Persistence

We end this section with by re-introducing persistence. Given a filtered complex, the ith complex K^i has associated boundary operators ∂_k^i, matrices M_k^i, and groups $\mathsf{C}_k^i, \mathsf{Z}_k^i, \mathsf{B}_k^i$ and H_k^i for all $i, k \geq 0$. Note that superscripts indicate the filtration index and are not related to cohomology. The *p-persistent kth homology group of K^i* is

$$\mathsf{H}_k^{i,p} \quad = \quad \mathsf{Z}_k^i \,/\, (\mathsf{B}_k^{i+p} \cap \mathsf{Z}_k^i). \tag{4}$$

The definition is well-defined: both groups in the denominator are subgroups of C_k^{l+p}, so their intersection is also a group, a subgroup of the numerator. The *p-persistent kth Betti number of K^i* is $\beta_k^{i,p}$, the rank of the free subgroup of $\mathsf{H}_k^{i,p}$. We may also define persistent homology groups using the injection $\eta_k^{i,p}\colon \mathsf{H}_k^i \to \mathsf{H}_k^{i+p}$, that maps a homology class into the one that contains it. Then, $\operatorname{im} \eta_k^{i,p} \simeq \mathsf{H}_k^{i,p}$ [11, 18]. We extend this definition over arbitrary PIDs, as before. Persistent homology groups are modules and Theorem 2.1 describes their structure.

3 The Persistence Module

In this section we take a different view of persistent homology in order to understand its structure. Intuitively, the computation of persistence requires compatible bases for H_k^i and H_k^{i+p}. It is not clear when a succinct description is available for the compatible bases. We begin this section by combining the homology of all the complexes in the filtration into a single algebraic structure. We then establish a correspondence that reveals a simple description over fields. Most significantly, we illustrate that the persistent homology of a filtered complex is simply the standard homology of a particular graded module over a polynomial ring. A simple application of the structure theorem (Theorem 2.1) gives us the needed description. We end this section by illustrating the relationship of our structures to the persistence equation (Equation (4).)

Definition 3.1 (persistence complex) A *persistence complex* $\mathcal{C}$ is a family of chain complexes $\{\mathsf{C}_*^i\}_{i\geq 0}$ over R, together with chain map's $f^i\colon \mathsf{C}_*^i \to \mathsf{C}_*^{i+1}$, so that we have the following diagram:

$$\mathsf{C}_*^0 \xrightarrow{f^0} \mathsf{C}_*^1 \xrightarrow{f^1} \mathsf{C}_*^2 \xrightarrow{f^2} \cdots .$$

Our filtered complex K with inclusion maps for the simplices becomes a persistence complex. Below, we show a portion of a persistence complex, with the chain complexes expanded. The filtration index increases horizontally to the right under the chain maps f^i, and the dimension decreases vertically to the bottom under the boundary operators ∂_k.

$$\begin{tikzcd}
{} \arrow[d, "\partial_3"'] & {} \arrow[d, "\partial_3"'] & {} \arrow[d, "\partial_3"'] & \\
\mathsf{C}_2^0 \arrow[r, "f^0"] \arrow[d, "\partial_2"'] & \mathsf{C}_2^1 \arrow[r, "f^1"] \arrow[d, "\partial_2"'] & \mathsf{C}_2^2 \arrow[r, "f^2"] \arrow[d, "\partial_2"'] & \cdots \\
\mathsf{C}_1^0 \arrow[r, "f^0"] \arrow[d, "\partial_1"'] & \mathsf{C}_1^1 \arrow[r, "f^1"] \arrow[d, "\partial_1"'] & \mathsf{C}_1^2 \arrow[r, "f^2"] \arrow[d, "\partial_1"'] & \cdots \\
\mathsf{C}_0^0 \arrow[r, "f^0"] & \mathsf{C}_0^1 \arrow[r, "f^1"] & \mathsf{C}_0^2 \arrow[r, "f^2"] & \cdots
\end{tikzcd}$$

Definition 3.2 (persistence module) A *persistence module* $\mathcal{M}$ is a family of R-modules M^i, together with homomorphisms $\varphi^i\colon M^i \to M^{i+1}$.

For example, the homology of a persistence complex is a persistence module, where φ^i simply maps a homology class to the one that contains it.

Definition 3.3 (finite type) A persistence complex $\{\mathsf{C}_*^i, f^i\}$ (persistence module $\{M^i, \varphi^i\}$) is of *finite type* if each component complex (module) is a finitely generated R-module, and if the maps f^i (φ^i, respectively) are isomorphisms for $i \geq m$ for some integer m.

As our complex K is finite, it generates a persistence complex $\mathcal{C}$ of finite type, whose homology is a persistence module $\mathcal{M}$ of finite type. We showed in the Introduction how such complexes arise in practice.

3.1 Correspondence

Suppose we have a persistence module $\mathcal{M} = \{M^i, \varphi^i\}_{i \geq 0}$ over ring R. We now equip $R[t]$ with the standard grading and define a graded module over $R[t]$ by

$$\alpha(\mathcal{M}) = \bigoplus_{i=0}^{\infty} M^i,$$

where the R-module structure is simply the sum of the structures on the individual components, and where the action of t is given by

$$t \cdot (m^0, m^1, m^2, \ldots) = (0, \varphi^0(m^0), \varphi^1(m^1), \varphi^2(m^2), \ldots).$$

That is, t simply shifts elements of the module up in the gradation.

Theorem 3.1 (correspondence) The correspondence α defines an equivalence of categories between the category of persistence modules of finite type over R and the category of finitely generated non-negatively graded modules over $R[t]$.

The proof is the Artin-Rees theory in commutative algebra [12].

Intuitively, we are building a single structure that contains all the complexes in the filtration. We begin by computing a direct sum of the complexes, arriving at a much larger space that is graded according to the filtration ordering. We then remember the time each simplex enters using a polynomial coefficient. For instance, simplex a enters the filtration in Figure 1 at time 0. To shift this simplex along the grading, we must multiply the simplex using t. Therefore, while a exists at time 0, $t \cdot a$ exists at time 1, $t^2 \cdot a$ at time 2, and so on. The key idea is that the filtration ordering is encoded in the coefficient polynomial ring. We utilize these coefficients in Section 4 to derive the persistence algorithm from the reduction scheme in Section 2.5.

3.2 Decomposition

The correspondence established by Theorem 3.1 suggests the non-existence of simple classifications of persistence modules over a ground ring that is not a field, such as $\mathbb{Z}$. It is well known in commutative algebra that the classification of modules over $\mathbb{Z}[t]$ is extremely complicated. While it is possible to assign interesting invariants to $\mathbb{Z}[t]$-modules, a simple classification is not available, nor is it ever likely to be available.

On the other hand, the correspondence gives us a simple decomposition when the ground ring is a field F. Here, the graded ring $F[t]$ is a PID and its only graded ideals are homogeneous of form (t^n), so the structure of the $F[t]$-module is described by sum (2) in Theorem 2.1:

$$\left(\bigoplus_{i=1}^{n} \Sigma^{\alpha_i} F[t] \right) \oplus \left(\bigoplus_{j=1}^{m} \Sigma^{\gamma_j} F[t]/(t^{n_j}) \right). \quad (5)$$

We wish to parametrize the isomorphism classes of $F[t]$-modules by suitable objects.

Definition 3.4 ($\mathcal{P}$-interval) A $\mathcal{P}$-interval is an ordered pair (i, j) with $0 \leq i < j \in \mathbb{Z}^{\infty} = \mathbb{Z} \cup \{+\infty\}$.

We associate a graded $F[t]$-module to a set $\mathcal{S}$ of $\mathcal{P}$-intervals via a bijection Q. We define $Q(i, j) = \Sigma^i F[t]/(t^{j-i})$ for $\mathcal{P}$-interval (i, j). Of course, $Q(i, +\infty) = \Sigma^i F[t]$. For a set of $\mathcal{P}$-intervals $\mathcal{S} = \{(i_1, j_1), (i_2, j_2) \ldots, (i_n, j_n)\}$, we define

$$Q(\mathcal{S}) = \bigoplus_{l=1}^{n} Q(i_l, j_l).$$

Our correspondence may now be restated as follows.

Corollary 3.1 The correspondence $\mathcal{S} \to Q(\mathcal{S})$ defines a bijection between the finite sets of $\mathcal{P}$-intervals and the finitely generated graded modules over the graded ring $F[t]$. Consequently, the isomorphism classes of persistence modules of finite type over F are in bijective correspondence with the finite sets of $\mathcal{P}$-intervals.

3.3 Interpretation

Before proceeding any further, we recap our work so far and relate it to prior results. Recall that our input is a filtered complex K and we are interested in its kth homology. In each dimension the homology of complex K^i becomes a vector space over a field, described fully by its rank β_k^i. We need to choose compatible bases across the filtration in order to compute persistent homology for the entire filtration. So, we form the persistence module corresponding to K, a direct sum of these vector spaces. The structure theorem states that a basis exists for this module that provides compatible bases for all the vector spaces. In particular, each $\mathcal{P}$-interval (i, j) describes a basis element for the homology vector spaces starting at time i until time $j - 1$. This element is a k-cycle e that is completed at time i, forming a new homology class. It also remains non-bounding until time j, at which time it joins the boundary group B_k^j. Therefore, the $\mathcal{P}$-intervals discussed here are precisely the so-called k-intervals utilized in [11] to describe persistent $\mathbb{Z}_2$-homology. That is, while component homology groups are torsionless, persistence appears as torsional and free elements of the persistence module.

Our interpretation also allows us to ask when $e + \mathsf{B}_k^l$ is a basis element for the persistent groups $\mathsf{H}_k^{l,p}$. Recall Equation (4). As $e \notin \mathsf{B}_k^l$ for all $l < j$, we know that $e \notin \mathsf{B}_k^{l+p}$ for $l + p < j$. Along with $l \geq i$ and $p \geq 0$, the three inequalities define a triangular region in the index-persistence plane, as drawn in Figure 5. The region gives us the values for which the k-cycle e is a basis element for $\mathsf{H}_k^{l,p}$. In other words, we have just shown a direct proof of the *k-triangle Lemma* in [11], which we restate here in a different form.

Lemma 3.1 Let $\mathcal{T}$ be the set of triangles defined by $\mathcal{P}$-intervals for the k-dimensional persistence module. The rank $\beta_k^{l,p}$ of $\mathsf{H}_k^{l,p}$ is the number of triangles in $\mathcal{T}$ containing the point (l, p).

Consequently, computing persistent homology over a field is equivalent to finding the corresponding set of $\mathcal{P}$-intervals.

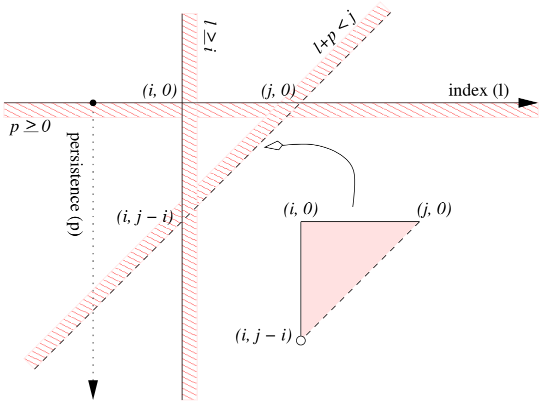


Figure 5. The inequalities $p \geq 0$, $l \geq i$, and $l+p < j$ define a triangular region in the index-persistence plane. This region defines when the cycle is a basis element for the homology vector space.

4 Algorithm for Fields

In this section we devise an algorithm for computing persistent homology over a field. Given the theoretical development of the last section, our approach is rather simple: we simplify the standard reduction algorithm using the properties of the persistence module. Our arguments give an algorithm for computing the $\mathcal{P}$-intervals for a filtered complex directly over the field F, without the need for constructing the persistence module. This algorithm is a generalized version of the pairing algorithm shown in [11].

4.1 Derivation

We use the small filtration in Figure 1 as a running example and compute over $\mathbb{Z}_2$, although any field will do. The persistence module corresponds to a $\mathbb{Z}_2[t]$-module by the correspondence established in Theorem 2.1. Table 1 reviews the degrees of the simplices of our filtration as homogeneous elements of this module.

a	b	c	d	ab	bc	cd	ad	ac	abc	acd
0	0	1	1	1	1	2	2	3	4	5

Table 1. Degree of simplices of filtration in Figure 1

Throughout this section we use $\{e_j\}$ and $\{\hat{e}_i\}$ to represent homogeneous bases for C_k and C_{k-1}, respectively. Relative to homogeneous bases, any representation M_k of ∂_k has the following basic property:

$$\deg \hat{e}_i + \deg M_k(i,j) = \deg e_j, \qquad (6)$$

where $M_k(i,j)$ denotes the element at location (i,j). We get

$$M_1 = \left[\begin{array}{c|ccccc} & ab & bc & cd & ad & ac \\ \hline d & 0 & 0 & t & t & 0 \\ c & 0 & 1 & t & 0 & t^2 \\ b & t & t & 0 & 0 & 0 \\ a & t & 0 & 0 & t^2 & t^3 \end{array}\right], \qquad (7)$$

for ∂_1 in our example. The reader may verify Equation (6) using this example for intuition, e.g. $M_1(4,4) = t^2$ as $\deg ad - \deg a = 2 - 0 = 2$, according to Table 1.

Clearly, the standard bases for chain groups are homogeneous. We need to represent $\partial_k \colon \mathsf{C}_k \to \mathsf{C}_{k-1}$ relative to the standard basis for C_k and a homogeneous basis for Z_{k-1}. We then reduce the matrix and read off the description of H_k according to our discussion in Section 2.5. We compute these representations inductively in dimension. The base case is trivial. As $\partial_0 \equiv 0$, $\mathsf{Z}_0 = \mathsf{C}_0$ and the standard basis may be used for representing ∂_1. Now, assume we have a matrix representation M_k of ∂_k relative to the standard basis $\{e_j\}$ for C_k and a homogeneous basis $\{\hat{e}_i\}$ for Z_{k-1}. For induction, we need to compute a homogeneous basis for Z_k and represent ∂_{k+1} relative to C_{k+1} and the computed basis. We begin by sorting basis $\hat{e}_i$ in reverse degree order, as already done in the matrix in Equation (7). We next transform M_k into the *column-echelon form* $\tilde{M}_k$, a lower staircase form shown in Figure 6 [17]. The steps have variable height, all landings have width equal to one, and non-zero elements may only occur beneath the staircase. A boxed value in the figure is a *pivot* and a row (column) with a pivot is called

$$\left[\begin{array}{cccccc}
\boxed{*} & 0 & & & & 0 \\
 & \boxed{*} & 0 & \cdots & & \\
* & * & 0 & & & \vdots \\
 & * & \boxed{*} & 0 & \cdots & \\
* & & * & 0 & \cdots & 0
\end{array}\right]$$

Figure 6. The column-echelon form. An $*$ indicates a non-zero value and pivots are boxed.

a *pivot row (column)*. From linear algebra, we know that $\operatorname{rank} M_k = \operatorname{rank} \mathsf{B}_{k-1}$ is the number of pivots in an echelon form. The basis elements corresponding to non-pivot columns form the desired basis for Z_k. In our example, we have

$$\tilde{M}_1 = \left[\begin{array}{c|ccccc}
 & cd & bc & ab & z_1 & z_2 \\
\hline
d & \boxed{\mathrm{t}} & 0 & 0 & 0 & 0 \\
c & t & \boxed{1} & 0 & 0 & 0 \\
b & 0 & t & \boxed{\mathrm{t}} & 0 & 0 \\
a & 0 & 0 & t & 0 & 0
\end{array}\right], \quad (8)$$

where $z_1 = ad - cd - t \cdot bc - t \cdot ab$, and $z_2 = ac - t^2 \cdot bc - t^2 \cdot ab$ form a homogeneous basis for Z_1.

The procedure that arrives at the echelon form is Gaussian elimination on the columns, utilizing elementary column operations of types (1, 3) only. Starting with the left-most column, we eliminate non-zero entries occurring in pivot rows in order of increasing row. To eliminate an entry, we use an elementary column operation of type (3) that maintains the homogeneity of the basis and matrix elements. We continue until we either arrive at a zero column, or we find a new pivot. If needed, we then perform a column exchange (type (1)) to reorder the columns appropriately.

Lemma 4.1 (Echelon Form) The pivots in column-echelon form are the same as the diagonal elements in normal form. Moreover, the degree of the basis elements on pivot rows is the same in both forms.

Proof: Because of our sort, the degree of row basis elements $\hat{e}_i$ is monotonically decreasing from the top row down. Within each fixed column j, $\deg e_j$ is a constant c. By Equation (6), $\deg M_k(i,j) = c - \deg \hat{e}_i$. Therefore, the degree of the elements in each column is monotonically increasing with row. We may eliminate non-zero elements below pivots using row operations that do not change the pivot elements or the degrees of the row basis elements. We then place the matrix in diagonal form with row and column swaps. □

The lemma states that if we are only interested in the degree of the basis elements, we may read them off from the echelon form directly. That is, we may use the following corollary of the standard structure theorem to obtain the description.

Corollary 4.1 Let $\tilde{M}_k$ be the column-echelon form for ∂_k relative to bases $\{e_j\}$ and $\{\hat{e}_i\}$ for C_k and Z_{k-1}, respectively. If row i has pivot $\tilde{M}_k(i,j) = t^n$, it contributes $\Sigma^{\deg \hat{e}_i} F[t]/t^n$ to the description of H_{k-1}. Otherwise, it contributes $\Sigma^{\deg \hat{e}_i} F[t]$. Equivalently, we get $(\deg \hat{e}_i, \deg \hat{e}_i + n)$ and $(\deg \hat{e}_i, \infty)$, respectively, as $\mathcal{P}$-intervals for H_{k-1}.

In our example, $\tilde{M}_1(1,1) = t$ in Equation (8). As $\deg d = 1$, the element contributes $\Sigma^1 \mathbb{Z}_2[t]/(t)$ or $\mathcal{P}$-interval (1,2) to the description of H_0.

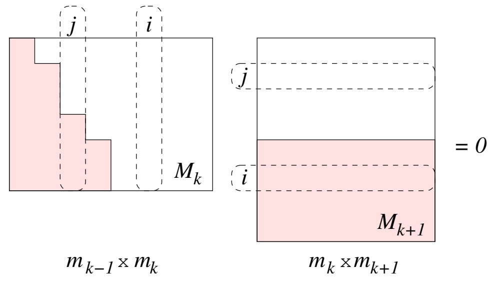


Figure 7. As $\partial_k \partial_{k+1} = 0$, $M_k M_{k+1} = 0$ and this is unchanged by elementary operations. When M_k is reduced to echelon form $\tilde{M}_k$ by column operations, the corresponding row operations zero out rows in M_{k+1} that correspond to pivot columns in $\tilde{M}_k$.

We now wish to represent ∂_{k+1} in terms of the basis we computed for Z_k. We begin with the standard matrix representation M_{k+1} of ∂_{k+1}. As $\partial_k \partial_{k+1} = 0$, $M_k M_{k+1} = 0$, as shown in Figure 7. Furthermore, this relationship is unchanged by elementary operations. Since the domain of ∂_k is the codomain of ∂_{k+1}, the elementary column operations we used to transform M_k into echelon form $\tilde{M}_k$ give corresponding row operations on M_{k+1}. These row operations zero out rows in M_{k+1} that correspond to non-zero pivot columns in $\tilde{M}_k$, and give a representation of ∂_{k+1} relative to the basis we just computed for Z_k. This is precisely what we are after. We can get it, however, with hardly any work.

Lemma 4.2 (Basis Change) To represent ∂_{k+1} relative to the standard basis for C_{k+1} and the basis computed for Z_k, simply delete rows in M_{k+1} that correspond to pivot columns in $\tilde{M}_k$.

Proof: We only used elementary column operations of types (1,3) in our variant of Gaussian elimination. Only the latter changes values in the matrix. Suppose we replace column i by (column i) + q(column j) in order to eliminate an element in a pivot row j, as shown in Figure 7. This operation amounts to replacing column basis element e_i by $e_i + qe_j$ in M_k. To effect the same replacement in the row basis for ∂_{k+1}, we need to replace row j with (row j) $-$ q(row i). However, row j is eventually zeroed-out, as shown in Figure 7, and row i is never changed by any such operation. □

Therefore, we have no need for row operations. We simply eliminate rows corresponding to pivot columns one dimension lower to get the desired representation for ∂_{k+1} in terms of the basis for Z_k. This completes the induction. In our example, the standard matrix representation for ∂_2 is

$$M_2 = \left[\begin{array}{c|cc} & abc & acd \\ \hline ac & t & t^2 \\ ad & 0 & t^3 \\ cd & 0 & t^3 \\ bc & t^3 & 0 \\ ab & t^3 & 0 \end{array}\right].$$

To get a representation in terms of C_2 and the basis (z_1, z_2) for Z_1 we computed earlier, we simply eliminate the bottom three rows. These rows are associated with pivots in $\tilde{M}_1$, according to Equation (8). We get

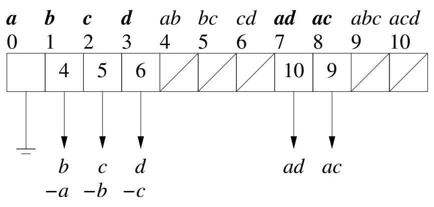
$$\check{M}_2 = \left[\begin{array}{c|cc} & abc & acd \\ \hline z_2 & t & t^2 \\ z_1 & 0 & t^3 \end{array}\right],$$

where we have also replaced ad and ac with the corresponding basis elements $z_1 = ad - bc - cd - ab$ and $z_2 = ac - bc - ab$.

4.2 Algorithm

Our discussion gives us an algorithm for computing $\mathcal{P}$-intervals of an $F[t]$-module over field F. It turns out, however, that we can simulate the algorithm over the field itself, without the need for computing the $F[t]$-module. Rather, we use two significant observations from the derivation of the algorithm. First, Lemma 4.1 guarantees that if we eliminate pivots in the order of decreasing degree, we may read off the entire description from the echelon form and do not need to reduce to normal form. Second, Lemma 4.2 tells us that by simply noting the pivot columns in each dimension and eliminating the corresponding rows in the next dimension, we get the required basis change.

Therefore, we only need column operations throughout our procedure and there is no need for a matrix representation. We represent the boundary operators as a set of boundary chains corresponding to the columns



a ***b*** ***c*** ***d*** *ab* *bc* *cd* ***ad*** ***ac*** *abc* *acd*
0 1 2 3 4 5 6 7 8 9 10
4 5 6 10 9
b $-a$
c $-b$
d $-c$
ad
ac

Figure 8. Data structure after running the algorithm on the filtration in Figure 1. Marked simplices are in bold italic.

of the matrix. Within this representation, column exchanges (type (1)) have no meaning, and the only operation we need is of type (3). Our data structure is an array T with a slot for each simplex in the filtration, as shown in Figure 8 for our example. Each simplex gets a slot in the table. For indexing, we need a full ordering of the simplices, so we complete the partial order defined by the degree of a simplex by sorting simplices according to dimension, breaking all remaining ties arbitrarily (we did this implicitly in the matrix representation.) We also need the ability to *mark* simplices to indicate non-pivot columns.

Rather than computing homology in each dimension independently, we compute homology in all dimensions incrementally and concurrently. The algorithm, as shown in Figure 9, stores the list of $\mathcal{P}$-intervals for H_k in L_k.

```
COMPUTEINTERVALS (K) {
  for $k = 0$ to $\dim(K)$ $L_k = \emptyset$;
  for $j = 0$ to $m - 1$ {
    d = REMOVEPIVOTROWS ($\sigma^j$);
    if ($d = \emptyset$) Mark $\sigma^j$;
    else {
      $i =$ maxindex $d$; $k = \dim \sigma^i$;
      Store $j$ and $d$ in $T[i]$;
      $L_k = L_k \cup \{(\deg \sigma^i, \deg \sigma^j)\}$
    }
  }
  for $j = 0$ to $m - 1$ {
    if $\sigma^j$ is marked and $T[j]$ is empty {
      $k = \dim \sigma^j$; $L_k = L_k \cup \{(\deg \sigma^j, \infty)\}$
    }
  }
}
```

Figure 9. Algorithm COMPUTEINTERVALS processes a complex of m simplices. It stores the sets of $\mathcal{P}$-intervals in dimension k in L_k.

When simplex σ^j is added, we check via procedure REMOVEPIVOTROWS to see whether its boundary chain d corresponds to a zero or pivot column. If the chain is empty, it corresponds to a zero column and

```
chain REMOVEPIVOTROWS ($\sigma$) {
    $k = \dim \sigma$; $d = \partial_k \sigma$;
    Remove unmarked terms in $d$;
    while ($d \neq \emptyset$) {
      $i = \text{maxindex}\, d$;
      if $T[i]$ is empty, break;
      Let $q$ be the coefficient of $\sigma^i$ in $T[i]$;
      $d = d - q^{-1} T[i]$;
    }
    return $d$;
  }
```

Figure 10. Algorithm REMOVEPIVOTROWS first eliminates rows not marked (not corresponding to the basis for Z_{k-1}), and then eliminates terms in pivot rows.

we mark σ^j: its column is a basis element for Z_k, and the corresponding row should not be eliminated in the next dimension. Otherwise, the chain corresponds to a pivot column and the term with the maximum index $i = \text{maxindex}\, d$ is the pivot, according the procedure described for the $F[t]$-module. We store index j and chain d representing the column in $T[i]$. Applying Corollary 4.1, we get $\mathcal{P}$-interval $(\deg \sigma^i, \deg \sigma^j)$. We continue until we exhaust the filtration. We then perform another pass through the filtration in search of infinite $\mathcal{P}$-intervals: marked simplices whose slot is empty.

We give the function REMOVEPIVOTROWS in Figure 10. Initially, the function computes the boundary chain d for the simplex. It then applies Lemma 4.2, eliminating all terms involving unmarked simplices to get a representation in terms of the basis for Z_{k-1}. The rest of the procedure is Gaussian elimination in the order of decreasing degree, as dictated by our discussion for the $F[t]$-module. The term with the maximum index $i = \max d$ is a potential pivot. If $T[i]$ is non-empty, a pivot already exists in that row, and we use the inverse of its coefficient to eliminate the row from our chain. Otherwise, we have found a pivot and our chain is a pivot column. For our example filtration in Figure 8, the marked 0-simplices $\{a, b, c, d\}$ and 1-simplices $\{ad, ac\}$ generate $\mathcal{P}$-intervals $L_0 = \{(0, \infty), (0, 1), (1, 1), (1, 2)\}$ and $L_1 = \{(2, 5), (3, 4)\}$, respectively.

4.3 Discussion

From our derivation, it is clear that the algorithm has the same running time as Gaussian elimination over fields. That is, it takes $O(m^3)$ in the worst case, where m is the number of simplices in the filtration. The algorithm is very simple, however, and represents the matrices efficiently. In our preliminary experiments, we have seen a linear time behavior for the algorithm.

5 Algorithm for PIDs

The correspondence we established in Section 3 eliminates any hope for a simple classification of persistent groups over rings that are not fields. Nevertheless, we may still be interested in their computation. In this section, we give an algorithm to compute the persistent homology groups $H_k^{i,p}$ of a filtered complex K for a fixed i and p. The algorithm we provide computes persistent homology over any PID D of coefficients by utilizing a reduction algorithm over that ring.

To compute the persistent group, we need to obtain a description of the numerator and denominator of the quotient group in Equation (4). We already know how to characterize the numerator. We simply reduce the standard matrix representation M_k^i of ∂_k^i using the reduction algorithm. The denominator, $\mathsf{B}_k^{i,p} = \mathsf{B}_k^{i+p} \cap \mathsf{Z}_k^i$, plays the role of the boundary group in Equation (4). Therefore, instead of reducing matrix M_{k+1}^i, we need to reduce an alternate matrix $M_{k+1}^{i,p}$ that describes this boundary group. We obtain this matrix as follows:

(1) We reduce matrix M_k^i to its normal form and obtain a basis $\{z^j\}$ for Z_k^i, using fact (ii) in Section 2.5. We may merge this computation with that of the numerator.

(2) We reduce matrix M_{k+1}^{i+p} to its normal form and obtain a basis $\{b^l\}$ for B_k^{i+p} using fact (iii) in Section 2.5.

(3) Let $N = [\{b^l\}\ \{z^j\}] = [B\ Z]$, that is, the columns of matrix N consist of the basis elements from the bases we just computed, and B and Z are the respective submatrices defined by the bases. We next reduce N to normal form to find a basis $\{u^q\}$ for its null-space. As before, we obtain this basis using fact (ii). Each $u^q = [\alpha^q\ \zeta^q]$, where α^q, ζ^q are vectors of coefficients of $\{b^l\}, \{z^j\}$, respectively. Note that $Nu^q = B\alpha^q + Z\zeta^q = 0$ by definition. In other words, element $B\alpha^q = -Z\zeta^q$ is belongs to the span of both bases. Therefore, both $\{B\alpha^q\}$ and $\{Z\zeta^q\}$ are bases for $\mathsf{B}_k^{i,p} = \mathsf{B}_k^{i+p} \cap \mathsf{Z}_k^i$. We form a matrix $M_{k+1}^{i,p}$ from either.

We now reduce $M_{k+1}^{i,p}$ to normal form and read off the torsion coefficients and the rank of $\mathsf{B}_k^{i,p}$. It is clear from the procedure that we are computing the persistent groups correctly, giving us the following.

Theorem 5.1 For coefficients in any PID, persistent homology groups are computable in the order of time and space of computing homology groups.

6 Experiments

In this section, we discuss experiments using an implementation of the persistence algorithm for arbitrary fields. Our aim is to further elucidate the contributions of this paper. We look at two scenarios where the previous algorithm would not be applicable, but where our algorithm succeeds in providing information about a topological space.

6.1 Implementation

We have implemented our field algorithm for $\mathbb{Z}_p$ for p a prime, and $\mathbb{Q}$ coefficients. Our implementation is in `C` and utilizes GNU MP, a multiprecision library, for exact computation [14]. We have a separate implementation for coefficients in $\mathbb{Z}_2$ as the computation is greatly simplified in this field. The coefficients are either 0, or 1, so there is no need for orienting simplices or maintaining coefficients. A k-chain is simply a list of simplices, those with coefficient 1. Each simplex is its own inverse, reducing the group operation to the *symmetric difference*, where the sum of two k-chains c, d is $c + d = (c \cup d) - (c \cap d)$. We use a 2.2 GHz Pentium 4 Dell PC with 1 GB RAM running Red Hat Linux 7.3 for computing the timings.

6.2 Data

Our algorithm requires a persistence complex as input. In the introduction, we discussed how persistence complexes arise naturally in practice. In Example 1.3, we discussed generating persistence complexes using excursion sets of Morse functions over manifolds. We have implemented a general framework for computing complexes of this type. We must emphasize, however, that our persistence software processes persistence complexes of any origin.

Our framework takes a tuple (K, f) as input and produces a persistence complex $\mathcal{C}(K, f)$ as output. K is a d-dimensional simplicial complex that triangulates an underlying manifold, and $f\colon \operatorname{vert} K \to \mathbb{R}$ is a discrete function over the vertices of K that we extend linearly over the remaining simplices of K. The function f acts as the Morse function over the manifold, but need not be Morse for our purposes. Frequently, our complex is augmented with a map $\varphi : K \to \mathbb{R}^d$ that immerses or embeds the manifold in Euclidean space. Our algorithm does not require φ for computation, but φ is often provided as a discrete map over the vertices of K and is extended linearly as before. For example, Figure 11 displays a triangulated Klein bottle, immersed in $\mathbb{R}^3$. For each dataset, Table 2 gives the number s_k of k-simplices, as well as the Euler characteristic $\chi = \sum_k (-1)^k s_k$. We use the Morse function to compute the excursion set filtration for each dataset. Table 3 gives information on the resulting filtrations.

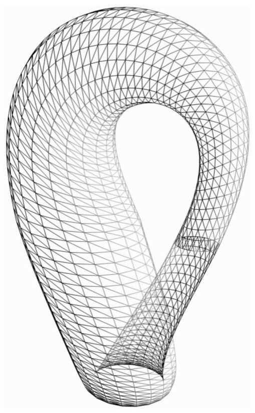


Figure 11. A wire-frame visualization of dataset K, an immersed triangulated Klein bottle with 4000 triangles.

	$\lvert K\rvert$	len	filt (s)	pers (s)
K	12,000	1,020	0.03	< 0.01
E	529,225	3,013	3.17	5.00
J	3,029,383	256	24.13	50.23

Table 3. Filtrations. The number of simplices in the filtration $|K| = \sum_i s_i$, the length of the filtration (number of distinct values of function f), time to compute the filtration, and time to compute persistence over $\mathbb{Z}_2$ coefficients.

6.3 Field Coefficients

A contribution of this paper is the generalization of the persistence algorithm to arbitrary fields. This contribution is important when the manifold under study contains torsion. To make this clear, we compute the homology of the Klein bottle using the persistence algorithm. Here, we are interested only in the Betti numbers of the final complex in the filtration for illustrative purposes. The non-orientability of the Klein bottle is visible in Figure 11. The change in triangle orientation at the parametrization boundary leads to a rendering artifact where two sets of triangles are front-facing. In homology, the non-orientability of the Klein bottle manifests itself as a torsional 1-cycle c where $2c$ is a boundary (indeed, it bounds the surface itself.) The homology groups over $\mathbb{Z}$ are:

$$\begin{aligned} \mathsf{H}_0(K) &= \mathbb{Z}, \\ \mathsf{H}_1(K) &= \mathbb{Z} \times \mathbb{Z}_2, \\ \mathsf{H}_2(K) &= \{0\}. \end{aligned}$$

	number s_k of k-simplices					χ
	0	1	2	3	4	
K	2,000	6,000	4,000	0	0	0
E	3,095	52,285	177,067	212,327	84,451	1
J	17,862	297,372	1,010,203	1,217,319	486,627	1

Table 2. Datasets. K is the Klein bottle, shown in Figure 11. E is potential around electrostatic charges. J is supersonic jet flow.

F	β_0	β_1	β_2	time (s)
$\mathbb{Z}_2$	1	2	1	0.01
$\mathbb{Z}_3$	1	1	0	0.23
$\mathbb{Z}_5$	1	1	0	0.23
$\mathbb{Z}_{3203}$	1	1	0	0.23
$\mathbb{Q}$	1	1	0	0.50

Table 4. Field coefficients. The Betti numbers of $\boldsymbol{K}$ computed over field $\boldsymbol{F}$ and time for the persistence algorithm. We use a separate implementation for $\mathbb{Z}_2$ coefficients.

Note that $\beta_1 = \operatorname{rank} \mathsf{H}_1 = 1$. We now use the "height function" as our Morse function, $f = z$, to generate the filtration in Table 3. We then compute the homology of dataset K with field coefficients using our algorithm, as shown in Table 4.

Over $\mathbb{Z}_2$, we get $\beta_1 = 2$ as homology is unable to recognize the torsional boundary $2c$ with coefficients 0 and 1. Instead, it observes an additional class of homology 1-cycles. By the Euler-Poincaré relation, $\chi = \sum_i \beta_i$, so we also get a class of 2-cycles to compensate for the increase in β_1 [16]. Therefore, $\mathbb{Z}_2$-homology misidentifies the Klein bottle as the torus. Over any other field, however, homology turns the torsional cycle into a boundary, as the inverse of 2 exists. In other words, while we cannot observe torsion in computing homology over fields, we can deduce its existence by comparing our results over different coefficient sets. Similarly, we can compare sets of $\mathcal{P}$-intervals from different computations to discover torsion in a persistence complex.

Note that our algorithm's performance for this dataset is about the same over arbitrary finite fields, as the coefficients do not get large. The computation over $\mathbb{Q}$ takes about twice as much time and space, since each rational is represented as two integers in GNU MP.

6.4 Higher Dimensions

A second contribution of this paper is the extension of the persistence algorithm from subcomplexes of $\mathbb{S}^3$ to complexes in arbitrary dimensions. We have already utilized this capability in computing the homology of the Klein bottle. We now examine the performance of this algorithm in higher dimensions. For practical motivation, we use large-scale time-varying volume data as input. Advances in data acquisition systems and computing technologies have resulted in the generation of massive sets of measured or simulated data. The datasets usually contain the time evolution of physical variables, such as temperature, pressure, or flow velocity at sample points in space. The goal is to identify and localize significant phenomena within the data. We propose using persistence as the significance measure.

The underlying space for our datasets is the four-dimensional space-time manifold. For each dataset, we triangulate the convex hull of the samples to get a triangulation. Each complex listed in Table 2 is homeomorphic to a four-dimensional ball and has $\chi = 1$. Dataset E contains the potential around electrostatic charges at each vertex. Dataset J records the supersonic flow velocity of a jet engine. We use these values as Morse functions to generate the filtrations. We then compute persistence over $\mathbb{Z}_2$ coefficients to get the Betti numbers. We give filtration sizes and timings in Table 3. Figure 12 displays β_2 for dataset J. We observe a large number of two-dimensional cycles (voids), as the co-dimension is 2. Persistence allows us to decompose this graph into the set of $\mathcal{P}$-intervals. Although there are 730,692 $\mathcal{P}$-intervals in dimension 2, most are empty as the topological attribute is created and destroyed at the same function level. We draw the 502 non-empty $\mathcal{P}$-intervals in Figure 13. We note that the $\mathcal{P}$-intervals represent a compact and general *shape descriptor* for arbitrary spaces.

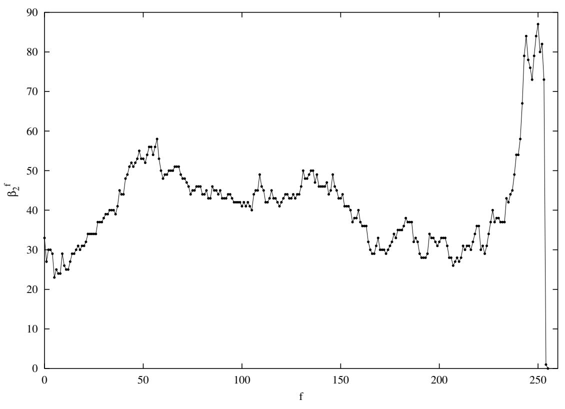


Figure 12. Graph of $\boldsymbol{\beta_2^f}$ for dataset J, where $\boldsymbol{f}$ is the flow velocity.

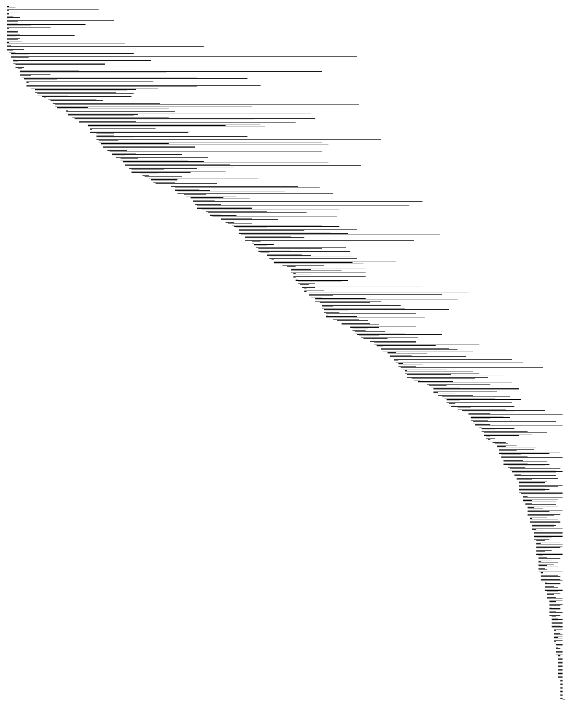


Figure 13. The 502 non-empty $\mathcal{P}$-intervals for dataset J in dimension 2. The amalgamation of these intervals gives the graph in Figure 12.

For the large data sets, we do not compute persistence over alternate fields as the computation requires in excess of 2 gigabytes of memory. In the case of finite fields $\mathbb{Z}_p$, we may restrict the prime p so that the computation fits within an integer. This is a reasonable restriction, as on most modern machines with 32-bit integers, it implies $p < 2^{16} - 1$. Given this restriction, any coefficient will be less than p and representable as a 4-byte integer. The GNU MP exact integer format, on the other hand, requires at least 16 bytes for representing any integer.

7 Conclusion

We believe the most important contribution of this paper is a reinterpretation of persistent homology within the classical framework of algebraic topology. Our interpretation allows us to:

1. establish a correspondence that fully describes the structure of persistent homology over any field, not only over $\mathbb{Z}_2$, as in the previous result,
2. and relate the previous algorithm to the classic reduction algorithm, thereby extending it to arbitrary fields and arbitrary dimensional complexes, not just subcomplexes of $\mathbb{S}^3$ as in the previous result.

We provide implementations of our algorithm for fields, and show that they perform quite well for large datasets. Finally, we give an algorithm for computing a persistent homology group with fixed parameters over arbitrary PIDs.

Our software for n-dimensional complexes enables us to analyze arbitrary-dimensional point cloud data and their derived spaces. One current project uses this implementation for feature recognition using a novel algebraic method [2]. Another project analyzes the topological structures in a high-dimensional data set derived from natural images [7]. Yet another applies persistence to derived spaces to arrive at compact shape descriptors for geometric objects [3, 5]. Future theoretical work include examining invariants for persistent homology over non-fields and defining *multivariate persistence*, where there is more than one persistence dimension. An example would be tracking a Morse function as well as density of sampling on a manifold. Finally, we have recently reimplemented the algorithm using the generic paradigm. This implementation will soon be a part of the CGAL library [4].

Acknowledgments

The first author thanks Herbert Edelsbrunner and John Harer for discussions on $\mathbb{Z}$-homology, and Leo Guibas for providing support and encouragement. Both authors thank Anne Collins for her thorough review of the manuscript and Ajith Mascarenhas for providing simplicial complexes for datasets E and J. The original data are part of the Advanced Visualization Technology Center's data repository and appears courtesy of Andrea Malagoli and Milena Micono of the Laboratory for Astrophysics and Space Research (LASR) at the University of Chicago. Figure 11 was rendered in Stanford Graphics Lab's *Scanalyze*.

References

[1] Basu, S. On bounding the Betti numbers and computing the Euler characteristic of semi-algebraic sets. *Discrete Comput. Geom. 22* (1999), 1–18.

[2] Carlsson, E., Carlsson, G., and de Silva, V. An algebraic topological method for feature identification, 2003. Manuscript.

[3] Carlsson, G., Zomorodian, A., Collins, A., and Guibas, L. Persistence barcodes for shapes. In *Proc. Symp. Geom. Process.* (2004), pp. 127–138.

[4] CGAL. Computational geometry algorithms library. http://www.cgal.org.

[5] Collins, A., Zomorodian, A., Carlsson, G., and Guibas, L. A barcode shape descriptor for curve

point cloud data. In *Proc. Symp. Point-Based Graph.* (2004), pp. 181–191.

[6] COX, D., LITTLE, J., AND O'SHEA, D. *Ideals, Varieties, and Algorithms*, second ed. Springer-Verlag, New York, NY, 1991.

[7] DE SILVA, V., AND CARLSSON, G. Topological estimation using witness complexes. In *Proc. Symp. Point-Based Graph.* (2004), pp. 157–166.

[8] DONALD, B. R., AND CHANG, D. R. On the complexity of computing the homology type of a triangulation. In *Proc. 32nd Ann. IEEE Sympos. Found Comput. Sci.* (1991), pp. 650–661.

[9] DUMAS, J.-G., HECKENBACH, F., SAUNDERS, B. D., AND WELKER, V. Computing simplicial homology based on efficient Smith normal form algorithms. In *Algebra, Geometry, and Software Systems* (2003), pp. 177–207.

[10] DUMMIT, D., AND FOOTE, R. *Abstract Algebra*. John Wiley & Sons, Inc., New York, NY, 1999.

[11] EDELSBRUNNER, H., LETSCHER, D., AND ZOMORODIAN, A. Topological persistence and simplification. *Discrete Comput. Geom. 28* (2002), 511–533.

[12] EISENBUD, D. *Commutative Algebra with a View Toward Algebraic Theory*. Springer, New York, NY, 1995.

[13] FRIEDMAN, J. Computing Betti numbers via combinatorial Laplacians. In *Proc. 28th Ann. ACM Sympos. Theory Comput.* (1996), pp. 386–391.

[14] GRANLUND, T. The GNU multiple precision arithmetic library. http://www.swox.com/gmp.

[15] GROMOV, M. Hyperbolic groups. In *Essays in Group Theory*, S. Gersten, Ed. Springer Verlag, New York, NY, 1987, pp. 75–263.

[16] MUNKRES, J. R. *Elements of Algebraic Topology*. Addison-Wesley, Reading, MA, 1984.

[17] UHLIG, F. *Transform Linear Algebra*. Prentice Hall, Upper Saddle River, NJ, 2002.

[18] ZOMORODIAN, A. *Computing and Comprehending Topology: Persistence and Hierarchical Morse Complexes*. PhD thesis, University of Illinois at Urbana-Champaign, 2001.