DUALITIES IN PERSISTENT (CO)HOMOLOGY

VIN DE SILVA

Department of Mathematics, Pomona College
610 N College Ave, Claremont, California

DMITRIY MOROZOV

Lawrence Berkeley National Lab
1 Cyclotron Road, Mailstop 50F-1650, Berkeley, CA 94720

MIKAEL VEJDEMO-JOHANSSON

School of Computer Science, Jack Cole Building, North Haugh
St Andrews KY16 9SX, Scotland

Abstract. We consider sequences of absolute and relative homology and cohomology groups that arise naturally for a filtered cell complex. We establish algebraic relationships between their persistence modules, and show that they contain equivalent information. We explain how one can use the existing algorithm for persistent homology to process any of the four modules, and relate it to a recently introduced persistent cohomology algorithm. We present experimental evidence for the practical efficiency of the latter algorithm.

1. Introduction

The subject of inverse problems deals, fundamentally, with the inference of shape. From some related measurements — such as a family of particular path integrals — we try to deduce geometric

E-mail addresses: vin.desilva@pomona.edu, dmitriy@mrzv.org, mikael@johanssons.org.

Date: September 16, 2011.

VdS has been partially supported by DARPA, through grants HR0011-05-1-0007 (TDA) and HR0011-07-1-0002 (SToMP), and holds a Digiteo Chair. DM has been partially supported by DARPA grant HR0011-05-1-0007 (TDA) and by the DOE Office of Science, Advanced Scientific Computing Research, under award number KJ0402-KRD047, under contract number DE-AC02-05CH11231. MVJ has been partially supported by the Office of Naval Research, through grant N00014-08-1-0931.

information. With the classical techniques in the field, with Fourier and other integral transforms, one can deduce an impressive amount of information. However, with non-linearity, and ill-posed, ill-conditioned situations, the classical methods need increasingly large amounts of regularization or data cleaning. Topology offers a family of methods that allow the inference of information — if not geometric, then at least topological — into the field. In particular, the recent development of persistent homology [1], and its applications to topological data analysis [2], demonstrate an approach to topological invariants that becomes applicable to high-dimensional, finite and discrete measurement sets.

To take an explicit example, geological *sonar* investigations employ inverse problem methods to investigate the geometric structure of the density sublevel sets in subterranean domains, relating density variations to occurrences of oil, water or mineral pockets. The kind of information sought starts out with a qualitative judgement: is there a pocket at all; are there several or few; are they connected or not? These first questions, before the shape can be given an explicit geometric description, are a matter of topological properties, and the study of sublevel sets of functions on domains is one of the most convincing uses of persistent homology.

The persistent homology algorithm of Edelsbrunner, Letscher, and Zomorodian [1] is now ten years old. In its natural general form [3], the input is a filtered 'space' (topological space, or simplicial complex, or abstract chain complex) and the output is a collection of half-open real intervals known as a barcode or a persistence diagram.

These barcodes contain one bar for each topological feature found – one bar for each homology class, representing a hole or a higher-dimensional void. These bars come with a starting point, indicating the focal level at which the feature first becomes visible, and an ending point, indicating the focal level at which the feature vanishes again. A fundamental tenet, as described in [2] is that the length of such a bar – the difference between when it shows up and when it vanishes – encodes the relevance of the feature. This emphasizes the topological features that are enveloped by a dense distribution of points, and yet have a geometrically large void in the middle.

In many applications, all that is required is the barcode. This tells us how many homological features exist at any given level of the filtration, and how many of those survive to any given subsequent level. This information is already very rich, and has been proven to be statistically robust [4, 5]. Sometimes more is required. The most common request is for geometric representatives of the features: in other words, explicit homology cycles representing each barcode interval. The original algorithm provides these cycles automatically: they are essential to the way in which the barcode intervals are calculated.

In fact, there are at least four natural persistent objects that can be derived from a filtered space. They are:

$$\text{persistent} \left\{ \begin{array}{c} \text{absolute} \\ \text{relative} \end{array} \right\} \left\{ \begin{array}{c} \text{homology} \\ \text{cohomology} \end{array} \right\}$$

The 'standard' object is persistent absolute homology, and most treatments focus on this. However, it has become increasingly clear that the other three objects are important in their own right. The transition between homology and cohomology is in some sense nothing more than the duality of vector spaces; persistent homology and cohomology have the same barcodes. However, homology cycles and cohomology cocycles are quite different, and some applications call for cocycles rather than cycles [6]. The occasional utility of relative rather than absolute homology is probably easier to grasp intuitively; for example see [7] for an application in sensor networks. It is easy to 'fake' the calculation of relative homology using absolute homology and a cone construction, but we point out that this trick is unnecessary.

Our goal in this paper is to provide a streamlined approach to calculating barcodes and (co)cycle representatives for all four persistent objects. We discuss this approach in terms of abstract algebra and in terms of matrix computations.

We observe that:

- absolute homology and cohomology have the same barcode;
- relative homology and cohomology have the same barcode;
- the absolute barcode and the relative barcode can be deduced from each other;
- the cycles and bounding chains of persistent absolute homology determine, and are determined by, the cycles and bounding chains of persistent relative homology;
- likewise, for absolute and relative cohomology cocycles and bounding cochains.

We discuss two different dualities. There is the standard duality which interchanges homology and cohomology. We call this 'pointwise' duality. More interestingly, there is a different duality which makes the following interchange:

$$\text{absolute homology} \leftrightarrow \text{relative cohomology}$$
$$\text{absolute cohomology} \leftrightarrow \text{relative homology}$$

We call this 'global' duality, and it appears only in the context of persistent topology. Global duality 'commutes' with all possible algorithms and theorems: a method for calculating persistent absolute homology will equally well calculate persistent relative cohomology, once the input data have been turned upside-down in a particular way.

Combining all of these equalities and dualities, it emerges that a single calculation (run twice) suffices to calculate all four persistent objects. Actually, we describe two different algorithms for that calculation: pHcol (the 'column algorithm') and pHrow (the 'row algorithm'). Here pHcol is essentially the classic algorithm of [1, 3]; pHrow organises the calculation quite differently. The preferred choice depends, in any given situation, on whether it is easier to look up rows or columns of the boundary matrix of the filtered space — the specific representation of the space usually biases this choice.

We are rewarded by an unexpected payoff. If we require only the absolute barcode, it turns out that the best choice is an optimised version of pHrow called pCoh (the 'cohomology algorithm'). We give experimental evidence to this effect. Standard practice has been to use pHcol. We therefore call on persistent topology library-writers to implement pCoh, and on persistent topology library-users to use it.

1.1. **Outline of paper.** Section 2 is devoted to the algebra underlying this work. In 2.1–2.5 we conduct the discussion at a high level (homology functors are assumed given, black-box style), and in 2.6–2.7 we go into the necessary chain-level details. In 2.8 we give a brief abstract description of the two dualities.

Section 3 is about matrix algorithms. In 3.1–3.4 we interpret the preceding algebra in terms of matrix decomposition (following [8], again black-box style). In 3.5 we present the two algorithms, pHcol and pHrow, and explain why they give the same output.

In Section 4 we relate the ideas in this paper to an earlier cohomology algorithm pCoh published in [6]. We indicate why we expect pCoh to be faster that pHcol and pHrow for computing barcodes of filtered simplicial complexes, and we verify this by experiment.

2. Algebra

We will assume that the reader is familiar with homology theory. Our preference is to use cellular homology, because it is a little more general than simplicial homology.

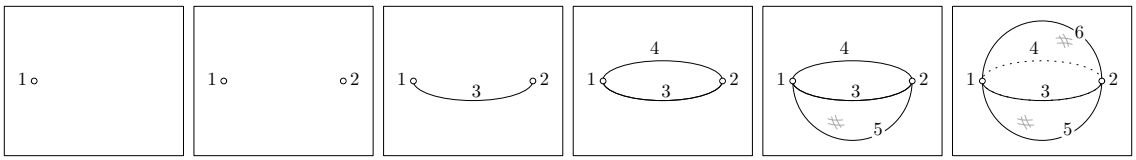
2.1. **Coefficients.** Individual (co)homology groups are defined with coefficients in a field $\mathbf{k}$, which remains fixed throughout this paper. Persistent (co)homology then has the structure of a graded module over the polynomial ring $\mathbf{k}[t]$. Many things go wrong when we replace the field $\mathbf{k}$ with a ring, in particular the ring of integers $\mathbf{Z}$. See [3].

2.2. **Filtered complexes.** We are interested in the persistent topology of filtered topological spaces. The simplest example is a filtered cell complex, which is a sequence $\mathbb{X}$ of cell complexes

$$\mathbb{X}: \quad X_1 \subset X_2 \subset \cdots \subset X_n = X_\infty \tag{2.1}$$

where X_1 is a vertex σ_1, and thereafter each complex is obtained from the previous one by adding a single cell: $X_i = X_{i-1} \cup \sigma_i$. Here the index set is $\{1, 2, \ldots, n\}$. Usually we attach real values a_i to the indices, which must satisfy $a_1 \leq a_2 \leq \cdots \leq a_n$.

Example. Our running example $\mathbb{S}$ will be a cellular filtration of the 2-sphere:



There are six cells, $\sigma_1, \ldots, \sigma_6$ which appear at times $a_i = i$, for $i = 1, \ldots, 6$.

2.3. **Persistent homology.** If we apply a homology functor $\mathsf{H}(-)$ to a filtered complex $\mathbb{X}$ we obtain a diagram:

$$\mathsf{H}(\mathbb{X}): \quad \mathsf{H}(X_1) \to \mathsf{H}(X_2) \to \cdots \to \mathsf{H}(X_n) \tag{2.2}$$

Typically $\mathsf{H}(-)$ denotes the k-dimensional homology $\mathsf{H}_k(-; \mathbf{k})$ or the total homology $\mathsf{H}_*(-; \mathbf{k})$. Then (2.2) is a diagram of finite-dimensional vector spaces and linear maps, also known as a **persistence module**.

A persistence module decomposes as a direct sum of **interval modules** [3]. These are labelled by ordered pairs of integers $[p, q]$, where $1 \leq p \leq q \leq n$. The pair $[p, q]$ indicates a feature which persists over the index set $\{p, \ldots, q\}$. We frequently interpret $[p, q]$ as the half-open real interval $[a_p, a_{q+1})$, with the convention that $a_{n+1} = \infty$.

The **persistence diagram** or **barcode** is the multiset of ordered pairs $[p, q]$ in the decomposition, or alternatively the multiset of half-open intervals $[a_p, a_{q+1})$. Thus we write:

$$\begin{aligned} \mathrm{Pers}(\mathsf{H}(\mathbb{X})) &= \{[p_1, q_1], \ldots, [p_m, q_m]\} \\ &= \{[a_{p_1}, a_{q_1+1}), \ldots, [a_{p_m}, a_{q_m+1})\} \end{aligned}$$

It is customary in applications to discard from the persistence diagram those intervals $[a_p, a_{q+1})$ for which $a_p = a_{q+1}$.

Example. In our running example, the intermediate spaces S_1, S_3, S_5 are all contractible, whereas S_2, S_4, S_6 are homeomorphic to the 0-sphere, 1-sphere, and 2-sphere, respectively. There are four intervals in the persistence diagram of $\mathsf{H}_*(\mathbb{S})$:

$$\mathrm{Pers}(\mathsf{H}_*(\mathbb{S})) = \{[1, 6]_0, [2, 2]_0, [4, 4]_1, [6, 6]_2\} = \{[1, \infty)_0, [2, 3)_0, [4, 5)_1, [6, \infty)_2\}$$

The subscript k in $[p, q]_k$ or $[a_p, a_{q+1})_k$ indicates that the feature occurs in k-dimensional homology.

2.4. **The four standard persistence modules.** The standard persistent homology module $\mathsf{H}_*(\mathbb{X})$ tells us how the absolute homology groups $\mathsf{H}_*(X_i)$ relate to each other as i varies. We can play the same game with the absolute cohomology groups $\mathsf{H}^*(X_i)$, the relative homology groups $\mathsf{H}_*(X_n, X_i)$, and the relative cohomology groups $\mathsf{H}^*(X_n, X_i)$. Here are the four sequences, lined up for comparison.

$$\begin{aligned}
\mathsf{H}_*(\mathbb{X}): \quad & \mathsf{H}_*(X_1) \quad \to \ \ldots \to \mathsf{H}_*(X_{n-1}) \quad \to \mathsf{H}_*(X_n) \\
\mathsf{H}^*(\mathbb{X}): \quad & \mathsf{H}^*(X_1) \quad \leftarrow \ \ldots \leftarrow \mathsf{H}^*(X_{n-1}) \quad \leftarrow \mathsf{H}^*(X_n) \\
\mathsf{H}_*(X_\infty, \mathbb{X}): \quad & \mathsf{H}_*(X_n) \to \mathsf{H}_*(X_n, X_1) \to \ \ldots \to \mathsf{H}_*(X_n, X_{n-1}) \\
\mathsf{H}^*(X_\infty, \mathbb{X}): \quad & \mathsf{H}^*(X_n) \leftarrow \mathsf{H}^*(X_n, X_1) \leftarrow \ \ldots \leftarrow \mathsf{H}^*(X_n, X_{n-1})
\end{aligned}$$

The persistence diagram for absolute cohomology is a multiset of integer ordered pairs $[p, q]$ with $1 \le p \le q \le n$. For relative homology and cohomology, the persistence diagrams are multisets of pairs $[p, q]$ with $0 \le p \le q \le n-1$. In all cases, we interpret $[p, q]$ as the half-open interval $[a_p, a_{q+1})$, with the convention that $a_0 = -\infty$ and $a_{n+1} = \infty$.

Example. In our running example, we compute

$$\mathrm{Pers}(\mathsf{H}_*(S_6, \mathbb{S})) = \{[0,0]_0, [2,2]_1, [4,4]_2, [0,5]_2\} = \{[-\infty, 1)_0, [2,3)_1, [4,5)_2, [-\infty, 6)_2\}.$$

For instance, at index 2 we note that there is a nontrivial element of $\mathsf{H}_1(S_6, S_2)$ represented by any arc connecting the two points of S_2. To be specific, the homology class is $[\sigma_3] = [\sigma_4]$. This class vanishes in $\mathsf{H}_1(S_6, S_3)$, and so it generates the interval $[2, 3)$.

The reader may detect a relationship between the barcodes for absolute and relative homology. We formalize this in the next section.

2.5. **Barcode isomorphisms.**

Proposition 2.3. *For all k,*

$$\begin{aligned}
\mathrm{Pers}(\mathsf{H}_k(\mathbb{X})) &= \mathrm{Pers}(\mathsf{H}^k(\mathbb{X})), \\
\mathrm{Pers}(\mathsf{H}_k(X_\infty, \mathbb{X})) &= \mathrm{Pers}(\mathsf{H}^k(X_\infty, \mathbb{X})).
\end{aligned}$$

In other words, homology and cohomology have identical barcodes.

Proof. The universal coefficients theorem [9, Thm 3.2] asserts that there is a natural isomorphism

$$\mathsf{H}^k(X; \mathbf{k}) \equiv \mathrm{Hom}(\mathsf{H}_k(X; \mathbf{k}), \mathbf{k}).$$

In other words, cohomology and homology are dual as vector spaces, and hence have the same dimension. 'Natural' implies that the induced maps

$$\mathsf{H}_k(X_i; \mathbf{k}) \to \mathsf{H}_k(X_j; \mathbf{k}) \quad \text{and} \quad \mathsf{H}^k(X_i; \mathbf{k}) \leftarrow \mathsf{H}^k(X_j; \mathbf{k})$$

are adjoint, and hence have the same rank. Because of the way the barcode is uniquely determined by dimensions and ranks, it follows that the absolute homology and cohomology barcodes are the same. This argument applies equally well to the relative barcodes. □

Notation. We partition each persistence diagram into two parts,

$$\mathrm{Pers} = \mathrm{Pers}_0 \cup \mathrm{Pers}_\infty,$$

where Pers_0 comprises the finite intervals $[a, b)$, and Pers_∞ the infinite intervals $[a, \infty)$ or $[-\infty, b)$.

Proposition 2.4. *For all k,*

$$\begin{aligned} \mathrm{Pers}_0(\mathsf{H}_k(\mathbb{X})) &= \mathrm{Pers}_0(\mathsf{H}_{k+1}(X_\infty, \mathbb{X})), \\ \mathrm{Pers}_\infty(\mathsf{H}_k(\mathbb{X})) &= \mathrm{Pers}_\infty(\mathsf{H}_k(X_\infty, \mathbb{X})), \end{aligned}$$

where the second 'equality' is interpreted as a bijection with $[a, \infty) \leftrightarrow [-\infty, a)$. Thus, persistent homology and relative homology barcodes carry the same information, with a dimension shift for the finite intervals.

The proof appears in Section 2.6.

Remark. Thus, provided we take the dimension shifts into account, all four barcodes carry exactly the same information. If we are only interested in barcodes, we can perform calculations in any one of the four basic sequences, whichever is the most convenient.

Since the last term of $\mathsf{H}_k(\mathbb{X})$ is the same as the first term of $\mathsf{H}_k(X_\infty, \mathbb{X})$, namely $\mathsf{H}_k(X_n)$, the two sequences can be concatenated into a single sequence, which we denote $\mathsf{H}_k(\mathbb{X}) \to \mathsf{H}_k(X_\infty, \mathbb{X})$. The index set for this sequence is $\{1, 2, \ldots, n = \bar{0}, \bar{1}, \bar{2}, \ldots, \overline{n-1}\}$, where we use barred numerals to indicate that we are in the relative homology part of the sequence. The persistence diagram for this complex will have intervals of three possible types:

- (p, q) where $1 \le p \le q < n$, written as $[p, q+1)$ or $[a_p, a_{q+1})$ in interval form.
- $(\bar{p}, \bar{q})$ where $0 < p \le q \le n-1$, written as $[\bar{p}, \overline{q+1})$ or $[\bar{a}_p, \bar{a}_{q+1})$.
- $(p, \bar{q})$ where $1 \le p \le n$, $0 \le q \le n-1$, written as $[p, \overline{q+1})$ or $[a_p, \bar{a}_{q+1})$.

Proposition 2.5. *The barcode* $\mathrm{Pers}\,(\mathsf{H}_k(\mathbb{X}) \to \mathsf{H}_k(X_\infty, \mathbb{X}))$ *comprises the following collection of intervals:*

- *An interval $[a, b)$ for every interval $[a, b)$ in* $\mathrm{Pers}_0(\mathsf{H}_k(\mathbb{X}))$.
- *An interval $[\bar{a}, \bar{b})$ for every interval $[a, b)$ in* $\mathrm{Pers}_0(\mathsf{H}_{k-1}(\mathbb{X}))$.
- *An interval $[a, \bar{a})$ for every interval $[a, \infty)$ in* $\mathrm{Pers}_\infty(\mathsf{H}_k(\mathbb{X}))$.

Proof. Note that the first two classes of interval in $\mathrm{Pers}\,(\mathsf{H}_k(\mathbb{X}) \to \mathsf{H}_k(X_\infty, \mathbb{X}))$ are those which do not meet the middle term $\mathsf{H}_k(X_n)$, and thus correspond exactly to finite intervals in $\mathrm{Pers}(\mathsf{H}_k(\mathbb{X}))$ and $\mathrm{Pers}(\mathsf{H}_k(X_\infty, \mathbb{X}))$. This explains the first two cases, once we make the translation $\mathrm{Pers}_0(\mathsf{H}_k(X_\infty, \mathbb{X})) = \mathrm{Pers}_0(\mathsf{H}_{k-1}(\mathbb{X}))$.

It remains to show is that the intervals of type $[a, \bar{b})$ are always of the form $[a, \bar{a})$.[1] To do this, we need to compare the right filtration of the sequence $\mathsf{H}_k(\mathbb{X})$ with the left filtration of the sequence $\mathsf{H}_k(X_\infty, \mathbb{X})$. The first filtration is the nested sequence of subspaces

$$\mathrm{Im}(\mathsf{H}_k(X_i) \to \mathsf{H}_k(X_n)), \qquad i = 1, 2, \ldots, n-1,$$

of $\mathsf{H}_k(X_n)$, and the second filtration is the nested sequence of subspaces

$$\mathrm{Ker}(\mathsf{H}_k(X_n) \to \mathsf{H}_k(X_n, X_i)) \qquad i = 1, 2, \ldots, n-1,$$

of $\mathsf{H}_k(X_n)$. But the image and kernel subspaces are equal for each i, by the homology long exact sequence for the pair (X_n, X_i). Thus the filtrations are the same. □

Remark. The sequence $\mathsf{H}_k(\mathbb{X}) \to \mathsf{H}_k(X_\infty, \mathbb{X})$ is not the same as the extended persistence [10]. The latter, defined for the sublevel sets of a real-valued function, requires the reversal of the cells in the relative half of the sequence — it translates into the use of the superlevel sets of the function. The meaning of extended persistence for a general filtered space is a lot less straight-forward. The most

[1] Thus the paired intervals $[a, \infty)$ and $[-\infty, a)$ in $\mathrm{Pers}_\infty(\mathsf{H}_k(\mathbb{X}))$ and $\mathrm{Pers}_\infty(\mathsf{H}_k(X_\infty, \mathbb{X}))$ are really the restrictions of a single interval $[a, \bar{a})$ in the concatenated sequence.

significant difference between the two sequences (besides the definition) are the extended pairs, the intervals restricting to $[a, \infty)$ in $\mathrm{Pers}_\infty(\mathsf{H}_k(\mathbb{X}))$. In our sequence, proposition 2.5, they are always of the form $[a, \bar{a})$; on the other hand, in extended persistence there is no such restriction: these pairs carry new information. Another notable difference is that the dualities in this paper apply to general filtered spaces; Poincaré and Lefschetz dualities involved in the analysis of the extended persistence require the domains to be manifolds.

2.6. **Persistent chain complexes.** We now give a more explicit description of the standard persistence modules, in terms of chain complexes. Among other things, this will lead to a clean proof of Proposition 2.4. Given a filtered cell complex $\mathbb{X} = \sigma_1 \cup \cdots \cup \sigma_n$, define a persistence module

$$\mathbb{C}: \quad C_1 \to C_2 \to \cdots \to C_n$$

where $C_i = \langle \sigma_1, \ldots, \sigma_i \rangle$, the vector space over $\mathbf{k}$ with basis elements labelled $\sigma_1, \ldots, \sigma_i$. We also have a boundary map: the boundary of any σ_j is a linear combination of cells which appear previously:

$$\partial \sigma_j = \sum_{i<j} D_{ij} \sigma_i$$

for some collection of coefficients D_{ij}. Geometrically, the cells σ_i for which $D_{ij} \neq 0$ will have dimension one less than the dimension of σ_j.

The boundary map satisfies $\partial^2 = 0$, and it restricts to boundary maps $\partial_i : C_i \to C_i$ for each i. Then $\mathsf{C}_*(X_i) = (C_i, \partial_i)$ is the chain complex[2] for the absolute homology of X_i, and $\mathsf{C}_*(\mathbb{X}) = (\mathbb{C}, \partial)$ is the persistent version for $\mathbb{X}$. Accordingly, we define the persistent absolute homology of $\mathbb{X}$ to be

$$\mathsf{H}_*(\mathbb{X}) = \mathsf{H}(\mathbb{C}, \partial) = \mathrm{Ker}(\partial)/\,\mathrm{Im}(\partial).$$

Shown explicitly as a persistence module, this is:

$$\mathsf{H}(\mathbb{C}, \partial): \quad \frac{\mathrm{Ker}(\partial_1)}{\mathrm{Im}(\partial_1)} \to \frac{\mathrm{Ker}(\partial_2)}{\mathrm{Im}(\partial_2)} \to \cdots \to \frac{\mathrm{Ker}(\partial_n)}{\mathrm{Im}(\partial_n)}.$$

For the absolute cohomology persistence module $\mathsf{H}^*(\mathbb{X})$, we define

$$\mathbb{C}^*: \quad C_1^* \leftarrow C_2^* \leftarrow \ldots \leftarrow C_n^*$$

where $C_i^* = \mathrm{Hom}(C_i, \mathbf{k}) = \langle \sigma_1^*, \sigma_2^*, \ldots, \sigma_i^* \rangle$, with $\{\sigma_i^*\}$ being the dual basis to $\{\sigma_i\}$. The coboundary $\delta = \partial^*$ is defined to be the adjoint to ∂. Then $\mathsf{C}^*(\mathbb{X}) = (\mathbb{C}^*, \delta)$ and

$$\mathsf{H}^*(\mathbb{X}) = \mathsf{H}(\mathbb{C}^*, \delta) = \mathrm{Ker}(\delta)/\,\mathrm{Im}(\delta).$$

Again, this is a persistence module (with arrows to the left).

Example. In our running example, the boundary map is given as follows:

$$\begin{aligned} \partial\sigma_1 &= \partial\sigma_2 = 0, \\ \partial\sigma_3 &= \partial\sigma_4 = \sigma_1 - \sigma_2, \\ \partial\sigma_5 &= \partial\sigma_6 = \sigma_3 - \sigma_4. \end{aligned}$$

This information is recorded in matrix D of Figure 1. The coboundary map is given as follows:

$$\begin{aligned} \delta\sigma_1^* &= -\delta\sigma_2 = \sigma_3^* + \sigma_4^*, \\ \delta\sigma_3^* &= -\delta\sigma_4^* = \sigma_5^* + \sigma_6^*, \\ \delta\sigma_5^* &= \quad \delta\sigma_6^* = 0. \end{aligned}$$

[2]For simplicity we generally suppress the homological grading within each C_i, which comes from the geometric dimensions of the cells associated to the generators. We will refer to it only when necessary.

This information is recorded in matrix $D^\perp$ of Figure 1.

The relative homology and cohomology persistence modules are defined as the homology of the persistence modules

$$\begin{aligned}(C_n/\mathbb{C}) \ : \quad & C_n \to (C_n/C_1) \ \to (C_n/C_2) \ \to \ \dots \ \to (C_n/C_{n-1}) \\ (C_n/\mathbb{C})^* : \quad & C_n^* \leftarrow (C_n/C_1)^* \leftarrow (C_n/C_2)^* \leftarrow \ \dots \ \leftarrow (C_n/C_{n-1})^*\end{aligned}$$

with boundary and coboundary maps induced from ∂, δ in the natural way. Thus

$$\begin{aligned}\mathsf{C}_*(X_\infty, \mathbb{X}) &= (C_n/\mathbb{C}, \partial), & \mathsf{C}^*(X_\infty, \mathbb{X}) &= ((C_n/\mathbb{C})^*, \delta). \\ \mathsf{H}_*(X_\infty, \mathbb{X}) &= \mathsf{H}(C_n/\mathbb{C}, \partial), & \mathsf{H}^*(X_\infty, \mathbb{X}) &= \mathsf{H}((C_n/\mathbb{C})^*, \delta).\end{aligned}$$

Remark. We note that the maps $\to$ of $\mathbb{C}$ and the maps $\leftarrow$ of $(C_n/\mathbb{C})^*$ are injective, whereas the maps $\leftarrow$ of $\mathbb{C}^*$ and the maps $\to$ of $(C_n/\mathbb{C})$ are surjective. In other words, absolute homology and relative cohomology are structurally akin to each other; and qualitatively different from absolute cohomology and relative homology. This is a symptom of the global duality mentioned in the introduction.

The main theorem of [1, 3] can be restated in the following way.

Theorem 2.6. *Given $\mathbb{C}, \partial$ as above, there exists a partition*

$$\{1, 2, \dots, n\} = F \sqcup G \sqcup H$$

with a bijective pairing $G \leftrightarrow H$, written as follows:

$$g \text{ is paired with } h \ \Leftrightarrow \ [g, h) \in \mathrm{Pairs} = \mathrm{Pairs}(\mathbb{C}, \partial).$$

Moreover, there is a new basis $\hat{\sigma}_1, \hat{\sigma}_2, \dots, \hat{\sigma}_n$ of C_n such that:

1. *$C_i = \langle \hat{\sigma}_1, \dots, \hat{\sigma}_i \rangle$ for all i.*
2. *$\partial \hat{\sigma}_f = 0$ for all $f \in F$.*
3. *$\partial \hat{\sigma}_h = \hat{\sigma}_g$, and hence $\partial \hat{\sigma}_g = 0$, for all $[g, h) \in \mathrm{Pairs}$.*

It follows that the persistence diagram $\mathrm{Pers}(\mathsf{H}(\mathbb{C}, \partial))$ consists of the intervals $[a_f, \infty)$ for $f \in F$ together with the intervals $[a_g, a_h)$ for $[g, h) \in \mathrm{Pairs}$. □

We note that item (1) is equivalent to the assertion that the leading term of each $\hat{\sigma}_i$ is σ_i (up to a nonzero scalar).

In the language of [1], the index set F identifies the positive simplices which remain unpaired, the index set G identifies the positive simplices which do get paired, and the index set H identifies the negative simplices. The vectors $\hat{\sigma}_f$ and $\hat{\sigma}_g$ are the cycles with leading terms σ_f and σ_g, and the vector $\hat{\sigma}_h$ is the chain with leading term σ_h which 'kills' the homology class of its paired $\hat{\sigma}_g$ by means of the equation $\partial \hat{\sigma}_h = \hat{\sigma}_g$.

Example. In our running example, $F = \{1, 6\}$, $G = \{2, 4\}$, $H = \{3, 5\}$ and $\mathrm{Pairs} = \{[2, 3), [4, 5)\}$. The new basis is

$$\begin{aligned}\hat{\sigma}_1 &= \sigma_1, & \hat{\sigma}_3 &= \sigma_3, & \hat{\sigma}_5 &= \sigma_5, \\ \hat{\sigma}_2 &= -\sigma_2 + \sigma_1, & \hat{\sigma}_4 &= -\sigma_4 + \sigma_3, & \hat{\sigma}_6 &= \sigma_6 - \sigma_5.\end{aligned}$$

The reader can easily verify that $\partial \hat{\sigma}_1 = \partial \hat{\sigma}_2 = \partial \hat{\sigma}_4 = \partial \hat{\sigma}_6 = 0$, that $\partial \hat{\sigma}_3 = \hat{\sigma}_2$, and $\partial \hat{\sigma}_5 = \hat{\sigma}_4$.

Proof of Proposition 2.4. The decomposition $\{1, 2, \dots, n\} = F \sqcup G \sqcup H$ and the new basis $\hat{\sigma}_1, \hat{\sigma}_2, \dots, \hat{\sigma}_n$ allow us to express $\mathbb{C}$ as a direct sum of persistent chain complexes:

$$\mathbb{C} = \bigoplus_{f \in F} \mathbb{C}_f \quad \oplus \quad \bigoplus_{[g,h) \in \mathrm{Pairs}} \mathbb{C}_{g,h}$$

where $\mathbb{C}_f = \langle \hat{\sigma}_f \rangle$ and $\mathbb{C}_{g,h} = \langle \hat{\sigma}_g, \hat{\sigma}_h \rangle$. Moreover, the boundary map ∂ respects this decomposition, mapping each summand into itself. We can therefore calculate $\mathsf{H}((C_n/\mathbb{C}), \partial)$ on each summand separately.

For summands of type $\mathbb{C}_f$, the persistence modules are constant over two phases, with index ranges $\{0, \dots, f-1\}$ and $\{f, \dots, n-1\}$. We can condense this information by representing them as two-term persistence modules (one term for each index range):

$$\begin{aligned} ((C_f)_n/\mathbb{C}_f): &\quad \langle \hat{\sigma}_f \rangle \to 0 \\ \mathrm{Ker}(\partial): &\quad \langle \hat{\sigma}_f \rangle \to 0 \\ \mathrm{Im}(\partial): &\quad 0 \to 0 \\ \mathsf{H} = \mathrm{Ker}(\partial)/\mathrm{Im}(\partial): &\quad \langle \hat{\sigma}_f \rangle \to 0 \end{aligned}$$

It follows that $\mathsf{H}((C_f)_n/\mathbb{C}_f)$ contributes the interval $[-\infty, a_f)$. This is generated by $[\hat{\sigma}_f]$ and hence has the same homological dimension as $[a_f, \infty)$ in $\mathrm{Pers}(\mathsf{H}(\mathbb{C}))$.

For summands of type $\mathbb{C}_{g,h}$ the persistence modules are constant over three phases, with index ranges $\{0, \dots, g-1\}$, $\{g, \dots, h-1\}$ and $\{h, \dots, n-1\}$:

$$\begin{aligned} ((C_{g,h})_n/\mathbb{C}_{g,h}): &\quad \langle \hat{\sigma}_g, \hat{\sigma}_h \rangle \to \langle \hat{\sigma}_h \rangle \to 0 \\ \mathrm{Ker}(\partial): &\quad \langle \hat{\sigma}_g \rangle \to \langle \hat{\sigma}_h \rangle \to 0 \\ \mathrm{Im}(\partial): &\quad \langle \hat{\sigma}_g \rangle \to 0 \to 0 \\ \mathsf{H} = \mathrm{Ker}(\partial)/\mathrm{Im}(\partial): &\quad 0 \to \langle \hat{\sigma}_h \rangle \to 0 \end{aligned}$$

It follows that $\mathsf{H}((C_{g,h})_n/\mathbb{C}_{g,h})$ contributes a single interval, $[a_g, a_h)$. This is generated by $[\hat{\sigma}_h]$, and hence has dimension one greater than $[a_g, a_h)$ in $\mathrm{Pers}(\mathsf{H}(\mathbb{C}))$, that being generated by $[\hat{\sigma}_g]$. □

The following table summarises the relationship between the three types of generator and the persistence intervals they generate.

generator	$\hat{\sigma}_f$	$\hat{\sigma}_g$	$\hat{\sigma}_h$
absolute homology	$[a_f, \infty)$	$[a_g, a_h)$	
relative homology	$[-\infty, a_f)$		$[a_g, a_h)_+$

The homological dimension of each interval is equal to the homological dimension of its generator. So, for any pair $[g, h) \in \mathrm{Pairs}$, the dimension of $[a_g, a_h)$ in $\mathsf{H}_*(X_\infty, \mathbb{X})$ is one greater than the dimension of $[a_g, a_h)$ in $\mathsf{H}_*(\mathbb{X})$. We indicate this in the table with a $_+$ subscript.

2.7. **Cohomology.** Persistent relative cohomology is structurally similar to persistent absolute homology. To make this apparent, let us introduce new notation, writing

$$\mathbb{C}^\perp: \quad C_1^\perp \to C_2^\perp \to \ \dots \ \to C_n^\perp$$

for the reverse of the sequence

$$(C_n/\mathbb{C})^*: \quad C_n^* \leftarrow (C_n/C_1)^* \leftarrow \ \dots \ \leftarrow (C_n/C_{n-1})^*,$$

so $C_i^\perp = (C_n/C_{n-i})^*$.

Recall that $\sigma_1^*, \dots, \sigma_n^*$ denotes the basis of C_n^* dual to the basis $\sigma_1, \dots, \sigma_n$ of C_n. If we write $\tau_i = \sigma_{n+1-i}^*$, then

$$C_i^\perp = (C_n/C_{n-i}) = \langle \sigma_n^*, \sigma_{n-1}^*, \dots, \sigma_{n+1-i}^* \rangle = \langle \tau_1, \tau_2, \dots, \tau_i \rangle$$

by elementary linear algebra. Moreover, if $\partial \sigma_j = \sum_{i<j} D_{ij} \sigma_i$ then $\delta \tau_j = \sum_{i<j} D_{ij}^\perp \tau_i$. where $D_{ij}^\perp = D_{(n-j),(n-i)}$.

On account of the formal similarity between $\mathbb{C}^\perp$ and $\mathbb{C}$ we conclude:

Corollary 2.7. *Suppose we have an algorithm which takes as input the sequence of cells $\sigma_1, \ldots, \sigma_n$ and their boundaries $\partial\sigma_1, \ldots, \partial\sigma_n$ and produces as output the persistent absolute homology of $\mathbb{X} = \sigma_1 \cup \cdots \cup \sigma_n$.*

Then the same algorithm applied to the sequence of formal cells $\tau_1, \ldots, \tau_n$ and coboundaries $\delta\tau_1, \ldots, \delta\tau_n$ computes the persistent relative cohomology of $\mathbb{X}$. □

Persistent absolute cohomology can be thought of as 'relative persistent relative cohomology'. More precisely, by elementary linear algebra:

Proposition 2.8. *The persistence module $(C_n^\perp/\mathbb{C}^\perp)$ is the reverse of $\mathbb{C}^*$, and the respective coboundary maps agree.* □

Corollary 2.9. *Suppose we have an algorithm which takes as input the sequence of cells $\sigma_1, \ldots, \sigma_n$ and their boundaries $\partial\sigma_1, \ldots, \partial\sigma_n$ and produces as output the persistent relative homology of $\mathbb{X} = \sigma_1 \cup \cdots \cup \sigma_n$.*

Then the same algorithm applied to the sequence of formal cells $\tau_1, \ldots, \tau_n$ and coboundaries $\delta\tau_1, \ldots, \delta\tau_n$ computes the persistent absolute cohomology of $\mathbb{X}$. □

Remark. We must transcribe the indices correctly for these two corollaries. If such an algorithm applied to the $\tau_i, \delta\tau_i$ produces a persistence interval $[p, q-1]$ for $\mathbb{C}^\perp$ then this is equivalent to $[n+1-q, n-p]$ for $(C_n/\mathbb{C})^*$, and hence to the half-open real interval $[a_{n+1-q}, a_{n+1-p})$.

Suppose now we apply Theorem 2.6 to $\mathbb{C}^\perp$ to obtain a partition $\{1, 2, \ldots, n\} = R \sqcup S \sqcup T$ and new generators $\hat{\tau}_i$, with $\delta\hat{\tau}_r = 0$, $\delta\hat{\tau}_s = 0$, and $\delta\hat{\tau}_t = \hat{\tau}_s$ for pairs $[s,t) \in \mathrm{Pairs}(\mathbb{C}^\perp, \delta)$. Then we obtain the following table:

generator	$\hat{\tau}_r$	$\hat{\tau}_s$	$\hat{\tau}_t$
relative cohomology	$[-\infty, a_{n+1-r})$	$[a_{n+1-t}, a_{n+1-s})_+$	
absolute cohomology	$[a_{n+1-r}, \infty)$		$[a_{n+1-t}, a_{n+1-s})$

By considering Proposition 2.3, we deduce that

$$R = n+1-F, \quad S = n+1-H, \quad T = n+1-G$$

and moreover $(s,t) \in \mathrm{Pairs}(\mathbb{C}^\perp, \delta)$ if and only if $(n+1-t, n+1-s) \in \mathrm{Pairs}(\mathbb{C}, \partial)$. Actually, this can also be inferred from the proof of the following proposition.

Proposition 2.10. *Let $\hat{\sigma}_1^*, \ldots, \hat{\sigma}_n^*$ denote the dual basis to $\hat{\sigma}_1, \ldots, \hat{\sigma}_n$, and write $\hat{\tau}_i = \hat{\sigma}_{n+1-i}^*$. Then the $\hat{\tau}_i$ are the generators described above (up to nonzero scalar multiples).*

Proof. By duality, we have $\delta\hat{\sigma}_f^* = 0$ for all $f \in F$, and $\delta\hat{\sigma}_g^* = \hat{\sigma}_h^*$ for all $[g,h) \in \mathrm{Pairs}(\mathbb{C})$. Moreover, σ_i^* is the *trailing* term of $\hat{\sigma}_i^*$. The proposition now follows, by bookkeeping. □

2.8. **A remark for the algebraically-minded.** According to [3], a persistence module can be regarded as a graded module over the ring $\mathbf{k}[t]$. In particular, $\mathbb{C}$ can be regarded as a free module over $\mathbf{k}[t]$ with n generators, $\sigma_1, \ldots, \sigma_n$, where σ_i has degree i. The boundary map $\partial : \mathbb{C} \to \mathbb{C}$ is then a homomorphism of graded modules.

The 'dual' of such a module can be taken with respect to the ground field $\mathbf{k}$ or the polynomial ring $\mathbf{k}[t]$. Thus we can define the global dual

$$\begin{aligned} \mathbb{C}^\circ &= \mathrm{Hom}_{\mathbf{k}[t]}(\mathbb{C}, \mathbf{k}[t]), \\ \text{where} \quad C_n^\circ &= \text{graded-module homomorphisms } \mathbb{C} \to \mathbf{k}[t] \text{ of degree } n; \end{aligned}$$

and the pointwise dual

$$\mathbb{C}^\dagger = \mathrm{Hom}_{\mathbf{k}}(\mathbb{C}, \mathbf{k}),$$
$$\text{where} \quad C_n^\dagger = \text{vector-space homomorphisms } C_{-n} \to \mathbf{k}.$$

These can be regarded as graded $\mathbf{k}[t]$-modules in a natural way. Moreover, the operations $-^\circ$ and $-^\dagger$ are contravariant functors, so in particular the boundary map on $\mathbb{C}$ induces boundary maps on the new modules.

The interested reader can verify that

$$\mathsf{H}(\mathbb{C}, \partial) = \text{persistent absolute homology of } \mathbb{X}$$
$$\mathsf{H}(\mathbb{C}^\dagger, \partial^\dagger) = \text{persistent absolute cohomology of } \mathbb{X}$$
$$\mathsf{H}(\mathbb{C}^\circ, \partial^\circ) = \text{persistent relative cohomology of } \mathbb{X}$$
$$\mathsf{H}(\mathbb{C}^{\circ\dagger}, \partial^{\circ\dagger}) = \text{persistent relative homology of } \mathbb{X}$$

up to calibrating the indices.

3. Matrix Algorithms

3.1. **The boundary matrix.** We can represent a filtered cell complex (at least, its homological information) by a strictly upper-triangular matrix D, whose entries $D[i, j]$ are the coefficients D_{ij} of the boundary map ∂ defined in Section 2.6. Thus the j-th column $D[j] = D[.., j]$ represents $\partial\sigma_j$. With the cells listed in the filtration order, the matrix D also encodes the filtration of the complex. Indeed, the top-left square submatrix $D[1..i, 1..i]$ is the boundary matrix for $X_i = \sigma_1 \cup \cdots \cup \sigma_i$, or, equivalently, the chain complex C_i. Thus D is a representation of the chain complex for persistent absolute homology, $\mathsf{C}_*(\mathbb{X}) = (\mathbb{C}, \partial)$.

If we flip the matrix D across its minor diagonal, we get the **anti-transpose** $D^\perp$, formally defined by $D^\perp[i, j] = D[n + 1 - j, n + 1 - i]$. Following the discussion in Section 2.7, we see that $D^\perp$ represents the cochain complex for persistent relative cohomology, $\mathsf{C}^*(X_\infty, \mathbb{X}) = ((C_n/\mathbb{C})^*, \delta)$. The top-left submatrix $D^\perp[1..i, 1..i]$ is the coboundary matrix for $\mathsf{C}^*(X_\infty, X_{n-i})$. Indeed, this is precisely the full coboundary matrix with entries in X_{n-i} removed.

It immediately follows that any procedure applied to matrix D that computes the intervals and generators of persistent absolute (resp. relative) homology, when applied to matrix $D^\perp$ will give us the intervals and generators of persistent relative (resp. absolute) cohomology.

3.2. **Persistence by matrix decomposition.** In [8], Cohen-Steiner et al. explain how to view the computation of persistent homology as a matrix decomposition problem, finding a factorization $D = RU$, where matrix R is **reduced** and U is invertible upper-triangular. Here we recap the relevant definitions.

For any matrix A, we define $\mathrm{low}_A(j)$ to be the index of the lowest non-zero entry in the j-th column of A (that is, the largest index i such that $A[i, j] \neq 0$); it is undefined if the column is zero. We say that matrix R is reduced if low_R is injective (over its domain of definition).

In what follows it is more convenient to look at the inverse of U, matrix $V = U^{-1}$. The decomposition $D = RU$ becomes $R = DV$. Whereas neither decomposition is unique, Cohen-Steiner et al. [8] show that the map low_R is. It is precisely this map that gives the persistence pairing: a class born at the step g of the filtration dies at the step h iff $\mathrm{low}_R(h) = g$.

Suppose we have a decomposition $R = DV$. If the column $R[i] = 0$ (so $\mathrm{low}_R(i)$ is undefined), then the column $V[i]$ is a cycle, by definition. Furthermore, since V is invertible upper-triangular, its diagonal entries are non-zero, and $V[i]$ is a cycle that does not exist in X_{i-1}, i.e. it is exactly the

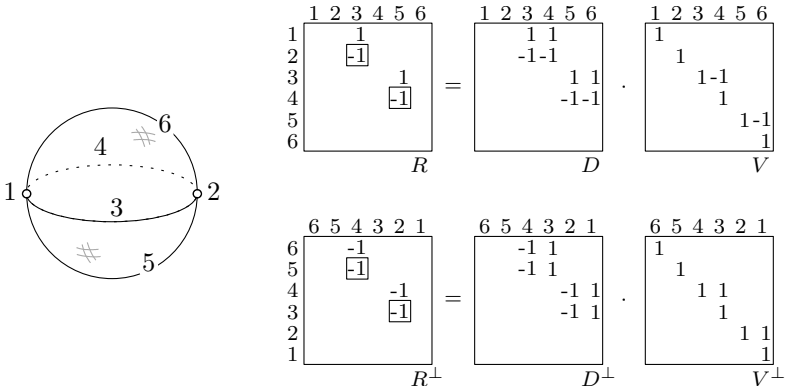


FIGURE 1. Filtration of a sphere, and the corresponding decompositions $R = DV$ and $R^\perp = D^\perp V^\perp$. Maps low_R and $\mathrm{low}_{R^\perp}$ are shown with squares around the entries. We show only the non-zero entries of the matrices.

cycle born at $\mathsf{H}_*(X_i)$. Similarly, if $R[j] \neq 0$, then it is the cycle that falls in the kernel of the map $\mathsf{H}_*(X_{j-1}) \to \mathsf{H}_*(X_j)$, and $V[j]$ is the chain that appears in X_j and has that cycle as its boundary.

3.3. **Homology generators.** To relate the matrix discussion to the algebra of the previous section, we observe that the new basis of Theorem 2.6 appears in the matrices R and V. The generator $\hat{\sigma}_f$ of the infinite interval $[a_f, \infty)$ is the column $V[f]$; the generator $\hat{\sigma}_g$ of the interval $[a_g, a_h)$ is the column $R[h]$; the chain $\hat{\sigma}_h$ that kills it is the column $V[h]$.

Example. See Figure 1 for the matrices R, D, V of our running example. The map low_R gives the absolute homology persistence intervals $\mathrm{Pers}(\mathsf{H}_*(\mathbb{S})) = \{[1,\infty)_0, [2,3)_0, [4,5)_1, [6,\infty)_2\}$ where the subscript indicates the dimension of the homology class. Columns of the matrices V and R give the cycles generating each of the intervals: they are $\hat{\sigma}_1 = V[1] = \sigma_1$, $\hat{\sigma}_2 = R[3] = \sigma_1 - \sigma_2$, $\hat{\sigma}_4 = R[5] = \sigma_3 - \sigma_4$, and $\hat{\sigma}_6 = V[6] = \sigma_6 - \sigma_5$, respectively.

In view of Proposition 2.4, we can also read off the generators for persistent relative homology: the column $V[f]$ is the generator for the interval $[-\infty, a_f)$, and the column $V[h]$ is the generator for $[a_g, a_h)$ (the chain that it represents becomes a relative cycle in $\mathsf{H}_*(X_\infty, X_g)$, and remains nonzero until $\mathsf{H}_*(X_\infty, X_h)$).

Example. The four intervals of $\mathrm{Pers}(\mathsf{H}_*(S_6, \mathbb{S})) = \{[-\infty, 1)_0, [2,3)_1, [4,5)_2, [-\infty, 6)_2\}$ have respective generators $\hat{\sigma}_1 = V[1] = \sigma_1$, $\hat{\sigma}_3 = V[3] = \sigma_3$, $\hat{\sigma}_5 = V[5] = \sigma_5$, and $\hat{\sigma}_6 = V[6] = \sigma_6 - \sigma_5$.

3.4. **Cohomology generators.** We now take the global dual, and consider persistent relative and absolute cohomology. This time we compute the decomposition $R^\perp = D^\perp V^\perp$ for the anti-transpose $D^\perp$ of D.

We must take care to track the indices correctly. As a matrix, the rows and columns of $D^\perp$ are labelled $\{1, \ldots, n\}$ in the usual order. However, row i and column i refer to cell σ_{n+1-i} in the original complex. If we define $i^* = n + 1 - i$, then we can think of the rows and columns as being labelled $\{n^*, \ldots, 1^*\}$, so that row i^* and column i^* do indeed refer to cell σ_i. The numerical labels in Figure 1 for $D^\perp, R^\perp, V^\perp$ should be thought of as starred labels.

The columns of $R^\perp$ and $V^\perp$ contain the cocycles of $\mathsf{H}^*(X_\infty, \mathbb{X})$ and the cochains that kill them. If $\mathrm{low}_{R^\perp}(g^*) = h^*$, then there is a finite interval $[a_g, a_h)$ generated by the cocycle $\hat{\sigma}^*_h = R^\perp[g^*]$ and killed by the cochain $\hat{\sigma}^*_g = V^\perp[g^*]$. If $R^\perp[f^*] = 0$, then there is an infinite interval $[-\infty, a_f)$ generated by $\hat{\sigma}^*_f = V^\perp[f^*]$.

Example. The persistent relative cohomology $\mathrm{Pers}(\mathsf{H}^*(S_6, \mathbb{S}))$ has four intervals $[-\infty, 6)_2$, $[4, 5)_2$, $[2, 3)_1$, $[-\infty, 1)_0$, generated respectively by $\hat{\sigma}_6^* = V^\perp[6^*] = \sigma_6^*$, $\hat{\sigma}_5^* = R^\perp[4^*] = -\sigma_6^* - \sigma_5^*$, $\hat{\sigma}_3^* = R^\perp[2^*] = -\sigma_3^* - \sigma_2^*$, $\hat{\sigma}_1^* = V^\perp[1^*] = \sigma_1^* + \sigma_2^*$.

Finally, we can read off the generators for persistent absolute cohomology $\mathsf{H}^*(\mathbb{X})$: when $R^\perp[f^*] = 0$, the column $V^\perp[f^*]$ is the cocycle which generates the interval $[-\infty, a_f)$; and when $\mathrm{low}_{R^\perp(g^*)} = h^*$, the column $V^\perp[g^*]$ is the cocycle which generates $[a_g, a_h)$.

Example. The persistent absolute cohomology $\mathrm{Pers}(\mathsf{H}^*(\mathbb{S}))$ has four intervals $[6, +\infty)_2$, $[4, 5)_1$, $[2, 3)_0$, $[1, +\infty)_0$ generated respectively by $\hat{\sigma}_6^* = V^\perp[6^*] = \sigma_6^*$, $\hat{\sigma}_4^* = V^\perp[4^*] = \sigma_4^*$, $\hat{\sigma}_2^* = V^\perp[2^*] = \sigma_2^*$, $\hat{\sigma}_1^* = V^\perp[1^*] = \sigma_1^* + \sigma_2^*$.

3.5. **Column and row algorithms.** The original persistence algorithm [1, 3] finds the pairing by processing matrix D column-by-column to obtain the reduced matrix R. In the context of $R = DV$ decomposition, one can express it as:

Algorithm 1 Column algorithm pHcol.

```
R = D; V = I
for i = 1 to n do
  while ∃ j < i with low_R(j) = low_R(i) do
    c = R[i][low_R i]/R[j][low_R j]
    R[i] = R[i] − cR[j]
    V[i] = V[i] − cR[j]
```

Here the definition of the constant c ensures that the lowest non-zero element in column i moves up after each iteration of the while loop. The condition of the while loop immediately implies that matrix R is reduced when the algorithm terminates. Furthermore, since we perform identical updates on R and V, we get an $R = DV$ decomposition.

The algorithm pHcol is essentially Gaussian elimination performed using column operations. More commonly one would use column operations, processing the matrix row-by-row from the bottom up:

Algorithm 2 Row algorithm pHrow.

```
R = D; V = I
for i = n down to 1 do
  indices = [j | low_R(j) = i]        # lows in the row i of R
  p = indices[0]                      # pivot
  for j ∈ indices[1..] do
    c = R[j][low_R j]/R[p][low_R p]
    R[j] = R[j] − cR[p]
    V[j] = V[j] − cV[p]
```

It is not difficult to see that this algorithm also produces an $R = DV$ decomposition where matrix R is reduced, and matrix V is invertible upper-triangular. What is less immediate is that the two algorithms produce identical decompositions, so we prove this fact formally. (Notice that the statement would not be true if, during pHrow, we tried to cancel all non-zero elements in row i of R, rather than restricting attention to the columns picked out by indices.)

Theorem 3.1 (Identical Output Theorem)**.** *The decompositions $R_c = DV_c$ and $R_r = DV_r$ produced by column and row algorithms respectively are identical, i.e. $R_c = R_r$ and $V_c = V_r$.*

Proof. We observe that once it determines the lowest non-zero element in a given column of matrix R, neither algorithm changes that column in any subsequent operations. Given a matrix $R = D$ we prove the claim by induction. The first column with the lowest non-zero entry in R is not modified by either algorithm. Suppose that the columns with the lowest non-zero entries below i are identical in both R_c and R_r, and V_c and V_r. During the computation of the column with the lowest non-zero entry in row i we add columns with $\mathrm{low}_R > i$ in a decreasing order dictated by the lowest non-zero entry of the column. Since the order and the columns are identical, so is the result. □

Remark. Recently Milosavljevic, Morozov, and Skraba [11] showed that one can compute persistence in matrix multiplication time.

Remark. One can apply the two algorithms of this section to the restricted matrix D_p that gives only the boundaries of the p-dimensional cells. We can still extract some information from the $R_p = D_pV_p$ decomposition of this matrix: the finite intervals $[g, h)$ in dimension $p - 1$ and the births in dimension p, i.e. the endpoints g or h of any p-dimensional interval.

4. Optimizations

4.1. **Cohomology algorithm.** One of our goals has been to relate our present work to an algorithm pCoh for persistent absolute cohomology that we described in [6]. We based that algorithm on the idea of maintaining a *right filtration* (defined in [12]); as a result it looks different from pHcol and pHrow above. In fact, we now show that one can view pCoh as an optimization of pHrow applied to the matrix $D^\perp$. We begin by reviewing the algorithm:

Algorithm 3 Cohomology algorithm pCoh.

```
Z⊥ = [], birth = []
for i = 1 to n do
  indices = [j | σ*_i ∈ δz*_j, z*_j unmarked in Z⊥]
  if indices are empty then
    prepend σ*_i to Z⊥ and i to birth
  else
    prepend a marked σ*_i to Z⊥ and i to birth
    p = indices[0]
    for j = 1 to size(indices) do
      c = (δZ⊥[indices[j]])[i]/(δZ⊥[p])[i]
      Z⊥[indices[j]] = Z⊥[indices[j]] − cZ⊥[p]
    mark Z⊥[p] and output the pair [birth[p], i)
```

List $Z^\perp$ maintains the cocycle basis for $H^*(X_i)$ in the right filtration order dictated by the filtration of the space. The marking above is for exposition only, in practice we drop a cocycle from the list $Z^\perp$ as soon as it dies. When a new cell σ_i enters, it is necessarily a cocycle (since it has no cofaces), but it may fall into a coboundary of a former cocycle, in which case (the else clause) we update the right filtration and drop the cocycle that σ_i kills.

To see that this algorithm is a variation of the row algorithm from the previous section, observe that the cocycles that it maintains are stored in the bottom-right corner of matrix $V^\perp$ during the execution of the row algorithm.

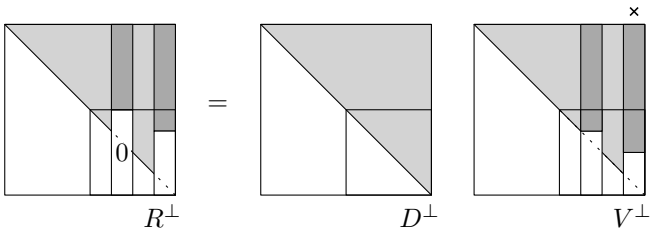


FIGURE 2. The structure of matrices $R^\perp = D^\perp V^\perp$ during the execution of the row algorithm.

Claim 4.1. *The matrix $Z^\perp$ in the cohomology algorithm after iteration i is equal to the bottom-right corner of the matrix $V^\perp[(n-i)..n, (n-i)..n]$ after the i-th iteration of the row algorithm.*

Proof. We prove the claim inductively. Denoting with $R_i^\perp, V_i^\perp, Z_i^\perp$ the various matrices after i iterations of both algorithms, assume the unmarked cocycles z_j^* in $Z_i^\perp$ are exactly the cocycles with $\operatorname{low}_{R^\perp}(j) > i$. In other words, the corresponding columns $R_i^\perp[j] = 0$. Furthermore assume that the two matrices are identical, i.e. $V_i^\perp = Z_i^\perp$. The claim is true when $i = 0$. Our goal is to show it is true for $i = k$ assuming it is true for $i = k - 1$.

At the k-th iteration, if cell σ_k^* does not appear in the coboundary of any cocycle, then its row in $R_{k-1}^\perp = \delta V_{k-1}^\perp = \delta Z_{k-1}^\perp$ is zero. It follows that it is not in the image of the map $\operatorname{low}_{R_{k-1}^\perp}$ and therefore neither algorithm performs any changes, so $V_k^\perp = Z_k^\perp$, and unmarked cocycles remain as claimed.

If cell σ_k^* is in the coboundary of a cocycle z_j^* then k is in the image $\operatorname{im}\operatorname{low}_{R_{k-1}^\perp}$. Moreover, from the inductive hypothesis the indices j of the columns of $R_{k-1}^\perp$ that have $\operatorname{low}_{R_{k-1}^\perp}(j) = i$ are exactly the unmarked cocycles in $Z_{k-1}^\perp$ that have σ_k^* in their coboundary. Therefore, the update performed by both algorithms is identical. □

Remark. Since the matrix R contains the final persistence pairing, expressed as the map low_R, the algorithm pHcol is commonly optimized to keep track only of this matrix (and ignore matrix V). In contrast, pCoh maintains only matrix $Z^\perp = V^\perp$.

4.2. **Practice.** The algorithm pCoh above highlights the difference between the column and the row versions of the persistence algorithm. pHcol stores all the dead cycles since it has no choice: any of them might be required at some future point in the reduction. pHrow, on the other hand, is able to 'examine the future' by inspecting any chosen row. It is therefore free to drop a column once it has determined its pairing and used it in the update. pCoh does so explicitly.

In practice, such row access may be difficult when computing homology: it requires quick access to the coboundary of a given cell (since that is what a row of D is). In simplicial complex implementations it is common to represent simplices as lists of vertices; then their boundary maps are easy to compute on the fly, while their coboundaries require a full preprocessing of the entire boundary matrix. By switching to cohomology we turn the tables: all the primitives necessary for the row algorithm (and in particular the optimized version given in this section) are readily available.

4.3. **Experiments.** The practical improvement resulting from these observations is startling. In the following table we compare the traditional persistent homology algorithm pHcol with the cohomology algorithm pCoh. We list the total number of operations performed (in terms of primitive operations during chain arithmetic), total running time, and peak space usage in terms of the number of elements stored.

Dataset	Algorithm	Operations	Time	Peak elements
M-50	pCoh	2,171,909,275	106 s	575,758
	pHcol	609,477,028,616	4160 s	6,461,866
T-10,000	pCoh	55,930,317	6 s	22,629
	pHcol	29,760,159,689	207 s	693,031

We used the C++ library Dionysus [13] to perform the above experiments. The homology algorithm pHcol in the above table computes only the matrix R since it suffices to extract the barcode. It also uses the original optimization of [1] and stores the non-zero coefficients only in the rows that correspond to the positive cells. M-50 is a filtration of an 8-skeleton of a Rips complex built on 50 random points of a Mumford dataset [14, 15] up to the maximum pairwise distance of 1.5; the largest complex consists of 663,901 simplices. T-10,000 is an alpha shape filtration of 10,000 points sampled on a torus embedded in $\mathbf{R}^3$; the size of the Delaunay triangulation is 557,727 simplices. The speed-up is encouraging. We would like to point out that these examples are not cherry-picked: we have yet to find a filtration on which pHcol is the faster of the two.

Conclusion. When combined, the algebraic and experimental observations suggest that if given a choice, one is better off using the cohomology algorithm. Most of the time one has such a choice: for example, when computing only the persistence diagram.

References

[1] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. *Discrete and Computational Geometry*, 28(4):511–533, 2002.

[2] Gunnar Carlsson. Topology and data. *American Mathematical Society*, 46(2):255–308, 2009.

[3] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete and Computational Geometry*, 33(2):249–274, 2005.

[4] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. *Discrete and Computational Geometry*, 37(1):103–120, 2007.

[5] Frédéric Chazal, David Cohen-Steiner, Marc Glisse, Leo Guibas, and Steve Oudot. Proximity of persistence modules and their diagrams. In *Proceedings of the Annual Symposium on Computational Geometry*, pages 237–246, 2009.

[6] Vin de Silva, Dmitriy Morozov, and Mikael Vejdemo-Johansson. Persistent cohomology and circular coordinates. *Discrete and Computational Geometry*, 45(4):737–759, 2011.

[7] Vin de Silva and Robert Ghrist. Coordinate-free coverage in sensor networks with controlled boundaries via homology. *International Journal of Robotics Research*, 25(12):1205–1222, December 2006.

[8] David Cohen-Steiner, Herbert Edelsbrunner, and Dmitriy Morozov. Vines and vineyards by updating persistence in linear time. In *Proceedings of the Annual Symposium on Computational Geometry*, pages 119–126, 2006.

[9] Allen Hatcher. *Algebraic topology*. Cambridge University Press, 2002.

[10] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Extending persistence using poincaré and lefschetz duality. *Foundations of Computational Mathematics*, 9(1):79–103, 2009.

[11] Nikola Milosavljevic, Dmitriy Morozov, and Primoz Skraba. Zigzag persistent homology in matrix multiplication time. In *Proceedings of the Annual Symposium on Computational Geometry*, pages 216–225, 2011.

[12] Gunnar Carlsson and Vin de Silva. Zigzag persistence. *Foundations of Computational Mathematics*, 10(4):367–405, 2010.

[13] Dmitriy Morozov. Dionysus library for computing persistent homology. `http://www.mrzv.org/software/dionysus`.

[14] Gunnar Carlsson, Tigran Ishkhanov, Vin de Silva, and Afra Zomorodian. On the local behavior of spaces of natural images. *International Journal of Computer Vision*, 76(1):1–12, January 2008.

[15] Ann B. Lee, Kim S. Pedersen, and David Mumford. The nonlinear statistics of high-contrast patches in natural images. Technical Report APPTS #01-3, Division of Applied Mathematics Brown University, December 2001.