





Efficient Algorithms for Complexes of Persistence Modules with Applications

Tamal K. Dey  

Department of Computer Science, Purdue University, West Lafayette, IN, USA

Florian Russold 

Institute of Geometry, Graz University of Technology, Austria

Shreyas N. Samaga  

Department of Computer Science, Purdue University, West Lafayette, IN, USA

Abstract

We extend the persistence algorithm, viewed as an algorithm computing the homology of a complex of free persistence or graded modules, to complexes of modules that are not free. We replace persistence modules by their presentations and develop an efficient algorithm to compute the homology of a complex of presentations. To deal with inputs that are not given in terms of presentations, we give an efficient algorithm to compute a presentation of a morphism of persistence modules. This allows us to compute persistent (co)homology of instances giving rise to complexes of non-free modules. Our methods lead to a new efficient algorithm for computing the persistent homology of simplicial towers and they enable efficient algorithms to compute the persistent homology of cosheaves over simplicial towers and cohomology of persistent sheaves on simplicial complexes. We also show that we can compute the cohomology of persistent sheaves over arbitrary finite posets by reducing the computation to a computation over simplicial complexes.

2012 ACM Subject Classification Theory of computation → Computational geometry; Mathematics of computing → Algebraic topology

Keywords and phrases Persistent (co)homology, Persistence modules, Sheaves, Presentations

Digital Object Identifier 10.4230/LIPIcs.SoCG.2024.51

Related Version *Full Version:* <https://arxiv.org/abs/2403.10958>

Supplementary Material *Software (Source Code):* <https://github.com/TDA-Jyamiti/Algos-cplx-pers-modules/> [12], archived at [swh:1:dir:6c13c2c3aeb94cc68377d695005250d1ab892cb7](https://www.swh.io/dir/6c13c2c3aeb94cc68377d695005250d1ab892cb7)

Funding This work is supported partially by NSF grants CCF 2049010 and 2301360 and by the Austrian Science Fund (FWF): W1230.

1 Introduction

The theory of persistence, a central building block of topological data analysis, is concerned with the study of persistent objects and their persistent homology. A *persistent object* $\vec{A}: \mathbb{N}_0 \rightarrow \mathbf{A}$ is a sequence of objects

$$\vec{A}: A_0 \xrightarrow{f_0} A_1 \xrightarrow{f_1} A_2 \xrightarrow{f_2} \dots \quad (1)$$

in a category \mathbf{A} with morphisms f_i . Let \mathbb{k} be a field. Assuming that there is a chain complex of \mathbb{k} -vector spaces $C_\bullet(A_i)$ (in a category denoted $\mathbf{Ch}(\mathbf{A})$) associated to these objects inducing their homology, a persistent object provides a persistent chain complex $C_\bullet(\vec{A}): \mathbb{N}_0 \rightarrow \mathbf{Ch}(\mathbf{A})$ ((2) right) or equivalently a chain complex of persistence modules ((2) left).



© Tamal K. Dey, Florian Russold, and Shreyas N. Samaga;

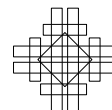
licensed under Creative Commons License CC-BY 4.0

40th International Symposium on Computational Geometry (SoCG 2024).

Editors: Wolfgang Mulzer and Jeff M. Phillips; Article No. 51; pp. 51:1–51:18

Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



$$\begin{array}{ccccccc}
 C_{k+1}(\vec{A}) : & C_{k+1}(A_0) & \xrightarrow{C_{k+1}(f_0)} & C_{k+1}(A_1) & \xrightarrow{C_{k+1}(f_1)} & C_{k+1}(A_2) & \xrightarrow{C_{k+1}(f_2)} \dots \\
 \downarrow \partial_{k+1} & \downarrow \partial_{k+1}^0 & & \downarrow \partial_{k+1}^1 & & \downarrow \partial_{k+1}^2 & \\
 C_k(\vec{A}) : & C_k(A_0) & \xrightarrow{C_k(f_0)} & C_k(A_1) & \xrightarrow{C_k(f_1)} & C_k(A_2) & \xrightarrow{C_k(f_2)} \dots \\
 \downarrow \partial_k & \downarrow \partial_k^0 & & \downarrow \partial_k^1 & & \downarrow \partial_k^2 & \\
 C_{k-1}(\vec{A}) : & C_{k-1}(A_0) & \xrightarrow{C_{k-1}(f_0)} & C_{k-1}(A_1) & \xrightarrow{C_{k-1}(f_1)} & C_{k-1}(A_2) & \xrightarrow{C_{k-1}(f_2)} \dots
 \end{array} \quad (2)$$

We call the sequence $C_{k+1}(\vec{A}) \xrightarrow{\partial_{k+1}} C_k(\vec{A}) \xrightarrow{\partial_k} C_{k-1}(\vec{A})$ where $C_k(\vec{A})$ is a persistence module and $\partial_k \circ \partial_{k+1} = 0$ a *complex of persistence modules*. The *persistent homology* $H_k(\vec{A}) := \ker \partial_k / \text{im } \partial_{k+1}$ of the complex of persistence modules (2) is given by the persistence module

$$H_k(\vec{A}) : H_k(A_0) \xrightarrow{H_k(f_0)} H_k(A_1) \xrightarrow{H_k(f_1)} H_k(A_2) \xrightarrow{H_k(f_2)} \dots$$

where $H_k(A_i) := \ker \partial_k^i / \text{im } \partial_{k+1}^i$ and $H_k(f_i)$ are induced by the chain maps $C_k(f_i)$. The goal of this paper is to present an efficient general purpose algorithm to compute the homology of a complex of persistence modules and demonstrate some of its applications.

The inspiration for our approach comes from the well-known persistence algorithm [13]. Suppose the persistent object in (1) is a finite filtration of simplicial complexes, i.e. A_i is a finite simplicial complex, f_i is an inclusion and $C_\bullet(\vec{A})$ is the complex of persistent simplicial chains. In [28] the authors observed that a persistence module can equivalently be viewed as a graded module over the polynomial ring $\mathbb{k}[t]$. Using this perspective, $C_k(\vec{A})$ is a free module with a basis given by the k -simplices in the filtration and the boundary morphisms ∂_k can be represented by matrices w.r.t. these bases. The persistence algorithm leverages this compressed representation of $C_\bullet(\vec{A})$ by its generators and computes the persistent homology of the whole filtration at once. This makes it much more efficient than naively computing $H_k(A_i)$ for each index i , where simplices would be considered multiple times.

If we allow the f_i 's to be arbitrary simplicial maps, we obtain what is called a *simplicial tower*. It turns out that this apparently simple modification brings a significant change at the algebraic level because the persistence modules $C_k(\vec{A})$ in the complex may no longer remain free (relations among generators may appear). Thus, in general, they do not admit a basis and we can not straightforwardly represent ∂_k by matrices and compute the homology using linear algebra over $\mathbb{k}[t]$. In [11, 20] the authors tackle this algebraic difficulty on a topological level, by expanding a tower into a filtration and then applying efficient algorithms for filtrations. We tackle the problem directly at the algebraic level by designing algorithms that can handle complexes of non-free modules. These algorithms enable a further generalization obtained by additionally considering algebraic information over a simplicial tower. If this algebraic information is provided by a cosheaf, we obtain the case of persistent cosheaf homology [23]. The persistent cosheaf homology is again the homology of a complex of not necessarily free persistence modules. But cosheaf homology is not a homotopy invariant, so the methods for the plain tower [11, 20] do not work in this case. Here we are forced to tackle the problem at an algebraic level.

One of our main observations is that we can compute the homology of a complex of non-free modules if we consider relations in addition to the generators. This brings *presentations* of modules into the picture which are morphisms from the free modules of relations to the free modules of generators. So, we design: 1) an efficient algorithm (Section 4) to compute

a presentation of a morphism of persistence modules by free modules which allows us to convert a complex of persistence modules into a *complex of presentations*, and 2) an efficient algorithm (Section 3) to compute the homology of a complex of presentations. In the spirit of the persistence algorithm our method considers a generator only once even though it may exist over a wide range of indices. In fact, our algorithm is a direct generalization of the persistence algorithm to which it specializes in the case of free modules.

At this point, we note that we are not aware of any computational approach in the TDA literature that deals with complexes of persistence modules and complexes of presentations in the full generality as our approach does. The closest along this line is the recent work in [24] where the author introduces a new framework of barcode bases and operations on them to compute a barcode basis of the homology of a complex of persistence modules in the context of distributed persistent homology computations. Our presentation algorithm does not require such a specific barcode form and can process general presentations. The paper also does not discuss how to compute barcode bases and maps between them from general morphisms of persistence modules which we address.

Applications. A central motivation of this paper is to provide computational tools for the ever-growing body of ideas to use methods from algebraic topology in applications. In recent years, the idea of using methods from sheaf theory in applications has gained traction in the field of TDA [3, 8, 15, 19, 22]. Instead of just considering a space by itself, sheaves allow us to study the behaviour of data over a space. Sheaves and their cohomology have been used in various applications, see e.g. [2, 4, 6, 16, 17, 18, 21]. Persistent versions of sheaves and cosheaves have also appeared in the TDA literature, see e.g. [10, 25, 26]. Moreover persistent (co)sheaves have been used as a framework for a distributed computation of (persistent) homology [9, 24, 27]. In [23], a general theory of persistent sheaf cohomology has been developed for which this paper establishes a complete computational framework.

We have already mentioned that our approach provides a novel efficient algorithm to compute the barcode of a given tower (Section 6.1). We demonstrate that it provides efficient algorithms to compute various flavors of persistent cosheaf homology and sheaf cohomology. We consider the persistent homology of a cosheaf over a varying simplicial complex (Section 6.2) and the cohomology of a persistent sheaf on a fixed simplicial complex (Section 6.3). We also show that we can reduce the computation of persistent sheaf cohomology over an arbitrary finite poset to a computation over a simplicial complex (Section 6.4).

We have a preliminary implementation of the two basic algorithms mentioned before. Experimental results suggest that our approach is not merely theoretical, but has the potential to be useful in practice (<https://github.com/TDA-Jyamiti/Algos-cplx-pers-modules/>).

2 Persistence modules, graded modules, and presentations

In this section we recall basic notions of persistence modules, graded modules and their presentations. A persistence module, as depicted in the top row of (3), is a functor $M: \mathbb{N}_0 \rightarrow \mathbf{vec}$ where \mathbf{vec} denotes the category of finite dimensional vector spaces. It is of *finite type*, if there exists an $m \in \mathbb{N}_0$ such that $M(i \leq j)$ is an isomorphism for all $i \geq m$. A morphism of persistence modules $\phi: M \rightarrow N$, as depicted in (3),

$$\begin{array}{ccccccc}
 M : & M(0) & \xrightarrow{M(0 \leq 1)} & M(1) & \xrightarrow{M(1 \leq 2)} & \dots & \xrightarrow{M(m-1 \leq m)} & M(m) & \xrightarrow{M(m \leq m+1)} & \dots \\
 \phi \downarrow & \downarrow \phi(0) & & \downarrow \phi(1) & & & & \downarrow \phi(m) & & \\
 N : & N(0) & \xrightarrow{N(0 \leq 1)} & N(1) & \xrightarrow{N(1 \leq 2)} & \dots & \xrightarrow{N(m-1 \leq m)} & N(m) & \xrightarrow{N(m \leq m+1)} & \dots
 \end{array} \tag{3}$$

is a natural transformation of functors $\mathbb{N}_0 \rightarrow \mathbf{vec}$. Let \mathbf{pMod} denote the category of persistence modules of finite type. An \mathbb{N}_0 -graded $\mathbb{k}[t]$ -module M is a direct sum of \mathbb{k} -vector spaces $\bigoplus_{i \in \mathbb{N}_0} M_i$ with the usual \mathbb{k} -action on the summands and a t^i -action for all $i \in \mathbb{N}_0$ such that $t^i \cdot M_j \subseteq M_{i+j}$. If $m \in M_i$ it is called a homogeneous element and we define its *degree* by $\deg(m) := i$. A (homogeneous) morphism $\phi: M \rightarrow N$ of graded $\mathbb{k}[t]$ -modules M and N is a map of $\mathbb{k}[t]$ -modules such that $\phi(M_i) \subseteq N_i$. Let $\mathbf{grMod}_{\mathbb{k}[t]}$ denote the category of finitely generated \mathbb{N}_0 -graded $\mathbb{k}[t]$ -modules. We know the following equivalence which allows us to use persistence modules and graded modules interchangeably in the following.

► **Proposition 1** ([7, Corollary 10][28, Theorem 3.1]). $\mathbf{pMod} \cong \mathbf{grMod}_{\mathbb{k}[t]}$.

In what follows, we assume that all persistence modules and graded modules are of finite type or finitely generated, respectively. Persistence modules as well as morphisms between them can be represented by collections of matrices. Their equivalent counterpart graded $\mathbb{k}[t]$ -modules can not be handled by matrices directly since they can have torsion in general. To handle them with methods of linear algebra over $\mathbb{k}[t]$ we use presentations.

► **Definition 2** (Presentation of module). *Let M be a finitely generated graded $\mathbb{k}[t]$ -module. A presentation of M is an exact sequence of the form*

$$0 \longleftarrow M \xleftarrow{\mu} P_0 \xleftarrow{p} P_1$$

where P_0 and P_1 are free finitely generated graded $\mathbb{k}[t]$ -modules. We call a presentation reduced, if p is a monomorphism.

► **Definition 3** (Presentation of morphism). *Let $\phi: M \rightarrow N$ be a morphism of finitely generated graded $\mathbb{k}[t]$ -modules M and N . A presentation of ϕ is a commutative diagram*

$$\begin{array}{ccccccc} 0 & \longleftarrow & M & \xleftarrow{\mu} & P_0 & \xleftarrow{p} & P_1 \\ & & \downarrow \phi & & \downarrow f_0 & & \downarrow f_1 \\ 0 & \longleftarrow & N & \xleftarrow{\nu} & Q_0 & \xleftarrow{q} & Q_1 \end{array} \quad (4)$$

where the rows are presentations of M and N respectively.

In the following, we also refer to the morphism of free modules $P_0 \xleftarrow{p} P_1$ in Definition 2 as a presentation of M . By the exactness assumption $M \cong \operatorname{coker} p$ and $\mu \cong (P_0 \xrightarrow{\pi} \operatorname{coker} p)$. We also refer to the right square in (4) as a morphism of presentations.

Given $a < b \in \mathbb{N}_0 \cup \{\infty\}$, let $\mathbb{I}_{[a, \infty)}$ denote the free graded $\mathbb{k}[t]$ -module generated by a single generator of degree a . Moreover, we denote by $\mathbb{I}_{[a, b)}$ the quotient module $\mathbb{I}_{[a, \infty)} / \mathbb{I}_{[b, \infty)}$. By the equivalence of Proposition 1, $\mathbb{I}_{[a, b)}$ corresponds to the indecomposable interval persistence module starting at index a and ending at index b . By the Theorems of Krull-Remak-Schmidt [1, Theorem 1] and Gabriel [14, Chapter 2.2] and Proposition 1, every finitely generated graded $\mathbb{k}[t]$ -module M is isomorphic to a finite direct sum of indecomposable interval modules $M \cong \bigoplus_{i=1}^d \mathbb{I}_{[a_i, b_i)}$. We call the multiset of intervals $\{[a_i, b_i) \mid 1 \leq i \leq d\}$ the *barcode* of M . For two interval modules $\mathbb{I}_{[a, b)}$ and $\mathbb{I}_{[c, d)}$ we have

$$\operatorname{Hom}(\mathbb{I}_{[a, b)}, \mathbb{I}_{[c, d)}) \cong \begin{cases} \mathbb{k} & \text{if } c \leq a < d \leq b \\ 0 & \text{else} \end{cases} . \quad (5)$$

In other words, if two bars overlap in a certain way as stated in (5), then, for every scalar $\lambda \in \mathbb{k}$, there is a unique morphism defined by multiplication with λ where the bars overlap.

Moreover, given two finitely generated graded modules M and N , we have

$$\text{Hom}(M, N) \cong \text{Hom}\left(\bigoplus_i \mathbb{I}_{[a_i, b_i]}, \bigoplus_j \mathbb{I}_{[c_j, d_j]}\right) \cong \bigoplus_i \bigoplus_j \text{Hom}(\mathbb{I}_{[a_i, b_i]}, \mathbb{I}_{[c_j, d_j]}). \tag{6}$$

We call the following reduced presentation of $\mathbb{I}_{[a, b]}$ an *elementary presentation*

$$0 \longleftarrow \mathbb{I}_{[a, b]} \longleftarrow \mathbb{I}_{[a, \infty]} \xleftarrow{\cdot t^{b-a}} \mathbb{I}_{[b, \infty]} \longleftarrow 0 \tag{7}$$

where we set $\mathbb{I}_{[b, \infty]}$ and the map to zero if $b = \infty$. The direct sum of presentations is defined by the pointwise direct sum of graded modules. Thus, assuming $b_i \neq \infty$ iff $1 \leq i \leq d' \leq d$, we obtain the following reduced presentation of M (up to isomorphism)

$$0 \longleftarrow \bigoplus_{i=1}^d \mathbb{I}_{[a_i, b_i]} \longleftarrow \bigoplus_{i=1}^d \mathbb{I}_{[a_i, \infty]} \xleftarrow{p} \bigoplus_{i=1}^{d'} \mathbb{I}_{[b_i, \infty]} \longleftarrow 0$$

as a direct sum of elementary presentations where p is of the form

$$p = \begin{pmatrix} t^{b_1-a_1} & 0 & \dots & 0 \\ 0 & t^{b_2-a_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & t^{b_{d'}-a_{d'}} \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{pmatrix}. \tag{8}$$

We call a presentation *canonical* if it is a direct sum of elementary presentations and has no elementary summands (7) with $a = b$. The matrix p of a canonical presentation has a form as in (8), i.e. every relation in P_1 maps to a unique generator in P_0 . We obtain a natural inclusion and projection of elementary summands, i.e. a commutative diagram of the form:

$$\begin{array}{ccccccc} 0 & \longleftarrow & \mathbb{I}_{[a_i, b_i]} & \longleftarrow & \mathbb{I}_{[a_i, \infty]} & \xleftarrow{\cdot t^{b_i-a_i}} & \mathbb{I}_{[b_i, \infty]} & \longleftarrow & 0 \\ & & \downarrow \iota & & \downarrow \iota_0 & & \downarrow \iota_1 & & \\ 0 & \longleftarrow & \bigoplus_l \mathbb{I}_{[a_l, b_l]} & \longleftarrow & \bigoplus_l \mathbb{I}_{[a_l, \infty]} & \xleftarrow{p} & \bigoplus_l \mathbb{I}_{[b_l, \infty]} & \longleftarrow & 0 \\ & & \downarrow \pi & & \downarrow \pi_0 & & \downarrow \pi_1 & & \\ 0 & \longleftarrow & \mathbb{I}_{[a_j, b_j]} & \longleftarrow & \mathbb{I}_{[a_j, \infty]} & \xleftarrow{\cdot t^{b_j-a_j}} & \mathbb{I}_{[b_j, \infty]} & \longleftarrow & 0 \end{array}$$

Presentations of finitely generated graded $\mathbb{k}[t]$ -modules and their morphisms can be represented by the matrices corresponding to the morphisms of free modules p, q, f_0, f_1 in (4) with labeled rows and columns recording the degree of the generators. When recording the degree of the generators, we can use coefficients in \mathbb{k} instead of coefficients in $\mathbb{k}[t]$ since the degree of column and row determines the polynomial factor t^r of an entry. If we have a morphism of presentations as in (4) where p and q are in canonical form, then, by the structure of p and q and by commutativity, f_1 is uniquely determined by f_0 . To represent such a morphism of presentations it is enough to store the matrix f_0 and for each row and column the degree of the generator in P_0 or Q_0 and its unique corresponding relation in P_1 or Q_1 .

3 Computing homology of complexes of presentations

In this section we develop an algorithm that computes the barcode of the homology of a complex of finitely generated graded modules where we assume that the complex is given by presentations. Consider the following sequence of finitely generated graded $\mathbb{k}[t]$ -modules

$$L \xrightarrow{\phi} M \xrightarrow{\psi} N$$

where $\psi \circ \phi = 0$. Suppose we are given reduced presentations of this sequence, i.e. a commutative diagram with exact rows of the form:

$$\begin{array}{ccccccccc} 0 & \longleftarrow & L & \xleftarrow{\lambda} & P_0 & \xleftarrow{p} & P_1 & \longleftarrow & 0 \\ & & \downarrow \phi & & \downarrow f_0 & & \downarrow f_1 & & \\ 0 & \longleftarrow & M & \xleftarrow{\mu} & Q_0 & \xleftarrow{q} & Q_1 & \longleftarrow & 0 \\ & & \downarrow \psi & & \downarrow g_0 & & \downarrow g_1 & & \\ 0 & \longleftarrow & N & \xleftarrow{\nu} & R_0 & \xleftarrow{r} & R_1 & \longleftarrow & 0 \end{array} \quad (9)$$

such that $g_0 \circ f_0 = 0$ and $g_1 \circ f_1 = 0$. The condition $g_0 \circ f_0 = 0$ and $g_1 \circ f_1 = 0$ means that (9) is a complex of presentations. In Example 8 one can find a concrete instance of such a complex of presentations. Our goal is to compute a presentation of $\ker \psi / \text{im } \phi$ from the presentations in (9), i.e., an exact sequence of the form:

$$0 \longleftarrow \ker \psi / \text{im } \phi \longleftarrow W_0 \xleftarrow{w} W_1$$

where $W_0 \xleftarrow{w} W_1$ is a morphism of free modules. We first construct a presentation of $\ker \psi$ through the maps

$$\begin{aligned} (g_0 \ -r) : Q_0 \times R_1 &\rightarrow R_0, \quad (g_0 \ -r)(x, y) := g_0(x) - r(y) \\ \begin{pmatrix} q \\ g_1 \end{pmatrix} : Q_1 &\rightarrow Q_0 \times R_1, \quad \begin{pmatrix} q \\ g_1 \end{pmatrix}(a) := (q(a), g_1(a)) \end{aligned}$$

where $\ker(g_0 \ -r) = \{(x, y) \in Q_0 \times R_1 \mid g_0(x) = r(y)\}$. Note that the kernel of a morphism of free modules is again a free module. Moreover, we define the map

$$\kappa : \ker(g_0 \ -r) \rightarrow \ker \psi, \quad \kappa(x, y) := \mu(x).$$

► **Proposition 4.** *The following exact sequence*

$$0 \longleftarrow \ker \psi \xleftarrow{\kappa} \ker(g_0 \ -r) \xleftarrow{\begin{pmatrix} q \\ g_1 \end{pmatrix}} Q_1$$

is a presentation of $\ker \psi$.

Next we show that the presentations of L , $\ker \psi$ and ϕ are compatible via the map

$$\begin{pmatrix} f_0 \\ 0 \end{pmatrix} : P_0 \rightarrow \ker(g_0 \ -r), \quad \begin{pmatrix} f_0 \\ 0 \end{pmatrix}(x) := (f_0(x), 0).$$

► **Proposition 5.** *The following diagram commutes:*

$$\begin{array}{ccccccccc} 0 & \longleftarrow & L & \xleftarrow{\lambda} & P_0 & \xleftarrow{p} & P_1 & \longleftarrow & 0 \\ & & \downarrow \phi & & \downarrow \begin{pmatrix} f_0 \\ 0 \end{pmatrix} & & \downarrow \begin{pmatrix} q \\ g_1 \end{pmatrix} & & \downarrow f_1 \\ 0 & \longleftarrow & \ker \psi & \xleftarrow{\kappa} & \ker(g_0 \ -r) & \xleftarrow{\quad} & Q_1 & \longleftarrow & 0 \end{array}$$

Finally, we define the presentation map from the relations to the generators

$$\begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix} : P_0 \times Q_1 \rightarrow \ker(g_0 - r), \quad \begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix} (x, y) := \begin{pmatrix} f_0 \\ 0 \end{pmatrix} (x) - \begin{pmatrix} q \\ g_1 \end{pmatrix} (y)$$

and denote by $\pi: \ker \psi \rightarrow \ker \psi / \text{im} \phi$ the quotient map.

► **Theorem 6.** *The following exact sequence*

$$0 \longleftarrow \ker \psi / \text{im} \phi \xleftarrow{\pi \circ \kappa} \ker(g_0 - r) \xleftarrow{\begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix}} P_0 \times Q_1$$

is a presentation of $\ker \psi / \text{im} \phi$.

Theorem 6 leads us to the following algorithm **PresHom** for computing the barcode of the persistence module or graded module $\ker \psi / \text{im} \phi$:

■ **Algorithm 1 PresHom.**

Input: A complex of presentations as in (9) given by the matrices $p, q, r, f_0, f_1, g_0, g_1$.

Step 1: Build the matrices $(g_0 \ -r)$ and $\begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix}$ and order the columns by degree.

Step 2: Column reduce $(g_0 \ -r)$ from left to right. The generators corresponding to the zero columns of the reduced matrix span $\ker(g_0 \ -r)$.

Step 3: Remove the rows of $\begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix}$ corresponding to non-zero columns of the reduced matrix $(g_0 \ -r)$.

Step 4: Column reduce the resulting matrix from left to right.

Output: The barcode can be read off from the final matrix $\begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix}$ in the following way: If the row r_j is a pivot row with pivot in column c_{j_i} , output the bar $[\text{deg}(r_j), \text{deg}(c_{j_i})]$ otherwise output the bar $[\text{deg}(r_j), \infty)$.

► **Theorem 7.** *The algorithm PresHom computes the barcode of $\ker \psi / \text{im} \phi$. If the matrices $p, q, r, f_0, f_1, g_0, g_1$ are of size $O(n) \times O(n)$ the algorithm takes $O(n^3)$ time.*

If we have a complex of free graded modules, then P_1, Q_1, R_1 and the morphisms p, q, r in (9) can be chosen as zero. In this case, the algorithm **PresHom** described above reduces to the persistence algorithm as described in [28].

► **Example 8.** Consider the following instance of the complex of presentations in (9)

$$\begin{array}{ccccccc} 0 & \longleftarrow & L & \longleftarrow & P_0 & \xleftarrow{\begin{pmatrix} 6 & 5 & 7 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}} & P_1 & \longleftarrow & 0 \\ & & \downarrow \phi & & \downarrow & & \downarrow & & \\ & & & & \begin{pmatrix} 2 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} & & \begin{pmatrix} 6 & 5 & 7 \\ 5 & 1 & 0 \\ 5 & 1 & 1 \\ 6 & 1 & 1 \end{pmatrix} & & \\ 0 & \longleftarrow & M & \longleftarrow & Q_0 & \xleftarrow{\begin{pmatrix} 5 & 5 & 6 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}} & Q_1 & \longleftarrow & 0 \\ & & \downarrow \psi & & \downarrow & & \downarrow & & \\ & & & & \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} & & \begin{pmatrix} 5 & 5 & 6 \\ 3 & 1 & 1 \\ 5 & 1 & 1 \end{pmatrix} & & \\ 0 & \longleftarrow & N & \longleftarrow & R_0 & \xleftarrow{\begin{pmatrix} 3 \\ 0 & (1) \end{pmatrix}} & R_1 & \longleftarrow & 0 \end{array}$$

51:8 Efficient Algorithms for Complexes of Persistence Modules with Applications

where the matrices are labeled by column and row degrees and have coefficients in \mathbb{Z}_2 . We build the matrices

$$(g_0 \ -r) = \begin{matrix} & 0 & 0 & 1 & 3 \\ 0 & (1 & 1 & 1 & 1) \end{matrix}, \quad \begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix} = \begin{matrix} & 2 & 1 & 1 & 5 & 5 & 6 \\ 0 & \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \end{matrix}$$

column reduce $(g_0 \ -r)$, delete the rows corresponding to non-zero columns from $\begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix}$ and order its columns by degree

$$(g_0 \ -r) = \begin{matrix} & 0 & 0 & 1 & 3 \\ 0 & (1 & 0 & 0 & 0) \end{matrix}, \quad \begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix} = \begin{matrix} & 1 & 1 & 2 & 5 & 5 & 6 \\ 0 & \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \end{matrix}$$

Now we column reduce $\begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix}$ and obtain the reduced matrix

$$\begin{pmatrix} f_0 & -q \\ 0 & -g_1 \end{pmatrix} = \begin{matrix} & 1 & 1 & 2 & 5 & 5 & 6 \\ 0 & \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{matrix} \quad (10)$$

Finally we read off the barcode of $\ker \psi / \text{im } \phi$ from (10) in the following way: Row 1 contributes the bar $[0, 1)$, row 2 contributes the empty bar $[1, 1)$, and row 3 contributes the bar $[3, 5)$.

4 Computing presentations of morphisms of persistence modules

The algorithm described in the previous section requires as input a complex of presentations of graded modules. In practice, when working with, for example, simplicial towers or persistent sheaves we cannot always assume that the input is given as a complex of presentations. Thus, to make use of our algorithm in various settings, we develop an efficient algorithm **PresPersMod** to compute a presentation of a morphism of persistence modules. Given a morphism of persistence modules $\phi: M \rightarrow N$ of finite type

$$\begin{array}{ccccccc} M_0 & \xrightarrow{A_0} & M_1 & \xrightarrow{A_1} & \cdots & \xrightarrow{A_{m-1}} & M_m \xrightarrow{\cong} \cdots \\ \downarrow C_0 & & \downarrow C_1 & & & & \downarrow C_m \\ N_0 & \xrightarrow{B_0} & N_1 & \xrightarrow{B_1} & \cdots & \xrightarrow{B_{m-1}} & N_m \xrightarrow{\cong} \cdots \end{array} \quad (11)$$

where we use the notation $M_i := M(i)$, $N_i := N(i)$ and the morphisms $M(i \leq i+1)$, $N(i \leq i+1)$ and $\phi(i)$ are input by the respective matrices A_i , B_i and C_i . Our goal is to compute a canonical presentation

$$\begin{array}{ccc} P_0 & \xleftarrow{p} & P_1 \\ f_0 \downarrow & & \downarrow f_1 \\ Q_0 & \xleftarrow{q} & Q_1 \end{array} \quad (12)$$

presenting a morphism of persistence modules isomorphic to (11). As explained in Section 2, it is enough to compute the matrix f_0 where each row and column additionally stores the degree of the corresponding generator $b(-)$ and its relation $d(-)$. Each generator-relation

pair in a canonical presentation corresponds to an interval summand in a persistence module isomorphic to the top or bottom row of (11). Hence, an alternative point of view is that the algorithm computes the barcodes of (11) while keeping track of how the bars map to each other (cf. (5) and (6)). The algorithm **PresPersMod** maintains a canonical presentation

$$\begin{array}{ccc} P_0^i & \xleftarrow{p^i} & P_1^i \\ f_0^i \downarrow & & \downarrow f_1^i \\ Q_0^i & \xleftarrow{q^i} & Q_1^i \end{array}$$

of a morphism of persistence modules obtained by restricting the original modules up to index i

$$\begin{array}{ccccccc} M_0 & \xrightarrow{A_0} & \dots & \xrightarrow{A_{i-1}} & M_i & \xrightarrow{\text{id}} & M_i & \xrightarrow{\text{id}} & \dots \\ \downarrow C_0 & & & & \downarrow C_i & & \downarrow C_i & & \\ N_0 & \xrightarrow{B_0} & \dots & \xrightarrow{B_{i-1}} & N_i & \xrightarrow{\text{id}} & N_i & \xrightarrow{\text{id}} & \dots \end{array} \tag{13}$$

while processing (11) from left $i = 0$ to right $i = m$. Iteratively, we build the matrix f_0^i with the birth- and death-time annotations. In a generic step, when we arrive at i from $i - 1$, the matrices A_i, B_i , and C_i are already transformed to A_i^t, B_i^t , and C_i^t respectively. We reduce these matrices further to A_i^r, B_i^r , and C_i^r which induce transformations of A_{i+1}, B_{i+1} , and C_{i+1} as we go forward in **PresPersMod** described below.

Initialization at $i = 0$. We start with the matrices A_0, B_0 and C_0 . Each column and row of C_0 corresponds to a basis element of M_0 and N_0 . Since we are at index 0, each of these basis elements has to start a new bar and they are mapped according to the matrix C_0 . Therefore, we set $f_0^0 := C_0$ with $b(c_j) = 0, d(c_j) = \infty$ and $b(r_k) = 0, d(r_k) = \infty$ for every column c_j and every row r_k of C_0 . Each bar at index 0 now corresponds to a generator in P_0^0 or Q_0^0 and their maps are determined by the matrix of generators f_0^0 . Initially, we set $A_0^t := A_0, B_0^t := B_0$ and $C_0^t := C_0$ respectively.

Moving from i to $i + 1$. Given the matrix f_0^i and the matrices A_i^t and B_i^t from the previous step. Each column c_j and row r_k of f_0^i corresponds to a bar in persistence modules isomorphic to (13). The bars with $d(c_j), d(r_k) < \infty$ already finished (died) and need no further processing. For the bars with $d(c_j), d(r_k) = \infty$ we need to determine which of them die entering index $i + 1$. Moreover, we need to determine how many new bars are born at index $i + 1$ and how they are mapped.

Each column of A_i^t and B_i^t corresponds to a column and row of f_0^i respectively and they are ordered w.r.t. their order in f_0^i . Column reduce A_i^t and B_i^t from left to right to have reduced matrices A_i^r and B_i^r respectively while performing the corresponding column and row operations on f_0^i to obtain the new matrix \bar{f}_0^i . If a_j is a zero column in A_i^r and \bar{c}_{i_j} is the corresponding column of \bar{f}_0^i , we set $d(\bar{c}_{i_j}) = i + 1$. We proceed in the same way for the columns of B_i^r and the rows of \bar{f}_0^i .

Every row of A_i^r or B_i^r that is not a pivot row could be zeroed out by row reductions. Hence, after a change of basis, every non-pivot row of A_i^r or B_i^r is a generator of the respective cokernel and thus generates a new bar born at index $i + 1$. We do not perform the row reduction on A_i^r or B_i^r but we transform A_{i+1}, B_{i+1} and C_{i+1} according to this change of bases to maintain matrices that are consistent with the bases induced by our presentation.

51:10 Efficient Algorithms for Complexes of Persistence Modules with Applications

These transformed matrices constitute A_{i+1}^t , B_{i+1}^t and C_{i+1}^t for the next iteration. Observe that every column of A_{i+1}^t and B_{i+1}^t and every column and row of C_{i+1}^t corresponds either to a non-pivot row of A_i^r or B_i^r and thus to a new born bar at index $i+1$ or to a pivot row with pivot in the j -th column and thus to the bar of the j -th column in A_i^r or B_i^r . We sort the columns of A_{i+1}^t and B_{i+1}^t and the columns and rows of C_{i+1}^t w.r.t. their order in \bar{f}_0^i where columns and rows for new generators are added at the end in arbitrary order. The matrix C_{i+1}^t , restricted to the columns corresponding to non-pivot rows of A_i^r (new generators), is added to \bar{f}_0^i to obtain $f_0^{i+1} := \bar{f}_0^i$. For each new column g_j and row h_k we set $b(g_j) = i+1$, $d(g_j) = \infty$ and $b(h_k) = i+1$, $d(h_k) = \infty$.

► **Theorem 9.** *Given a morphism as in (11), the algorithm **PresPersMod** computes a canonical presentation (12) in $O(n^3)$ time where $\dim(M_i), \dim(N_i) = O(n_i)$ and $n = \sum_{i=0}^m n_i$.*

► **Example 10.** Consider the following morphism of persistence modules over \mathbb{Z}_2

$$\begin{array}{ccccccc} M_0 & \xrightarrow{\begin{pmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}} & M_1 & \xrightarrow{\begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}} & M_2 & \xrightarrow{\cong} & \dots \\ \downarrow \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix} & & \downarrow \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix} & & \downarrow \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} & & \\ N_0 & \xrightarrow{\begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}} & N_1 & \xrightarrow{\begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}} & N_2 & \xrightarrow{\cong} & \dots \end{array}$$

We initialize $f_0^0 = C_0$ and set the birth- and death-time of every generator to 0 and ∞ :

$$f_0^0 = \begin{array}{ccc} \begin{matrix} c_1 & c_2 & c_3 \\ r_1 & \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ r_3 & 1 & 0 & 0 \end{pmatrix} \end{matrix} & , & \begin{matrix} b & \begin{matrix} c_1 & c_2 & c_3 \\ \begin{pmatrix} 0 & 0 & 0 \\ \infty & \infty & \infty \end{pmatrix} \end{matrix} & \begin{matrix} r_1 & r_2 & r_3 \\ \begin{pmatrix} 0 & 0 & 0 \\ \infty & \infty & \infty \end{pmatrix} \end{matrix} \end{matrix}$$

We column reduce A_0 and B_0 while performing the corresponding operations on f_0^0 :

$$A_0^r = \begin{pmatrix} c_1 & c_2 & c_3 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, \quad B_0^r = \begin{pmatrix} r_1 & r_2 & r_3 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \bar{f}_0^0 = \begin{array}{ccc} \begin{matrix} c_1 & c_2 & c_3 \\ r_1 & \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ r_3 & 1 & 1 & 1 \end{pmatrix} \end{matrix} \end{array}$$

We perform the basis transformation on A_1 , B_1 and C_1 :

$$A_1^t = \begin{pmatrix} c_1 & c_2 & c_4 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad B_1^t = \begin{pmatrix} r_1 & r_4 & r_5 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \quad C_1^t = \begin{array}{ccc} \begin{matrix} c_1 & c_2 & c_4 \\ r_1 & \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ r_5 & 0 & 0 & 0 \end{pmatrix} \end{matrix} \end{array}$$

We update \bar{f}_0^0 in the following way: The column c_4 and the rows r_4 and r_5 of C_1^t correspond to non-pivot rows of A_0^r or B_0^r . We add them as new columns and rows of f_0^1 born at index 1. The columns c_3 of A_0^r and r_2, r_3 of B_0^r are zero-columns, thus we set $d(c_3), d(r_2), d(r_3) = 1$:

$$f_0^1 = \begin{array}{ccc} \begin{matrix} c_1 & c_2 & c_3 & c_4 \\ r_1 & \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ r_3 & 1 & 1 & 0 \\ r_4 & 0 & 0 & 1 \\ r_5 & 0 & 0 & 0 \end{pmatrix} \end{matrix} & , & \begin{matrix} b & \begin{matrix} c_1 & c_2 & c_3 & c_4 \\ \begin{pmatrix} 0 & 0 & 0 & 1 \\ \infty & \infty & 1 & \infty \end{pmatrix} \end{matrix} & \begin{matrix} r_1 & r_2 & r_3 & r_4 & r_5 \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 \\ \infty & 1 & 1 & \infty & \infty \end{pmatrix} \end{matrix} \end{matrix}$$

Since A_1^t and B_1^t are already column reduced we can directly transform C_2 :

$$C_2^t = \begin{array}{ccc} \begin{matrix} c_4 & c_5 & c_6 \\ r_1 & \begin{pmatrix} 1 & 1 & 1 \\ r_4 & 1 & 1 \\ r_5 & 0 & 1 \end{pmatrix} \end{matrix} \end{array}$$

and update $\bar{f}_0^1 = f_0^1$ by adding columns c_5, c_6 of C_2^t corresponding to non-pivot rows of A_1^t . Since c_1 and c_2 are zero-columns of A_1^t , we set $d(c_1) = d(c_2) = 2$ and obtain the final result:

$$f_0^2 = \begin{matrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{matrix} \begin{pmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad b \begin{pmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & r_1 & r_2 & r_3 & r_4 & r_5 \\ 0 & 0 & 0 & 1 & 2 & 2 & 0 & 0 & 0 & 1 & 1 \\ 2 & 2 & 1 & \infty & \infty & \infty & \infty & 1 & 1 & \infty & \infty \end{pmatrix}$$

5 Complexes of presentations

The algorithm introduced in the previous section allows us to compute a canonical presentation of a complex of persistence modules. To obtain a valid input for our homology algorithm the resulting presentations need to form a complex, i.e., a sequence of presentations connected by morphisms where composition of consecutive morphisms is zero. The problem is that an arbitrary presentation of a complex of graded modules is not necessarily a complex of presentations. Consider the following complex of persistence modules over \mathbb{Z}_2

$$\mathbb{I}_{[1,2)} \xrightarrow{\phi} \mathbb{I}_{[0,2)} \xrightarrow{\psi} \mathbb{I}_{[0,1)}$$

where ϕ and ψ are the identity map on the overlap of the intervals and 0 elsewhere. Note that the composition is zero because the first and the third interval do not overlap. The following canonical presentation of the corresponding complex of $\mathbb{k}[t]$ -modules

$$\begin{array}{ccccccc} 0 & \longleftarrow & \mathbb{I}_{[1,2)} & \longleftarrow & \mathbb{I}_{[1,\infty)} & \xleftarrow{t} & \mathbb{I}_{[2,\infty)} & \longleftarrow & 0 \\ & & \downarrow \phi & & \downarrow t & & \downarrow 1 & & \\ 0 & \longleftarrow & \mathbb{I}_{[0,2)} & \longleftarrow & \mathbb{I}_{[0,\infty)} & \xleftarrow{t^2} & \mathbb{I}_{[2,\infty)} & \longleftarrow & 0 \\ & & \downarrow \psi & & \downarrow 1 & & \downarrow t & & \\ 0 & \longleftarrow & \mathbb{I}_{[0,1)} & \longleftarrow & \mathbb{I}_{[0,\infty)} & \xleftarrow{t} & \mathbb{I}_{[1,\infty)} & \longleftarrow & 0 \end{array} \tag{14}$$

is not a complex of presentations because $1 \circ t \neq 0$ and $t \circ 1 \neq 0$. Using Proposition 11, we show that we can always modify a canonical presentation of a complex of graded modules to a complex of presentations.

► **Proposition 11.** Consider a commutative diagram of canonical presentations of the form:

$$\begin{array}{ccccccc} 0 & \longleftarrow & \mathbb{I}_{[a,b)} & \longleftarrow & \mathbb{I}_{[a,\infty)} & \xleftarrow{t^{b-a}} & \mathbb{I}_{[b,\infty)} & \longleftarrow & 0 \\ & & \downarrow \iota & & \downarrow \iota_0 & & \downarrow \iota_1 & & \\ 0 & \longleftarrow & M & \xleftarrow{\mu} & P_0 & \xleftarrow{p} & P_1 & \longleftarrow & 0 \\ & & \downarrow \phi & & \downarrow f_0 & & \downarrow f_1 & & \\ 0 & \longleftarrow & N & \xleftarrow{\nu} & Q_0 & \xleftarrow{q} & Q_1 & \longleftarrow & 0 \\ & & \downarrow \pi & & \downarrow \pi_0 & & \downarrow \pi_1 & & \\ 0 & \longleftarrow & \mathbb{I}_{[c,d)} & \longleftarrow & \mathbb{I}_{[c,\infty)} & \xleftarrow{t^{d-c}} & \mathbb{I}_{[d,\infty)} & \longleftarrow & 0 \end{array} \tag{15}$$

If not $(c < d \leq a < b)$, then $(\pi \circ \phi \circ \iota = 0 \iff \pi_0 \circ f_0 \circ \iota_0 = 0 \iff \pi_1 \circ f_1 \circ \iota_1 = 0)$.

It implies that given canonical presentations as in (9), the only case where the entry of the matrix $g_0 \circ f_0$, corresponding to the bars $\mathbb{I}_{[a,b)}$ and $\mathbb{I}_{[c,d)}$, can be non-zero is when $c < d \leq a < b$. In this case, we can add a new bar $\mathbb{I}_{[e,e)}$ of length zero with $d \leq e \leq a$ to the

middle presentation in (9), and map $\mathbb{I}_{[a,\infty)} \rightarrow \mathbb{I}_{[e,\infty)}$ and $\mathbb{I}_{[e,\infty)} \rightarrow \mathbb{I}_{[c,\infty)}$ in such a way that the resulting map $\mathbb{I}_{[a,b)} \rightarrow \mathbb{I}_{[c,d)}$ is zero. Given the morphisms of presentations in (14), we add $\mathbb{I}_{[1,\infty)} \xleftarrow{1} \mathbb{I}_{[1,\infty)}$ to the middle term to obtain the following complex of presentations:

$$\begin{array}{ccccccc}
 0 & \longleftarrow & \mathbb{I}_{[1,2)} & \longleftarrow & \mathbb{I}_{[1,\infty)} & \xleftarrow{t} & \mathbb{I}_{[2,\infty)} & \longleftarrow & 0 \\
 & & \downarrow \phi & & \downarrow \begin{pmatrix} t \\ 1 \end{pmatrix} & & \begin{pmatrix} t^2 & 0 \\ 0 & 1 \end{pmatrix} & & \downarrow \begin{pmatrix} 1 \\ t \end{pmatrix} \\
 0 & \longleftarrow & \mathbb{I}_{[0,2)} & \longleftarrow & \mathbb{I}_{[0,\infty)} \oplus \mathbb{I}_{[1,\infty)} & \xleftarrow{\quad} & \mathbb{I}_{[2,\infty)} \oplus \mathbb{I}_{[1,\infty)} & \longleftarrow & 0 \\
 & & \downarrow \psi & & \downarrow \begin{pmatrix} 1 & t \end{pmatrix} & & \downarrow \begin{pmatrix} t & 1 \end{pmatrix} & & \\
 0 & \longleftarrow & \mathbb{I}_{[0,1)} & \longleftarrow & \mathbb{I}_{[0,\infty)} & \xleftarrow{t} & \mathbb{I}_{[1,\infty)} & \longleftarrow & 0
 \end{array}$$

► **Theorem 12.** *Let $L \xrightarrow{\phi} M \xrightarrow{\psi} N$ be a complex of finitely generated graded $\mathbb{k}[t]$ -modules, then there exists a complex of presentations presenting $L \xrightarrow{\phi} M \xrightarrow{\psi} N$.*

► **Corollary 13.** *Given a complex of graded $\mathbb{k}[t]$ -modules $L \xrightarrow{\phi} M \xrightarrow{\psi} N$ and a canonical presentation as in (9) of size $n := \max\{\text{rank}(P_0), \text{rank}(Q_0), \text{rank}(R_0)\}$, we can modify (9) such that the modified presentation is a reduced complex of presentations of size $O(n)$.*

The modification mentioned in Corollary 13 can be implemented in time $O(n^3)$ in the following way:

■ **Algorithm 2** An $O(n^3)$ implementation of the modification mentioned in Corollary 13.

Input: The matrices f_0 and g_0 with birth/death annotations representing canonical presentations of ϕ and ψ .

Step 1: Compute the matrix product $g_0 \circ f_0$.

Step 2: For every non-zero column c_j of $g_0 \circ f_0$, add a row $(0, \dots, 0, \frac{1}{j}, 0, \dots, 0)$ with birth and death index $b(c_j)$ to the end of f_0 and the column $-c_j$ with birth and death index $b(c_j)$ to the end of g_0 .

Output: The modified matrices f_0 and g_0 .

6 Applications

6.1 Persistent homology of simplicial towers

We consider a simplicial tower:

$$\vec{K} : K_0 \xrightarrow{f_0} K_1 \xrightarrow{f_1} \dots \xrightarrow{f_{m-1}} K_m \quad (16)$$

where K_i is a finite simplicial complex and f_i an arbitrary simplicial map. Our goal is to compute the barcode of the homology persistence module

$$H_k(\vec{K}) : H_k(K_0) \xrightarrow{H_k(f_0)} H_k(K_1) \xrightarrow{H_k(f_1)} \dots \xrightarrow{H_k(f_{m-1})} H_k(K_m) \xrightarrow{\cong} \dots \quad (17)$$

where we artificially extend finite persistence modules by isomorphisms to infinite persistence modules of finite type to fit them into the algebraic framework discussed in Section 2. From an input tower (16), we can construct the following complex of persistence modules for computing its persistent homology

$$C_{k+1}(\vec{K}) \xrightarrow{\partial_{k+1}} C_k(\vec{K}) \xrightarrow{\partial_k} C_{k-1}(\vec{K}) \tag{18}$$

where $C_k(\vec{K})$ is the persistence module of simplicial k -chains in \vec{K} defined by

$$C_k(\vec{K}) : C_k(K_0) \xrightarrow{C_k(f_0)} C_k(K_1) \xrightarrow{C_k(f_1)} \dots \xrightarrow{C_k(f_{m-1})} C_k(K_m) \xrightarrow{\cong} \dots \tag{19}$$

Since simplices can be collapsed in a tower, the linear maps $C_k(f_i)$ are not necessarily injective. Thus, in general $C_k(\vec{K})$ is not a free module, but we can use the pipeline developed in Sections 4, 5 and 3 to compute the barcode of (17).

Tower Algorithm. In practice a tower is usually represented as a sequence of elementary inclusions (adding a single simplex) and elementary collapses (collapsing exactly two vertices). Every tower can be represented in this way [11], so w.l.o.g. we assume that (16) is a sequence of elementary inclusions and collapses. Every collapse can be thought of as making some simplices (chains) collapse trivially to 0, and some other pairs of simplices merging together. The number of such trivial collapses and mergings cannot be more than the number of elementary inclusions. So, we define the size n of the tower as the number of elementary inclusions in \vec{K} . The special kind of persistence modules (19), arising from simplicial towers, significantly simplify the computation of a presentation of (18). The reason for this simplification is that in the matrices $C_k(f_i)$ in (19) each column is either zero or contains a single 1 which implies that each column is reduced by a single other column and there is no need to explicitly construct these matrices. In fact, a canonical presentation of $C_k(\vec{K})$ can be computed in $O(n)$ time directly by observing elementary inclusions (creating generators), trivial collapses (pairing simplices to itself), and mergings (pairing the relation to the simplex in the merge appearing later in the filtration). Corresponding to every merge, we have a basis change which needs to be reflected by a sum of two columns in the boundary matrices that connect two modules. This will incur quadratic complexity. By Proposition 11 and the fact that the boundary of a simplicial chain can not die before the chain itself, this computation of a canonical presentation of the complex in (18) yields a complex of presentations.

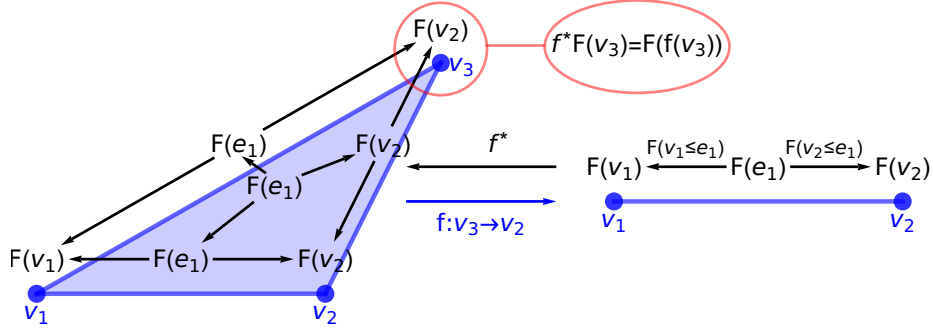
► **Theorem 14.** *Given a simplicial tower (16) of size n , the Tower Algorithm outlined above computes the persistent homology, i.e., the barcode of (17) in $O(n^3)$ time.*

6.2 Persistent cosheaf homology over simplicial towers

In this section and the next, we consider cosheaves and sheaves over finite simplicial complexes. Briefly, as depicted in Figure 1, a (co)sheaf is an assignment of vector spaces on simplices of a simplicial complex and linear maps among them, which satisfy certain conditions of commutativity. Similar to the homology of a simplicial complex, the homology of a cosheaf over a simplicial complex can be computed from a chain complex where the boundary operators are determined by how the vector spaces over k -simplices map to the vector spaces over their $(k - 1)$ -dimensional faces.

The persistent homology of a tower can be viewed as a special case of persistent cosheaf homology over a tower where the cosheaf is the constant cosheaf. We consider a simplicial tower as in the lower row of (20) and a cosheaf $F : K_m^{\text{op}} \rightarrow \mathbf{vec}$ over K_m .

$$\begin{array}{ccccccc}
 F^0 & \xleftarrow{f_0^*} & F^1 & \xleftarrow{f_1^*} & \dots & \xleftarrow{f_{m-1}^*} & F^m \\
 \vec{K} : & K_0 & \xrightarrow{f_0} & K_1 & \xrightarrow{f_1} & \dots & \xrightarrow{f_{m-1}} & K_m
 \end{array} \tag{20}$$



■ **Figure 1** The pullback f^*F of a cosheaf F along an elementary collapse $f : v_3 \rightarrow v_2$.

As shown in Figure 1, we use the inverse image functors f_i^* to iteratively pull back $F = F^m$ to cosheaves on all the K_i by defining $F^i = f_i^* F^{i+1}$. We now use the induced maps $H_k(f_i) : H_k(K_i, F^i) \rightarrow H_k(K_{i+1}, F^{i+1})$ to construct the cosheaf homology persistence module

$$H_k(\vec{K}, F) : H_k(K_0, F^0) \xrightarrow{H_k(f_0)} H_k(K_1, F^1) \xrightarrow{H_k(f_1)} \dots \xrightarrow{H_k(f_{m-1})} H_k(K_m, F^m) \xrightarrow{\cong} \dots \quad (21)$$

The persistent cosheaf homology is the homology of the complex of persistence modules

$$C_{k+1}(\vec{K}, F) \xrightarrow{\partial_{k+1}} C_k(\vec{K}, F) \xrightarrow{\partial_k} C_{k-1}(\vec{K}, F)$$

where $C_k(\vec{K}, F)$ is the persistence module of finite type defined by

$$C_k(\vec{K}, F) : C_k(K_0, F^0) \xrightarrow{C_k(f_0)} C_k(K_1, F^1) \xrightarrow{C_k(f_1)} \dots \xrightarrow{C_k(f_{m-1})} C_k(K_m, F^m) \xrightarrow{\cong} \dots$$

Cosheaf Algorithm. As in the case of simplicial towers, the persistence modules $C_k(\vec{K}, F)$ are not necessarily free. We can again assume that the tower is given by a sequence of elementary inclusions and collapses and define the size n of the instance (\vec{K}, F) as $n := \# \text{ elementary inclusions} \times \max_{\sigma \in K_m} \dim(F(\sigma))$. We can now use our pipeline developed in Sections 4, 5 and 3 to compute the barcode of (21) where the same simplifications as in the tower case (Section 6.1) apply.

► **Theorem 15.** *Given an instance (\vec{K}, F) of size n as in (20), the Cosheaf Algorithm above computes the persistent cosheaf homology, i.e., the barcode of (21) in $O(n^3)$ time.*

6.3 Cohomology of persistent sheaves over simplicial complexes

In this section we compute the cohomology of a persistent sheaf over a simplicial complex. We consider a persistent sheaf $\vec{F} : \mathbb{N}_0 \rightarrow \mathbf{Shv}(K, \mathbf{vec})$ of finite type on a finite simplicial complex K , i.e. a diagram of sheaves and sheaf morphisms (see Figure 2) over K

$$\vec{F} : F_0 \xrightarrow{\phi_0} F_1 \xrightarrow{\phi_1} \dots \xrightarrow{\phi_{m-1}} F_m \xrightarrow{\cong} \dots \quad (22)$$

where ϕ_i is an isomorphism for all $i \geq m$. As depicted in Figure 3, a persistent sheaf can also be viewed as a sheaf of persistence modules. The figure shows parts of the sheaf in (23) of interval persistence modules on a simplicial complex containing a triangle and its faces where the maps are the identity on the overlap of intervals and zero elsewhere:

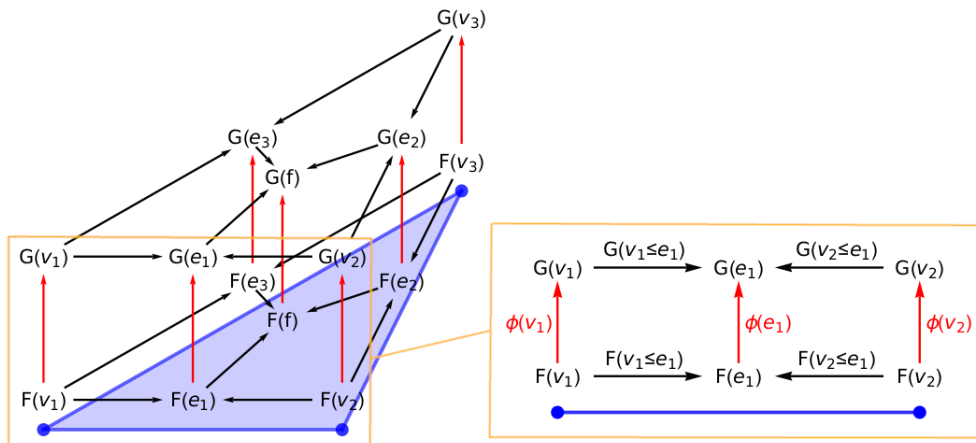


Figure 2 A sheaf morphism $\phi: F \rightarrow G$ over a triangle and its faces.

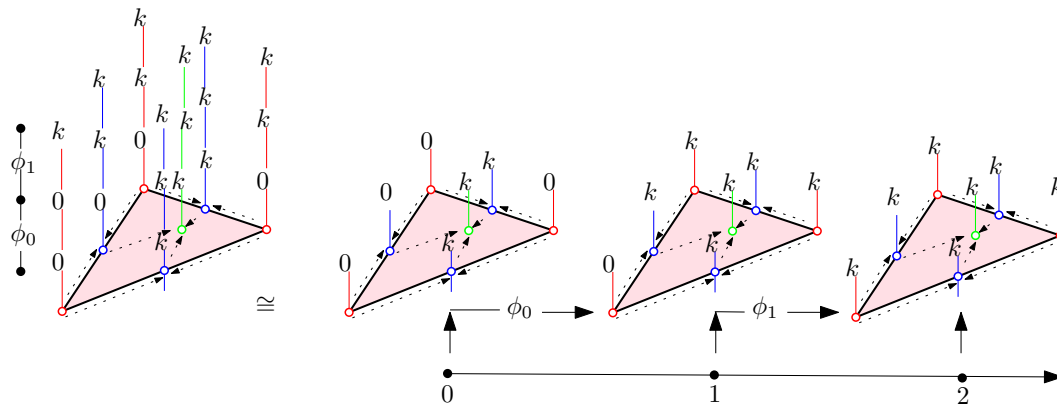


Figure 3 Two equivalent viewpoints of a persistent sheaf over a triangle and its faces.

$$\begin{array}{ccccc}
 & & \mathbb{I}_{[1,7]} & & \\
 & \swarrow & & \searrow & \\
 \mathbb{I}_{[1,6]} & \longrightarrow & \mathbb{I}_{[0,3]} & \longleftarrow & \mathbb{I}_{[0,5]} \\
 \swarrow & & \uparrow & & \swarrow \\
 \mathbb{I}_{[2,6]} & \longrightarrow & \mathbb{I}_{[0,5]} & \longleftarrow & \mathbb{I}_{[1,5]}
 \end{array} \tag{23}$$

Our goal is to compute the persistent sheaf cohomology of \vec{F} , i.e. the barcode of the following persistence module:

$$H^k(X, \vec{F}) : H^k(K, F_0) \xrightarrow{H^k(K, \phi_0)} H^k(K, F_1) \xrightarrow{H^k(K, \phi_1)} \dots \xrightarrow{H^k(K, \phi_{m-1})} H^k(K, F_m) \xrightarrow{\cong} \dots \tag{24}$$

Persistent Sheaf Algorithm. We first assume that the input for the computation is given by a collection of matrices representing the internal maps $F_i(\sigma \leq \tau)$ of the sheaves for all $0 \leq i \leq m$ and $\sigma < \tau \in K$ and a collection of matrices representing the sheaf morphisms $\phi_i(\sigma)$ for all $0 \leq i < m$ and $\sigma \in K$. We define the size of the input as $n := \sum_{i=0}^m n_i$ where $n_i := \sum_{\sigma \in K} \dim(F_i(\sigma))$.

From the input we can construct the following complex of persistence modules

$$C^{k-1}(K, \vec{F}) \xrightarrow{\delta^{k-1}} C^k(K, \vec{F}) \xrightarrow{\delta^k} C^{k+1}(K, \vec{F}) \quad (25)$$

where $C^k(K, \vec{F})$ is a persistence module of finite type defined by

$$C^k(K, \vec{F}) : C^k(K, F_0) \xrightarrow{C^k(K, \phi_0)} C^k(K, F_1) \xrightarrow{C^k(K, \phi_1)} \dots \xrightarrow{C^k(K, \phi_{m-1})} C^k(K, F_m) \xrightarrow{\cong} \dots$$

Using the pipeline developed in Sections 4, 5 and 3, we can compute a presentation of (25), transform it into a complex of presentations and compute the barcode of the cohomology. As an example, the complex of presentations in Example 8 in Section 3 is a presentation of (25) for $k = 1$ and the persistent sheaf in (23).

► **Theorem 16.** *Given a persistent sheaf (22) of size n , the Persistent Sheaf Algorithm above computes the persistent sheaf cohomology, i.e., the barcode of (24), in $O(n^3)$ time.*

In practice, it is unlikely that one is just handed a persistent sheaf in the form described above. In an application the persistent sheaf has to be constructed in the first place. To construct a persistent sheaf $\vec{F} : \mathbb{N}_0 \rightarrow \mathbf{Shv}(K, \mathbf{vec})$ or equivalently a sheaf of persistence modules $\vec{F} : K \rightarrow \mathbf{pMod}$, one has to construct a persistence module $\vec{F}(\sigma) : \mathbb{N}_0 \rightarrow \mathbf{vec}$ over every simplex $\sigma \in K$ and connect them by a morphism of persistence modules $\vec{F}(\sigma \leq \tau)$ if $\sigma <_1 \tau$. The idea of sheaves is to organize local information over a space. We use this idea to compute presentations $0 \leftarrow \vec{F}(\sigma) \leftarrow P_0^\sigma \xleftarrow{P} P_1^\sigma$ of the modules over each simplex and the morphisms $\vec{F}(\sigma \leq \tau)$ locally in time $O(\max\{\sum_{i=0}^m \dim(F_i(\sigma)), \sum_{i=0}^m \dim(F_i(\tau))\}^3)$. This can be computed in a distributed manner while constructing the persistent sheaf. From these local presentations we can assemble a presentation of (25). After this construction / preprocessing step we can define the size of the presentation as $\bar{n} := \sum_{\sigma \in K} \text{rank}(P_0^\sigma)$. The size of \bar{n} depends on the structure of the modules $\vec{F}(\sigma)$. If all their summands restricted to $[0, m]$ are intervals of length 0, then $\bar{n} = n$, but, if all their summands are intervals of length m , then $\bar{n} = \frac{n}{m}$. The remaining steps in our pipeline take $O(\bar{n}^3)$ time. Therefore, if one can take advantage of distributed computation of the input and local presentations our approach can be significantly faster than $O(n^3)$.

6.4 Cohomology of persistent sheaves over posets

So far our algorithms allow us to compute the persistent sheaf cohomology barcode of persistent sheaves over simplicial complexes. Discrete sheaves and their persistent version can be defined more generally on finite posets. The problem is that for arbitrary finite posets there is no direct analog to the explicit (co)chain complex (of polynomial size) computing the sheaf cohomology over a simplicial complex. For arbitrary posets one could use the general definition of sheaf cohomology via derived functors (see [5] for the non-persistent case). We now show that we can reduce the computation of persistent sheaf cohomology over a finite poset to a computation over a simplicial complex. Let X be a finite poset and $\vec{F} : \mathbb{N}_0 \rightarrow \mathbf{Shv}(X, \mathbf{vec})$ a persistent sheaf. The order complex $\mathcal{K}(X)$ is the simplicial complex defined by all totally ordered subsets of X , i.e. $\mathcal{K}(X) := \{\{x_0, \dots, x_k\} | x_0 < \dots < x_k \in X\}$. It comes with a surjective projection morphism $f : \mathcal{K}(X) \rightarrow X$ defined by $f(x_0 < \dots < x_k) := x_k$. We can now pull back \vec{F} along this projection f to a persistent sheaf on $\mathcal{K}(X)$. Given $x_\bullet = (x_0 < \dots < x_k) \leq (y_0 < \dots < y_l) = y_\bullet \in \mathcal{K}(X)$ we define this pullback by $f^* \vec{F}(x_\bullet) := \vec{F}(x_k)$ and $f^* \vec{F}(x_\bullet \leq y_\bullet) := \vec{F}(x_k \leq y_l)$. The following theorem shows that the persistent sheaf cohomology of \vec{F} on the poset X is isomorphic to the persistent sheaf cohomology of $f^* \vec{F}$ on the simplicial complex $\mathcal{K}(X)$.

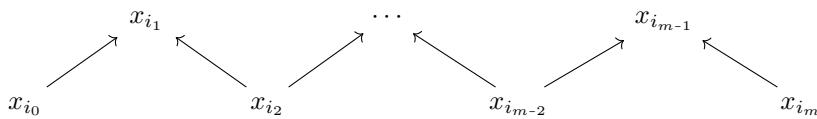
► **Theorem 17.** *If X is a finite poset, \vec{F} a persistent sheaf on X and $f: \mathcal{K}(X) \rightarrow X$ the projection from the order complex, then $H^k(X, \vec{F}) \cong H^k(\mathcal{K}(X), f^*\vec{F})$.*

We are now able to compute the persistent sheaf cohomology over any finite poset by pulling back to the order complex and using our pipeline for simplicial complexes. Unfortunately this approach is only practical for small instances since the size of the order complex is exponential in the size of the poset. This raises the question if it is possible to compute the persistent sheaf cohomology over a general poset in polynomial time. In the following we give an example of a class of posets, which can have exponentially large order complexes, where we can compute the persistent cohomology in polynomial time.

Let X be a zigzag-poset, i.e. a poset with the following Hasse diagram

$$x_0 \longleftrightarrow x_1 \longleftrightarrow \cdots \longleftrightarrow x_{n-1} \longleftrightarrow x_n \tag{26}$$

where each arrow can point to the left or to the right. Let \vec{F} be a persistent sheaf on X . We now construct a poset X' in the following way: Suppose the points in X are ordered from left to right as in (26). Let x_{i_0} be the minimal element in X with the smallest index $i_0 \in \{0, \dots, n\}$. Let x_{i_1} be the maximal element of X with minimal index $i_1 > i_0$. Let x_{i_2} be the minimal element of X with minimal index $i_2 > i_1$. We proceed in this way until we reach the minimal element x_{i_m} with maximal index i_m and define $X' := \{x_{i_0}, \dots, x_{i_m}\}$. The constructed zigzag-poset X' is an alternating zigzag poset of the form:



and is a subposet of X , i.e. there is an order-preserving inclusion $\iota: X' \hookrightarrow X$. We can pull back the persistent sheaf \vec{F} along this inclusion to obtain a persistent sheaf $\iota^*\vec{F}$ on X' . Since X' is the poset of a one-dimensional simplicial complex, our algorithms are applicable to $\iota^*\vec{F}$. Now, we can show that the cohomology of \vec{F} on X agrees with the cohomology of $\iota^*\vec{F}$ on X' .

► **Theorem 18.** *Let X be a zigzag-poset and $\iota: X' \hookrightarrow X$ the inclusion of the alternating subposet. If \vec{F} is a persistent sheaf on X , then $H^k(X, \vec{F}) \cong H^k(X', \iota^*\vec{F})$.*

References

- 1 Gorô Azumaya. Corrections and supplementaries to my paper concerning Krull-Remak-Schmidt’s theorem. *Nagoya Mathematical Journal*, 1:117–124, 1950.
- 2 Federico Barbero, Cristian Bodnar, Haitz Sáez de Ocáriz Borde, Michael Bronstein, Petar Veličković, and Pietro Liò. Sheaf neural networks with connection laplacians. In *Proceedings of Topological, Algebraic, and Geometric Learning Workshops 2022*, volume 196 of *Proceedings of Machine Learning Research*, pages 28–36. PMLR, 25 February–22 July 2022.
- 3 Nicolas Berkouk, Grégory Ginot, and Steve Oudot. Level-sets persistence and sheaf theory. *CoRR*, abs/1907.09759, 2019. [arXiv:1907.09759](https://arxiv.org/abs/1907.09759).
- 4 Cristian Bodnar, Francesco Di Giovanni, Benjamin Chamberlain, Pietro Lió, and Michael Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gns. In *Advances in Neural Information Processing Systems*, volume 35, pages 18527–18541. Curran Associates, Inc., 2022.
- 5 Adam Brown and Ondrej Draganov. Discrete microlocal morse theory, 2022. [arXiv:2209.14993](https://arxiv.org/abs/2209.14993).

- 6 Adam Brown and Bei Wang. Sheaf-theoretic stratification learning from geometric and topological perspectives. *Discrete and Computational Geometry*, 65, 2021. doi:10.1007/s00454-020-00206-y.
- 7 René Corbet and Michael Kerber. The representation theorem of persistence revisited and generalized. *Journal of Applied and Computational Topology*, 2(1-2):1–31, 2018. doi:10.1007/s41468-018-0015-3.
- 8 Justin Curry. *Sheaves, Cosheaves and Applications*. PhD thesis, University of Pennsylvania, 2014. arXiv:1303.3255.
- 9 Justin Curry, Robert Ghrist, and Vidit Nanda. Discrete morse theory for computing cellular sheaf cohomology. *Foundations of Computational Mathematics*, 16, 2013. doi:10.1007/s10208-015-9266-8.
- 10 Justin Curry, Washington Mio, Tom Needham, Osman Berat Okutan, and Florian Russold. Convergence of lera y cosheaves for decorated mapper graphs, 2023. arXiv:2303.00130.
- 11 Tamal K. Dey, Fengtao Fan, and Yusu Wang. Computing topological persistence for simplicial maps. In *Proc. 13th Annu. Sympos. Comput. Geom. (SoCG)*, pages 345:345–345:354, 2014.
- 12 Tamal K. Dey, Florian Russold, and Shreyas N. Samaga. TDA-Jyamiti/Algos-cplx-pers-modules. Software, NSF 2301360, swHId: swH:1:dir:6c13c2c3aeb94cc68377d695005250d1ab892cb7, (visited on 21/05/2024). URL: <https://github.com/TDA-Jyamiti/Algos-cplx-pers-modules/>.
- 13 Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. *Discrete & Computational Geometry*, 28(4):511–533, November 2002.
- 14 Peter Gabriel. Unzerlegbare Darstellungen I. *Manuscripta Mathematica*, 6(1):71–103, 1972. doi:10.1007/BF01298413.
- 15 Robert Ghrist. *Elementary Applied Topology*. CreateSpace Independent Publishing Platform, 2014.
- 16 Robert Ghrist and Yasuaki Hiraoka. Applications of sheaf cohomology and exact sequences on network codings, 2011. URL: <https://www2.math.upenn.edu/~ghrist/preprints/networkcodingshort.pdf>.
- 17 Jakob Hansen and Thomas Gebhart. Sheaf neural networks, 2020. arXiv:2012.06333.
- 18 Jakob Hansen and Robert Ghrist. Opinion dynamics on discourse sheaves. *SIAM Journal on Applied Mathematics*, 81:2033–2060, 2021. doi:10.1137/20M1341088.
- 19 Masaki Kashiwara and Pierre Schapira. Persistent homology and microlocal sheaf theory. *Journal of Applied and Computational Topology*, 2, 2018. doi:10.1007/s41468-018-0019-z.
- 20 Michael Kerber and Hannah Schreiber. Barcodes of towers and a streaming algorithm for persistent homology. In *33rd International Symposium on Computational Geometry, SoCG 2017*, volume 77 of *LIPICs*, pages 57:1–57:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017.
- 21 Michael Robinson. Understanding networks and their behaviors using sheaf theory. *2013 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2013 - Proceedings*, 2013. doi:10.1109/GlobalSIP.2013.6737040.
- 22 Michael Robinson. *Topological Signal Processing*. Mathematical Engineering. Springer Berlin Heidelberg, 2014.
- 23 Florian Russold. Persistent sheaf cohomology, 2022. arXiv:2204.13446.
- 24 Álvaro Torras-Casas. Distributing persistent homology via spectral sequences. *Discrete & Computational Geometry*, pages 1–40, 2023.
- 25 Xiaoqi Wei, Jiahui Chen, and Guo-Wei Wei. Persistent topological Laplacian analysis of SARS-CoV-2 variants. *J Comput Biophys Chem.*, January 2023.
- 26 Xiaoqi Wei and Guo-Wei Wei. Persistent sheaf Laplacians, 2022. arXiv:2112.10906.
- 27 Hee Rhang Yoon and Robert Ghrist. Persistence by parts: Multiscale feature detection via distributed persistent homology, 2020. arXiv:2001.01623.
- 28 Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete and Computational Geometry*, 33:249–274, 2005. doi:10.1007/s00454-004-1146-y.