Computational Topology for Data Analysis: Notes from Book by

Tamal Krishna Dey
Department of Computer Science
Purdue University
West Lafayette, Indiana, USA 46907

Yusu Wang
Halıcıoğlu Data Science Institute
University of California, San Diego
La Jolla, California, USA 92093

Topic 6: Towers

We have considered filtrations so far for defining persistence and stability of their diagrams. In a filtration, the connecting maps between consecutive spaces or complexes are inclusions. Assuming a discrete subset of reals, $A : a_0 \leq a_1 \leq \cdots \leq a_n$, as index set, we write a filtration as:

$$\mathcal{F} : X_{a_0} \hookrightarrow X_{a_1} \hookrightarrow \cdots \hookrightarrow X_{a_n}$$

A more generalized scenario occurs when the inclusions are replaced with continuous maps for space filtrations and simplicial maps for simplicial filtrations: $x_{ij} : X_{a_i} \to X_{a_j}$. In that case, we call the sequence a space and a simplicial *tower* respectively:

$$\mathcal{X} : X_{a_0} \xrightarrow{x_{01}} X_{a_1} \xrightarrow{x_{12}} \cdots \xrightarrow{x_{(n-1)n}} X_{a_n} \tag{6.1}$$

Considering the homology group of each space (complex resp.) in the sequence, we obtain a sequence of vector spaces connected with linear maps, which we have seen before. Specifically, we obtain the following *tower* of vector spaces:

$$\mathsf{H}_p\mathcal{X} : \mathsf{H}_p(X_{a_0}) \xrightarrow{x_{01*}} \mathsf{H}_p(X_{a_1}) \xrightarrow{x_{12*}} \cdots \xrightarrow{x_{(n-1)n*}} \mathsf{H}_p(X_{a_n})$$

In the above sequence each linear map x_{ij*} is the homomorphism induced by the map x_{ij}. We have already seen that persistent homology of such a sequence of vector spaces and linear maps are well defined. However, since the linear maps here are not induced by inclusions, the original persistence algorithm as described in the previous chapter does not work. In section 6.2.1, we describe a new algorithm to compute the persistence diagram of simplicial towers.

But, before presenting the algorithms, we generalize the notion of stability for towers. We have seen such a notion for persistence modules arising out of filtrations. Here, we adapt it to a tower.

6.1 Stability of towers

Just like the previous chapter, we define the stability with respect to the perturbation of the towers themselves forgetting the functions who generate them. This requires a definition of a distance between towers at simplicial (space) levels and homology levels.

It turns out that it is convenient and sometimes appropriate if the objects (spaces, simplicial complexes, or vector spaces) in a tower are indexed over the positive real axis instead of a discrete subset of it. This, in turn, requires to spell out the connecting map between every pair of objects.

Definition 1 (Tower)**.** A *tower* indexed in an ordered set $A \subseteq \mathbb{R}$ is any collection $\mathsf{T} = \{T_a\}_{a \in A}$ of objects T_a, $a \in A$, together with maps $t_{a,a'} : T_a \to T_{a'}$ so that $t_{a,a} = \mathrm{i}d$ and $t_{a',a''} \circ t_{a,a'} = t_{a,a''}$ for all $a \leq a' \leq a''$. Sometimes we write $\mathsf{T} = \{T_a \xrightarrow{t_{a,a'}} T_{a'}\}_{a \leq a'}$ to denote the collection with the maps. We say that the tower T has resolution r if $a \geq r$ for every $a \in A$.

When T is a collection of topological spaces connected with continuous maps, we call it a *space tower*. When it is a collection of simplicial complexes connected with simplicial maps, we call it a *simplicial tower*, and when it is a collection of *vector spaces* connected with linear maps, we call it a *vector space tower*.

Remark 1. *As we have already seen, in practice, it may happen that a tower needs to be defined over a discrete set or more generally an index set A that is only a subposet of $\mathbb{R}$. In such a case, one can 'embed' A into $\mathbb{R}$ and convert the input to a tower according to Definition* **??** *by assuming that for any $a < a' \in A$ with $(a, a') \notin A$ and for any $a \leq b < b' < a'$, $t_{b,b'}$ is an isomorphism.*

Definition 2 (Interleaving of simplicial (space) towers)**.** Let $\mathcal{X} = \{X_a \xrightarrow{x_{a,a'}} X_{a'}\}_{a \leq a'}$ and $\mathcal{Y} = \{Y_a \xrightarrow{y_{a,a'}} Y_{a'}\}_{a \leq a'}$ be two towers of simplicial complexes (spaces resp.) indexed in $\mathbb{R}$. For any real $\varepsilon \geq 0$, we say that they are ε-interleaved if for every $a \in \mathbb{R}$ one can find simplicial maps (continuous maps resp.) $\varphi_a : X_a \to Y_{a+\varepsilon}$ and $\psi_a : Y_a \to X_{a+\varepsilon}$ so that:

(i) $\psi_{a+\varepsilon} \circ \varphi_a$ and $x_{a,a+2\varepsilon}$ are contiguous (homotopic resp.),

(ii) $\varphi_{a+\varepsilon} \circ \psi_a$ and $y_{a,a+2\varepsilon}$ are contiguous (homotopic resp.).

(iii) $\varphi_{a'} \circ x_{a,a'}$ and $y_{a+\varepsilon,a'+\varepsilon} \circ \varphi_a$ are contiguous (homotopic resp.),

(iv) $x_{a+\varepsilon,a'+\varepsilon} \circ \psi_a$ and $\psi_{a'} \circ y_{a,a'}$ are contiguous (homotopic resp.).

If no such finite ε exists, we say the two towers are ∞-interleaved.

These four conditions are summarized by requiring that the four diagrams below commute up to contiguity (homotopy resp.):

$$\begin{tikzcd}
X_a \arrow[rr, "x_{a,a+2\varepsilon}"] \arrow[dr, "\varphi_a"] & & X_{a+2\varepsilon} \\
& Y_{a+\varepsilon} \arrow[ur, "\psi_{a+\varepsilon}"] &
\end{tikzcd}
\qquad
\begin{tikzcd}
& X_{a+\varepsilon} \arrow[dr, "\varphi_{a+\varepsilon}"] & \\
Y_a \arrow[ur, "\psi_a"] \arrow[rr, "y_{a,a+2\varepsilon}"] & & Y_{a+2\varepsilon}
\end{tikzcd}
\tag{6.2}$$

$$\begin{tikzcd}
X_a \arrow[r, "x_{a,a'}"] \arrow[d, "\varphi_a"] & X_{a'} \arrow[d, "\varphi_{a'}"] \\
Y_{a+\varepsilon} \arrow[r, "y_{a+\varepsilon,a'+\varepsilon}"'] & Y_{a'+\varepsilon}
\end{tikzcd}
\qquad
\begin{tikzcd}
X_{a+\varepsilon} \arrow[r, "x_{a+\varepsilon,a'+\varepsilon}"] & X_{a'+\varepsilon} \\
Y_a \arrow[u, "\psi_a"] \arrow[r, "y_{a,a'}"'] & Y_{a'} \arrow[u, "\psi_{a'}"]
\end{tikzcd}$$

If we replace the operator '+' by the multiplication '$\cdot$' with respect to the indices in the above definition, then we say that $\mathcal{X}$ and $\mathcal{Y}$ are *multiplicatively ε-interleaved*. By interleaving we will mean additive interleaving by default and use the term multiplicative interleaving where necessary to signify that the shift is multiplicative rather than additive.

Definition 3 (Interleaving distance between simplicial (space) towers)**.** The interleaving distance between two simplicial (space) towers $\mathcal{X}$ and $\mathcal{Y}$ is:

$$\mathsf{d}_I(\mathcal{X}, \mathcal{Y}) = \inf_{\varepsilon}\{\mathcal{X} \text{ and } \mathcal{Y} \text{ are } \varepsilon\text{–}\textit{interleaved}\}.$$

Similar to the simplicial (space) towers, we can define interleaving of vector space towers. But, in that case, we replace contiguity (homotopy) with equality in conditions (i) through (iv).

Definition 4 (Interleaving of vector space towers)**.** Let $\mathbb{U} = \{\mathrm{U}_a \xrightarrow{u_{a,a'}} \mathrm{U}_{a'}\}_{a \le a'}$ and $\mathbb{V} = \{\mathrm{V}_a \xrightarrow{v_{a,a'}} \mathrm{V}_{a'}\}_{a \le a'}$ be two vector space towers indexed in $\mathbb{R}$. For any real $\varepsilon \ge 0$, we say that they are ε-interleaved if for each $a \ge r$ one can find linear maps $\varphi_a : \mathrm{U}_a \to \mathrm{V}_{a+\varepsilon}$ and $\psi_a : \mathrm{V}_a \to \mathrm{U}_{a+\varepsilon}$ so that:

(i) for all $a \in \mathbb{R}$, $\psi_{a+\varepsilon} \circ \varphi_a = u_{a,a+2\varepsilon}$,

(ii) for all $a \in \mathbb{R}$, $\varphi_{a+\varepsilon} \circ \psi_a = v_{a,a+2\varepsilon}$.

(iii) for all $a' \ge a$, $\varphi_{a'} \circ u_{a,a'} = v_{a+\varepsilon,a'+\varepsilon} \circ \varphi_a$,

(iv) for all $a' \ge a$, $u_{a+\varepsilon,a'+\varepsilon} \circ \psi_a = \psi_{a'} \circ v_{a,a'}$.

If no such finite ε exists, we say the two towers are ∞-interleaved.

Analogous to the simplicial (space) towers, if we replace the operator '+' by the multiplication '$\cdot$' in the above definition, then we say that $\mathbb{U}$ and $\mathbb{V}$ are *multiplicatively ε-interleaved*.

Definition 5 (Interleaving distance between vector space towers)**.** The interleaving distance between two towers of vector spaces $\mathbb{U}$ and $\mathbb{V}$ is:

$$\mathsf{d}_I(\mathbb{U}, \mathbb{V}) = \inf_{\varepsilon}\{\mathbb{U} \text{ and } \mathbb{V} \text{ are } \varepsilon\textit{-interleaved}\}.$$

Suppose that we have two simplicial (space) towers $\mathcal{X} = \{X_a \overset{x_{a,a'}}{\to} X_{a'})\}$ and $\mathcal{Y} = \{Y_a \overset{y_{a,a'}}{\to} Y_{a'})\}$. Consider the two vector space towers also called homology towers obtained by taking the homology groups of the complexes (spaces), that is,

$$\mathbb{V}_X = \{\mathsf{H}_p(X_a) \overset{x_{(a,a')*}}{\to} \mathsf{H}_p(X_{a'})\} \text{ and } \mathbb{V}_Y = \{\mathsf{H}_p(Y_a) \overset{y_{(a,a')*}}{\to} \mathsf{H}_p(Y_{a'})\}.$$

The following should be obvious because simplicial (continuous resp.) maps become linear maps and contiguous (homotopic resp.) maps become equal at the homology level.

Proposition 1. $\mathsf{d}_I(\mathbb{V}_X, \mathbb{V}_Y) \le \mathsf{d}_I(\mathcal{X}, \mathcal{Y})$.

One can recognize that the vector space tower is a persistence module defined earlier. Therefore, we can use interval module decomposition to define the persistence diagram $\mathrm{Dgm}\mathbb{V}$ of the tower $\mathbb{V}$. Recall that d_b denotes the bottleneck distance between persistence diagrams. *Isometry theorem* as stated earlier also hold for towers that are *tame*, that is, towers with all linear maps having finite rank.

Theorem 2. *For any two tame towers $\mathbb{U}$ and $\mathbb{V}$, we have* $\mathsf{d}_b(\mathrm{Dgm}(\mathbb{U}), \mathrm{Dgm}(\mathbb{V})) = \mathsf{d}_I(\mathbb{U}, \mathbb{V})$.

Combining Proposition 1 and Theorem 2, we obtain the following result.

Theorem 3. *Let $\mathcal{X}$ and $\mathcal{Y}$ be two simplicial (space) towers and $\mathbb{V}_X$ and $\mathbb{V}_Y$ be their homology towers respectively that are tame. Then,* $\mathsf{d}_b(\mathrm{Dgm}(\mathbb{V}_X), \mathrm{Dgm}(\mathbb{V}_Y)) \le \mathsf{d}_I(\mathcal{X}, \mathcal{Y})$.

We want to apply the above result to translate the multiplicative interleaving distances into a bottleneck distance of the persistence diagrams. For that we need to consider log scale. Given a persistence diagram Dgm, we denote its *log-scaled version* $\mathrm{Dgm}_{\log}$ to be the diagram consisting of the set of points $\{(\log x, \log y) \mid (x, y) \in \mathrm{Dgm}\}$. In log scale a multiplicative interleaving turns into an additive interleaving by which the following corollary is deduced immediately from Theorem 3.

Corollary 4. *Let $\mathcal{X}$ and $\mathcal{Y}$ be two simplicial (space) towers with a positive resolution that are multiplicatively c-interleaved and $\mathbb{V}_X$ and $\mathbb{V}_Y$ be their homology towers respectively that are tame. Then,*

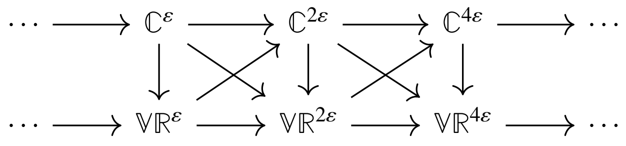
$$\mathsf{d}_b(\mathrm{Dgm}_{\log}(\mathbb{V}_X), \mathrm{Dgm}_{\log}(\mathbb{V}_Y)) \leq \log c.$$

Interleaving between Čech and Rips filtrations: We show an example where we can use the stability result in Theorem 3. Let $P \subseteq M$ be a finite subset of a metric space (M, d). Consider the Rips and Čech-filtrations:

$$\mathcal{R} : \{\mathbb{VR}^{\varepsilon}(P) \hookrightarrow \mathbb{VR}^{\varepsilon'}(P)\}_{0<\varepsilon\leq\varepsilon'} \text{ and } \mathcal{C} : \{\mathbb{C}^{\varepsilon}(P) \hookrightarrow \mathbb{C}^{\varepsilon'}(P)\}_{0<\varepsilon\leq\varepsilon'}.$$

We know that the following inclusions hold.

$$\mathbb{C}^{\varepsilon}(P) \subseteq \mathbb{VR}^{\varepsilon}(P) \subseteq \mathbb{C}^{2\varepsilon}(P).$$



$$\begin{tikzcd}
\cdots \arrow[r] & \mathbb{C}^{\varepsilon} \arrow[r] \arrow[d] \arrow[dr] & \mathbb{C}^{2\varepsilon} \arrow[r] \arrow[d] \arrow[dr] & \mathbb{C}^{4\varepsilon} \arrow[r] \arrow[d] & \cdots \\
\cdots \arrow[r] & \mathbb{VR}^{\varepsilon} \arrow[r] \arrow[ur] & \mathbb{VR}^{2\varepsilon} \arrow[r] \arrow[ur] & \mathbb{VR}^{4\varepsilon} \arrow[r] & \cdots
\end{tikzcd}$$

Figure 6.1: Čech and Rips complexes interleave multiplicatively.

Figure 6.1 illustrates that Čech an Rips complexes are multiplicatively 2-interleaved. Then, according to Corollary 4, the persistence diagrams $\mathrm{Dgm}_{\log}\mathcal{C}$ and $\mathrm{Dgm}_{\log}\mathcal{R}$ have bottleneck distance of $\log 2 = 1$.

6.2 Computing persistence of simplicial towers

In this section, we present an algorithm for computing the persistence of a simplicial tower. Consider a simplicial tower $\mathcal{K} : K_0 \xrightarrow{f_0} K_1 \xrightarrow{f_1} K_2 \cdots \xrightarrow{f_{n-1}} K_n$ and the map $f_{ij} : K_i \to K_j$ where $f_{ij} = f_{j-1} \circ \cdots \circ f_{i+1} \circ f_i$. To compute the persistent homology for a simplicial filtration, the persistence algorithm in the previous chapter essentially maintains a consistent basis by computing the image $f_{ij_*}(B_i)$ of a basis B_i of $\mathsf{H}_*(K_i)$. As the algorithm moves through an inclusion in the filtration, the homology basis elements get created (birth) or are destroyed (death). Here, for towers, instead of a consistent homology basis, we maintain a consistent cohomology basis. We need to be aware that, for cohomology, the induced maps from $f_{ij} : K_i \to K_j$ are reversed, that is, $f^*_{ij} : \mathsf{H}^*(K_i) \leftarrow \mathsf{H}^*(K_j)$. So, if B^i is a cohomology basis of $\mathsf{H}^*(K_i)$ maintained by the algorithm, it computes implicitly the preimage $f^{*-1}_{ij}(B^i)$. Dually, this implicitly maintains a consistent homology basis and thus captures all information about persistent homology as well.

6.2.1 Annotations

We maintain a consistent cohomology basis using a notion called annotations [2] which are binary vectors assigned to simplices. These annotations are updated as we go forward through the sequence in the given tower. This implicitly maintains a cohomology basis in the reverse direction whose birth and death coincide with the death and birth respectively of a consistent homology basis.

Definition 6 (Annotation and valid annotation)**.** Given a simplicial complex K, Let $K(p)$ denote the set of p-simplices in K. An *annotation* for $K(p)$ is an assignment $\mathsf{a} : K(p) \to \mathbb{Z}_2^g$ of a binary vector $\mathsf{a}_\sigma = \mathsf{a}(\sigma)$ of length g to each p-simplex $\sigma \in K$. The binary vector a_σ is called the annotation for σ. Each entry '0' or '1' of a_σ is called its element. Annotations for simplices provide an annotation for every p-chain c_p: $\mathsf{a}_{c_p} = \Sigma_{\sigma \in c_p} \mathsf{a}_\sigma$.

An annotation $\mathsf{a} : K(p) \to \mathbb{Z}_2^g$ is *valid* if following two conditions are satisfied:

1. $g = \text{rank}\, \mathsf{H_p(K)}$, and
2. two p-cycles z_1 and z_2 have $\mathsf{a}_{z_1} = \mathsf{a}_{z_2}$ iff their homology classes are identical, i.e. $[z_1] = [z_2]$.

Proposition 5. *The following two statements are equivalent.*

1. *An annotation* $\mathsf{a} : K(p) \to \mathbb{Z}_2^g$ *is valid*
2. *The cochains* $\{\phi_i\}_{i=1,\cdots,g}$ *given by* $\phi_i(\sigma) = \mathsf{a}_\sigma[i]$ *for every* $\sigma \in K(p)$ *are cocycles whose cohomology classes* $\{[\phi_i]\}, i = 1, \ldots, g$*, constitute a basis of* $\mathsf{H}^p(K)$.

In light of the above result, an annotation is simply one way to represent a cohomology basis. However, by representing the corresponding basis as an explicit vector associated with each simplex, it localizes the basis to each simplex. As a result, we can update the cohomology basis locally by changing the annotations locally (see Proposition 8). This point of view also helps to reveal how we can process elementary collapses, which are neither inclusions nor deletions, by transferring annotations (see Proposition 9).

6.2.2 Algorithm

Consider the persistence module $\mathsf{H}_*\mathcal{K}$ induced by a simplicial tower $\mathcal{K} : \{K_i \xrightarrow{f_i} K_{i+1}\}$ where every f_i is a so-called elementary simplicial map which we will introduce shortly.

$$\mathsf{H}_*\mathcal{K} : \ \mathsf{H}_*(K_0) \xrightarrow{f_{0*}} \mathsf{H}_*(K_1) \xrightarrow{f_{1*}} \mathsf{H}_*(K_2) \cdots \xrightarrow{f_{n-1*}} \mathsf{H}_*(K_n)$$

Instead of tracking a consistent homology basis for the module $\mathsf{H}_*\mathcal{K}$, we track a cohomology basis in the dual module $\mathsf{H}^*\mathcal{K}$ where the homomorphisms are in reverse direction:

$$\mathsf{H}^*\mathcal{K}:\ \mathsf{H}^*(K_0) \xleftarrow{f_0^*} \mathsf{H}^*(K_1) \xleftarrow{f_1^*} \mathsf{H}^*(K_2) \cdots \xleftarrow{f_{n-1}^*} \mathsf{H}^*(K_n)$$

As we move from left to right in the above sequence, the annotations *implicitly maintain a cohomology basis* whose elements are also *time stamped* to signify when a basis element is born or dies. We should keep in mind that the *birth* and *death* of a cohomology basis element coincides with the *death* and *birth* of a homology basis element because the two modules run in opposite directions.

To jump start the algorithm, we need annotations for simplices in K_0 at the beginning. This can be achieved by considering an arbitrary filtration of K_0 and then applying the generic algorithm as we describe for inclusions in Section 6.2.3. The first vertex in this filtration gets the annotation of [1].

Before describing the algorithm, we observe a simple fact that simplicial maps can be decomposed into elementary maps which let us design simpler atomic steps for the algorithm.

Definition 7 (Elementary simplicial maps)**.** A simplicial map $f : K \to K'$ is called *elementary* if it is of one of the following two types:

- f is injective, and K' has at most one more simplex than K. In this case, f is called an *elementary inclusion*.
- f is not injective but is surjective, and the vertex map f_V is injective everywhere except on a pair $\{u, v\} \subseteq V(K)$. In this case, f is called an *elementary collapse*. An elementary collapse maps a pair of vertices into a single vertex, and is injective on every other vertex.

We observe that any simplicial map is a composition of elementary simplicial maps.

Proposition 6. *If $f : K \to K'$ is a simplicial map, then there are elementary simplicial maps f_i*

$$K = K_0 \xrightarrow{f_0} K_1 \xrightarrow{f_2} K_2 \cdots \xrightarrow{f_{n-1}} K_n = K' \text{ so that } f = f_{n-1} \circ f_{n-2} \circ \cdots \circ f_0.$$

In view of Proposition 6, it is sufficient to show how one can design the persistence algorithm for an elementary simplicial map. At this point, we make a change in the definition 7 of elementary simplicial maps that eases further discussions. We let f_V to be identity (which is an injective map) everywhere except possibly on a pair of vertices $\{u, v\} \subseteq V(K)$ for which f_V maps to one of these two vertices, say u, in K'. This change can be implemented by renaming the vertices in K' that are mapped onto injectively.

6.2.3 Elementary inclusion

Consider an elementary inclusion $K_i \hookrightarrow K_{i+1}$. Assume that K_i has a valid annotation. We describe how we obtain a valid annotation for K_{i+1} from that of K_i after inserting the p-simplex $\sigma = K_{i+1} \setminus K_i$. We compute the annotation $\mathsf{a}_{\partial\sigma}$ for the boundary $\partial\sigma$ in K_i and take actions as

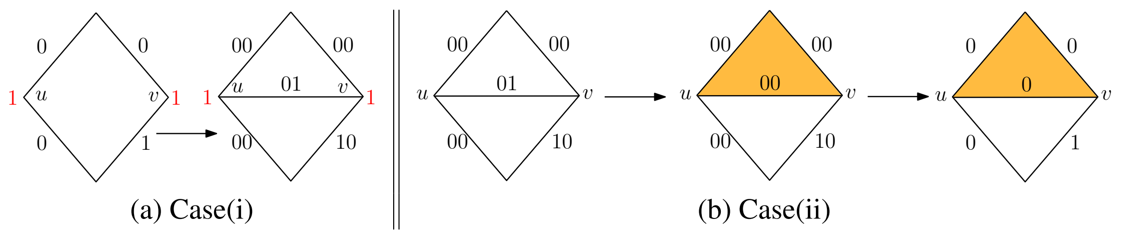


Figure 6.2: Case(i) of inclusion: the boundary $\partial uv = u + v$ of the edge uv has annotation $1 + 1 = 0$. After its addition, every edge gains an element in its annotation which is 0 for all except the edge uv. Case (ii) of inclusion: the boundary of the top triangle has annotation 01. It is added to the annotation of uv which is the only edge having the second element 1. Consequently the second element is zeroed out for every edge, and is then deleted.

follows.

Case (i): If $\mathsf{a}_{\partial\sigma}$ is a zero vector, the class $[\partial\sigma]$ is trivial in $\mathsf{H}_{p-1}(K_i)$. This means that σ creates a p-cycle in K_{i+1} and by duality a p-cocycle is killed while going left from K_{i+1} to K_i. In this case we augment the annotations for all p-simplices by one element with a time stamp $i + 1$, that is, the annotation $[b_1, b_2, \cdots, b_g]$ for every p-simplex τ is updated to $[b_1, b_2, \cdots, b_g, b_{g+1}]$ with b_{g+1} being time stamped $i + 1$. We set $b_{g+1} = 0$ for $\tau \neq \sigma$ and $b_{g+1} = 1$ for $\tau = \sigma$. The element b_i of a_σ is set to zero for $1 \leq i \leq g$. Other annotations for other simplices remain unchanged. See Figure 6.2(a).

Case (ii): If $\mathsf{a}_{\partial\sigma}$ is not a zero vector, the class of the $(p-1)$-cycle $\partial\sigma$ is nontrivial in $\mathsf{H}_{p-1}(K_i)$. Therefore, σ kills the class of this $(p-1)$-cycle and a corresponding class of $(p-1)$-cocycles is born in the reverse direction. We simulate it by forcing $\mathsf{a}_{\partial\sigma}$ to be zero which affects other annotations as well. Let $i_1, i_2, \cdots, i_k$ be the set of indices in non-decreasing order so that $b_{i_1}, b_{i_2}, \cdots, b_{i_k}$ are all of the nonzero elements in $\mathsf{a}_{\partial\sigma} = [b_1, b_2, \cdots, b_{i_k}, \cdots, b_g]$. Recall that ϕ_j denotes the $(p-1)$-cocycle given by its evaluation $\phi_j(\sigma') = \mathsf{a}_{\sigma'}[j]$ for every $(p-1)$-simplex $\sigma' \in K_i$ (Proposition 5). With this notation, the cocycle $\phi = \phi_{i_1} + \phi_{i_2} + \cdots + \phi_{i_k}$ is born after deleting σ in the reverse direction. This cocyle does not exist after time b_{i_k} in the reverse direction. In other words, the cohomology class $[\phi]$ which is born leaving the time $i + 1$ is killed at time b_{i_k}. This pairing matches that of the standard persistence algorithm where the youngest basis element is chosen to be paired among all those ones whose combination is killed. We add the vector $\mathsf{a}_{\partial\sigma}$ to the annotation of every $(p-1)$-simplex whose i_k-th element is nonzero. This zeroes out the i_k-th element of the annotation for every $(p-1)$-simplex and at the same time updates other elements so that a valid annotation according to Proposition 5 is maintained. We simply delete i_k-th element from the annotation for every $(p-1)$-simplex. See Figure 6.2(b). We further set the annotation a_σ for σ to be a zero-vector of length s, where s is the length of the annotation vector of every p-simplex at this point.

Notice that determining if we have case (i) or (ii) can be done easily in $O(pg)$ time by checking the annotation of $\partial\sigma$. Indeed, this is achieved because the annotation already localizes the cohomology basis to each individual simplex.

Before going to the next case of elementary collapse, here we present Algorithm 1:Annot for computing the annotations for all simplices in a given simplicial complex using the setps of elementary inclusions. The algorithm proceeds with increasing dimension because it needs to have the annotaions of $(p-1)$-simplices before dealing with p-simplices. It starts with vertices whose annotations are readily computable. In the following algorithm K^p denotes the p-skeleton of the input simplicial d-complex K.

Algorithm 1 Annot(K)

```
Input:
  $K$: input complex
Output:
  Annotation for every simplex in $K$
1:  Let $m := |K^0|$
2:  For every vertex $v_i \in K^0$, assign an $m$-vector $\mathsf{a}(v_i)$ where $\mathsf{a}(v_i)[j] = 1$ iff $j = i$
3:  for $p = 1 \to d$ do
4:      for all simplex $\sigma \in K^p$ do
5:          Let annotation of every $p$-simplex be a vector of length $g$ so far
6:          if $\mathsf{a}(\partial\sigma) \neq 0$ then
7:              assign $\mathsf{a}(\sigma)$ to be a 0 vector of size $g$
8:              pick any non-zero entry $b_u$ in $\mathsf{a}(\partial\sigma)$
9:              add $\mathsf{a}(\partial\sigma)$ to every $(p-1)$-simplex $\sigma'$ s.t. $\mathsf{a}(\sigma')[u] = 1$
10:             delete $u$th entry from annotation of every $(p-1)$-simplex
11:         else
12:             extend $\mathsf{a}(\tau)$ for every $p$-simplex $\tau$ so far added by appending a 0 bit
13:             create vector $\mathsf{a}(\sigma)$ of length $g+1$ with only the last bit being 1
14:         end if
15:     end for
16: end for
```

6.2.4 Elementary collapse

The case for handling collapse is more interesting. It has three distinct steps, (i) elementary inclusions to satisfy the so called link condition, (ii) local annotation transfer to prepare for the collapse, and (iii) collapse of the simplices with updated annotations. We explain each of these steps now.

The elementary inclusions that may precede the final collapse are motivated by a result that connects collapses with the change in (co)homology. Consider an elementary collapse $K_i \xrightarrow{f_i} K_{i+1}$ where the vertex pair (u, v) collapses to u. The following link condition, introduced in [3] and later used to preserve homotopy [1], becomes relevant.

Definition 8. A vertex pair (u, v) in a simplicial complex K_i satisfies the *link condition* if the edge $uv \in K_i$ and $\mathrm{Lk}\, u \cap \mathrm{Lk}\, v = \mathrm{Lk}\, uv$. An elementary collapse $f_i : K_i \to K_{i+1}$ satisfies the link condition if the vertex pair on which f_i is not injective satisfies the link condition.

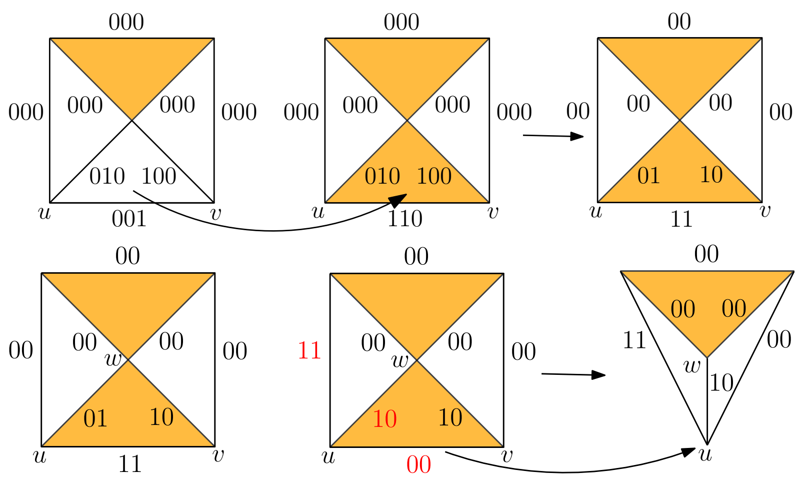


Figure 6.3: Annotation updates for elementary collapse: inclusion of a triangle so as to satisfy the link condition (upper row), annotation transfer and actual collapse (lower row); annotation 11 of the vanishing edge *uv* is added to all edges (cofaces) adjoining *u*.

Proposition 7 ([1])**.** *If an elementary collapse $f_i : K_i \to K_{i+1}$ satisfies the link condition, then the underlying spaces $|K_i|$ and $|K_{i+1}|$ remain homotopy equivalent. Hence, the induced homomorphisms $f_{i_*} : \mathsf{H}_*(K_i) \to \mathsf{H}_*(K_{i+1})$ and $f_i^* : \mathsf{H}^*(K_i) \leftarrow \mathsf{H}^*(K_{i+1})$ are isomorphisms.*

If an elementary collapse satisfies the link condition, we can perform the collapse knowing that the (co)homology does not change. Otherwise, we know that the (co)homology is affected by the collapse and it should be reflected in our updates for annotations.

$$\begin{tikzcd} K_i \arrow[r, "f_i"] \arrow[dr, "j"'] & K_{i+1} \\ & \hat{K}_i \arrow[u, "f_i'"'] \end{tikzcd}$$

The diagram at the left provides a precise means to carry out the change in (co)homology. Let S be the set of simplices in non-decreasing order of dimensions, whose absence from K_i makes (u, v) violate the link condition. For each such simplex $\sigma \in S$, we modify the annotations of every simplex which we would have done if σ were to be inserted. Thereafter, we carry out the rest of the elementary collapse. In essence, implicitly, we obtain an intermediate complex $\hat{K}_i = K_i \cup S$ where the diagram on the left commutes. Here, f_i' is induced by the same vertex map that induces f_i, and j is an inclusion. This means that the persistence of f_i is identical to that of $f_i' \circ j$ which justifies our action of elementary inclusions followed by the actual collapses.

We remark that this is the only place where we may insert implicitly a simplex σ in the current approach. The number of such σ is usually much smaller than the number of simplices that one may need for a *coning* strategy detailed in [5] to process simplicial towers.

Next, we transfer annotations in $\hat{K}_i$. This step locally changes the annotations for simplices containing the vertices u and/or v. The following definition facilitates the description.

Definition 9. For an elementary collapse $f_i : K_i \to K_{i+1}$, a simplex $\sigma \in K_i$ is called *vanishing* if the cardinality of $f_i(\sigma)$ is one less than that of σ. Two simplices σ and σ' are called *mirror* pairs if one contains u and the other v, and share rest of the vertices. In Figure 6.3(lower row), the vanishing simplices are $\{uv, uvw\}$ and the mirror pairs are $\{u, v\}$, $\{uw, vw\}$.

In an elementary collapse that sends (u, v) to u, all vanishing simplices need to be deleted, and all simplices containing v need to be pulled to corresponding ones containing the vertex u (which are their mirror partners). We update the annotations in such a way that the annotations of all vanishing simplices become zero, and those of each pair of mirror simplices become the same. Once this is achieved, the collapse is implemented by simply deleting the vanishing simplices and replacing v with u in all simplices containing v without changing their annotations. The following proposition provides the justification behind the specific update operations that we perform.

Proposition 8. *Let K be a simplicial complex and $\mathsf{a} : K(p) \to \mathbb{Z}_2^g$ be a valid annotation. Let $\sigma \in K(p)$ be any p-simplex and τ any of its $(p-1)$-faces. Then, adding a_σ to the annotation for all cofaces of τ of codimension 1 (including σ) produces a valid annotation for $K(p)$. Furthermore, the cohomology basis corresponding to the annotations (Proposition 5) remains unchanged by this modification.*

Consider an elementary collapse $f_i : K_i \to K_{i+1}$ that sends (u, v) to u. We update the annotations for simplices in K_i as follows. First, note that the vanishing simplices are exactly those simplices containing the edge $\{u, v\}$. For every p-simplex containing $\{u, v\}$, i.e., a vanishing simplex, exactly two of its $(p - 1)$-faces are mirror simplices, and all other remaining $(p - 1)$-faces are vanishing simplices. Let σ be a vanishing p-simplex and τ be its $(p - 1)$-face that is a mirror simplex containing u. We add a_σ to the annotations for all cofaces of τ of codimension 1 including σ. We call this an *annotation transfer* for σ. By Proposition 8, the new annotation generated by this process corresponds to the old cohomology basis for K_i. This new annotation has a_σ as zero since $\mathsf{a}_\sigma + \mathsf{a}_\sigma = 0$. See the the lower row of Figure 6.3. We perform the above operation for each vanishing simplex. It turns out that by using the relations of vanishing simplices and mirror simplices, each mirror simplex eventually acquires an identical annotation to that of its partner. Specifically, we have the following observation.

Proposition 9. *After all possible annotation transfers involved in a collapse, (i) each vanishing simplex has a zero annotation; and (ii) each mirror simplex τ has the same annotation as its mirror partner simplex τ'.*

Subsequent to the annotation transfer, the annotation of $\hat{K}_i$ fits for actual collapse since each pair of mirror simplices which are collapsed to a single simplex get the identical annotation and the vanishing simplex acquires the zero annotation. Furthermore, Proposition 8 tells us that the cohomology basis does not change by annotation transfer which aligns with the fact that $f_i'^* : \mathsf{H}^*(K_{i+1}) \to \mathsf{H}^*(\hat{K}_i)$ is indeed an isomorphism. Accordingly, no time stamp changes after the annotation transfer and the actual collapse. Propositions 5.2 and 5.3 in [4] provide formal statements justifying the algorithm for annotation updates.

The persistence diagram of a given simplicial tower $\mathcal{K}$ can be retrieved easily from the annotation algorithm. Each time during an elementary operation either we add a new element into the annotation of all p-simplices for some $p \geq 0$ or delete an element from the annotations of all of them. During the deletion, we add the point (bar) (a, b) into $\mathrm{Dgm}_p\mathcal{K}$ where b is the current time of deletion (death) and a is the time stamp of the element when it was added (birth).

6.3 Notes and Exercises

Computation of persistent homology induced by simplicial towers generalizing filtrations were first considered in the context of TDA by Dey, Fan, Wang [4]. They gave two approaches to compute persistence diagrams for such towers, one by converting a tower to a zigzag filtration which we will see later and the other by considering annotations in combination with the link conditions allowing edge collapses without altering homotopy types which is described in Section 6.2.1. The first approach apparently increases the size of the filtration which motivated the second approach. Kerber and Schreiber showed that indeed the first approach can be leveraged to produce filtrations instead of zigzag filtrations and without blowing up sizes [6].

Exercises

1. Show that the inequality in Proposition 1 cannot be imporved to equality by giving a counterexample.
2. Prove Proposition 5
3. Prove Proposition 6
4. Prove Proposition 8
5. Prove Proposition 7
6. For computing the persistence of towers, we checked the link condition for all dimensions. Argue that it is sufficient to check the condition only for three relevant dimensions.

Bibliography

[1] Dominique Attali, André Lieutier, and David Salinas. Efficient data structure for representing and simplifying simplicial complexes in high dimensions. In *Proc. 27th Annu. ACM Sympos. Comput. Geom.*, SoCG '11, pages 501–509, New York, NY, USA, 2011. ACM.

[2] Oleksiy Busaryev, Sergio Cabello, Chao Chen, Tamal K. Dey, and Yusu Wang. Annotating simplices with a homology basis and its applications. In *Algorithm Theory - SWAT 2012 - 13th Scandinavian Symposium and Workshops, Helsinki, Finland, July 4-6, 2012. Proceedings*, pages 189–200, 2012.

[3] Tamal K. Dey, Herbert Edelsbrunner, Sumanta Guha, and Dmitry V. Nekhayev. Topology preserving edge contraction. *Publications de l' Institut Mathematique (Beograd)*, 60:23–45, 1999.

[4] Tamal K. Dey, Fengtao Fan, and Yusu Wang. Computing topological persistence for simplicial maps. *CoRR*, abs/1208.5018, 2012.

[5] Tamal K. Dey, Fengtao Fan, and Yusu Wang. Computing topological persistence for simplicial maps. In *Proceedings of the Thirtieth Annual Symposium on Computational Geometry*, SOCG'14, pages 345:345–345:354, New York, NY, USA, 2014. ACM.

[6] Michael Kerber and Hannah Schreiber. Barcodes of towers and a streaming algorithm for persistent homology. *Discrete & Computational Geometry*, 61(4):852–879, Jun 2019.