Faculty of Mathematics
University of Duisburg-Essen
Thea-Leymann-Straße 9, 45127 Essen

UNIVERSITÄT DUISBURG ESSEN

Open-Minded

Master's Thesis

Localizing Holes in Data Sets Using Persistent Homology

Mario Günzel
3015730

Supervisor:
Prof. Dr. Jochen Heinloth

Co-Supervisor:
Dr. Daniel Harrer

October 21, 2019

Contents

1 Introduction

Analyzing data sets in 2 or 3 dimensions can be achieved by using a visualization tool. We are able to count the connected components and search for holes and tunnels to obtain an understanding of the structure of data sets. In higher dimensions we cannot

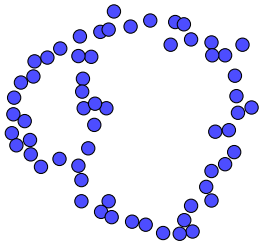


Figure 1.1: A data set with two holes and one connected component.

follow this approach since we do not have an intuition for the definition of a hole. The tool to formalize this notion in arbitrary dimensions is the topological invariant *homology*. We will state an algorithm for its computation.

The foundation for this algorithm is data in form of a finite point cloud in some metric space as the real coordinate space $\mathbb{R}^n$. For each set of k data points we add a k-simplex if balls around the points with a given radius intersect pairwise. As a

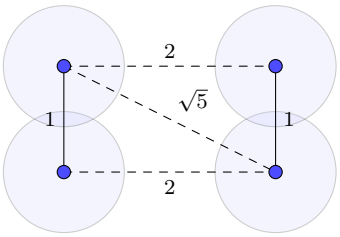


Figure 1.2: Simplicial complex obtained from points.

result we obtain a *simplicial complex*, for which its homology is defined as in [Hat02, Section 2.1].

The constructed complex depends crucially on the prescribed radius. Increasing the radius enlarges the complex. We store this information as a sequence of inclusions of simplicial complexes. By going through the sequence, new holes can be formed and holes can vanish. We capture this behavior by *persistent homology*, which can distinguish whether a hole survives. Later on, we will be able to draw *barcodes*, which describe the lifetimes of holes in the sequence by intervals as in Figure 1.3. In the right picture we see one hole for radii in the interval $[1, \frac{\sqrt{5}}{2})$.

In this thesis we will study and implement an algorithm to compute persistent homology of such sequences of simplicial complexes. We will use it to compute the homology

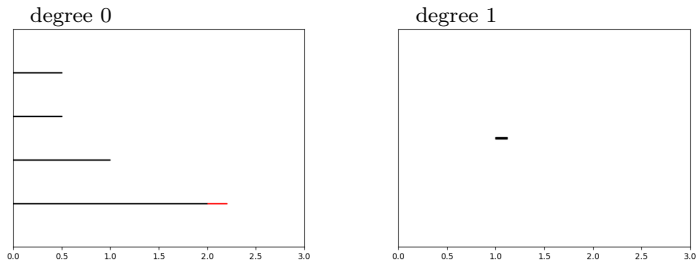


Figure 1.3: Barcodes for Figure 1.2.

of finitely many points on a circle. In contrast to the case in [AA17] for infinitely many points, we do not only obtain the homology classes of odd-dimensional spheres.

To get further insights into the structure of data sets, we want to localize its holes. For this purpose, we partition the complex into smaller pieces and construct a sequence having the local parts at its ground level and the *Mayer-Vietoris blowup*, which is homotopy equivalent to the complex itself, at its highest level. By computing persistent homology of this sequence, we obtain a good local description of the holes.

Persistent homology and localized homology are important tools in topological data analysis. They are widely used in various scientific areas for example to analyze the structure of proteins [XW14], for fast tumor segmentation in medicine [QTT+18] or to improve machine learning models [GND+19].

2 Complexes and Homology

For this thesis a basic knowledge of algebraic topology is required. We want to recall the most important definitions and shortly discuss our understanding of *holes* in higher dimensions. At the end of this chapter we are going to study ways to make homology an available tool for data sets. We mainly follow [Hat02] and [EH10] for foundations on complexes and homology.

2.1 Singular Homology

Homology is a topological invariant, which can be used to describe holes of general topological spaces. Let us consider a hole in the two dimensional real space. We can describe it by constructing a rectangle as a sum of lines around the hole. The rectangle defines a hole if it cannot be filled completely.

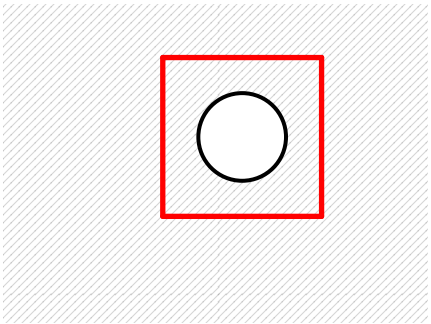


Figure 2.1: Describing a hole by a sum of lines.

We note that there might be several descriptions for this hole like a deformation of the rectangle or another polygon. By using homology, we do not distinguish these cases. We take a quotient that identifies all these different polygons whose pairwise differences can be filled. A similar approach will be used in higher dimensions n, where we work with general $(n-1)$-simplices instead of lines. Now we put the idea into a formal definition:

Definition 2.1 (Simplex). For all $n \in \mathbb{Z}_{\geqslant 0}$ we define an *n-simplex* $\Delta^n \subseteq \mathbb{R}^d, d \in \mathbb{Z}_{\geqslant 0}$ to be the convex hull of $n+1$ affinely independent vertices. The *dimension* of an n-simplex Δ^n is n. There are the following standard constructions:

(1) An n-simplex can be described as a subset of $\mathbb{R}^n$ by taking the convex hull of the vertices $\{e^i\}_{i=0,\dots,n}$, where $e^0 = (0,\dots,0)$ and $e^i, i \neq 0$ are the standard basis vectors of $\mathbb{R}^n$.

(2) In a similar way, we obtain an n-simplex in $\mathbb{R}^{n+1}$ as convex hull of the basis vectors $e^1,\dots,e^{n+1}$.

(3) If we have an n-simplex as convex hull of the vertices $\{v_0, \dots, v_n\}$, then each subset $\varnothing \neq J \subseteq \{0, \dots, n\}$ defines a *subsimplex* of dimension $|J| - 1$ by taking the convex hull of the vertices $\{v_j\}_{j \in J}$:

$$\iota_J : \Delta^J := \operatorname{conv}\{v_j\}_{j \in J} \hookrightarrow \Delta^n$$

A subsimplex of dimension $n - 1$ is called a *face* of the simplex.

We note that an 1-simplex is just a line and a 2-simplex is a triangle. For the sake of better readability we write $[n] := \{0, \dots, n\}$ in the following.

Definition 2.2 (Singular chain complex). Let X be a topological space. For all $n \in \mathbb{Z}_{\geqslant 0}$, we define a *singular n-chain* to be a finite formal sum $\sigma = \sum_{i=1}^{m} \lambda_i \sigma_i$ of continuous maps $\sigma_i : \Delta^n \to X$ with coefficients λ_i in $\mathbb{Z}$. The *set of all n-chains* in X is defined by $C_n(X)$. It is a free $\mathbb{Z}$-module and has the set of all continuous maps $\Delta^n \to X$ as basis. Furthermore, we set $C_n(X) := 0$ for all $n \in \mathbb{Z}_{<0}$. For each basis element $\sigma \in C_n(X)$, we define its boundary as

$$\partial_n(\sigma) = \begin{cases} 0 & , n = 0 \\ \sum_{j=0}^{n} (-1)^j \sigma \circ \iota_{[n]-\{j\}} & , \text{else} \end{cases} \in C_{n-1}(X)$$

where $\iota_{[n]-\{j\}}$ is the inclusion of a subsimplex as in Definition 2.1 (3). This yields a *boundary map* $\partial_n : C_n(X) \to C_{n-1}(X)$ for all $n \in \mathbb{Z}$ by extending linearly. We obtain a sequence

$$\dots \xrightarrow{\partial_{n+1}} C_n(X) \xrightarrow{\partial_n} C_{n-1}(X) \xrightarrow{\partial_{n-1}} \dots \xrightarrow{\partial_1} C_0(X) \xrightarrow{\partial_0} 0 \xrightarrow{\partial_{-1}} \dots,$$

which is called the *singular chain complex* $(C_\bullet(X), \partial_\bullet)$.

We note that the rectangle from Figure 2.1 can be regarded as a singular 1-chain. It is in $\ker(\partial_1)$ since the endpoints of the lines cancel out, and since it cannot be filled, it is not in the image of ∂_2. Formalizing this leads to

Definition 2.3 (Singular homology). The n-th *singular homology* of a topological space X is defined as the quotient module

$$H_n(X) := \ker(\partial_n) / \operatorname{im}(\partial_{n+1})$$

for all $n \in \mathbb{Z}$.

Each generator of the simplicial homology $H_n(X)$ for $n \in \mathbb{Z}_{\geqslant 0}$ represents an n-dimensional hole of the topological space X since it is a sum of n-simplices whose boundary vanishes and which cannot be filled by $(n + 1)$-simplices. By definition of the singular chains we have $H_n(X) = 0$ for all $n \in \mathbb{Z}_{<0}$.

2.2 Simplicial Homology

We already know how to describe holes in arbitrary dimensions by using singular homology. The problem is that singular chain complexes are not algorithmically computable since their chain modules have infinitely many basis elements. We use *simplicial homology* to get a finite description of the chain groups.

To define simplicial homology we need a triangulation of the topological space. Since we construct topological spaces for data sets in Section 2.4 as unions of simplices, they are equipped with such a triangular structure.

Definition 2.4 (Simplicial complex). Let $d \in \mathbb{Z}_{\geqslant 0}$. We call a set K of simplices in $\mathbb{R}^d$ a *simplicial complex* if for all simplices $\sigma \in K$ each face of σ is in K and for all simplices $\sigma, \eta \in K$ their union $\sigma \cap \eta$ is either empty or also in K.

Each simplex is uniquely defined by its vertices. This observation leads to the following

Definition 2.5 (Abstract simplicial complex). A family A of finite sets is called an *abstract simplicial complex* if for all sets $a \in A$ each subset $b \subseteq a$ is again in the collection A. We call a set $a \in A$ an *abstract simplex*. Each subset $b \subseteq a$ with cardinality $|b| = |a| - 1$ is a *face* of the abstract simplex.

An abstract simplicial complex is a simplicial complex without an associated geometry. We do not have to take care of intersections and it is easy to store those simplices and compute their boundaries, which is why they are more useful for our purposes. Every simplicial complex can be viewed as an abstract simplicial complex by replacing each simplex by the set of its vertices. Furthermore, every finite abstract simplicial complex with d vertices can be embedded into $\mathbb{R}^{d-1}$ by defining each simplex as a subsimplex of $\Delta^{d-1} \subseteq \mathbb{R}^{d-1}$. Hence, it can be viewed as a simplicial complex. We conclude that for finitely many simplices, the definitions of a simplicial complex and an abstract simplicial complex are equivalent. Since we construct complexes out of finite data sets, in many cases we will not distinguish between those two definitions.

To make a similar construction as in Definition 2.2, we have to equip the simplices with an orientation. This can be realized by ordering each set representing an abstract simplex and referring to them as tuples.

Definition 2.6 (Simplicial chain complex). Let $\mathcal{K}$ be an (abstract) simplicial complex. Then $C_n(\mathcal{K})$ for $n \in \mathbb{Z}_{\geqslant 0}$ is the set of all $\mathbb{Z}$-linear combinations of oriented n-simplices in $\mathcal{K}$, where similar simplices with different orientation will be identified up to a sign in the following way: Let $(v_0, \ldots, v_n)$ and $(\widetilde{v}_0, \ldots, \widetilde{v}_n)$ be oriented simplices which differ by a permutation π. Then they are identified up to multiplication with the signum $\mathsf{sign}(\pi)$ of the permutation. Formally, we have

$$C_n(\mathcal{K}) := \bigoplus_{\substack{\{v_0,\ldots,v_n\}\in\mathcal{K} \\ n\text{-simplex}}} \mathbb{Z} \cdot (v_0, \ldots, v_n) \Big/ \sim$$

where $\sim$ is the identification described above. Furthermore, for all $n \in \mathbb{Z}_{<0}$ we set $C_n(\mathcal{K}) := 0$. We can define a boundary map $\partial_n : C_n(\mathcal{K}) \to C_{n-1}(\mathcal{K})$ by mapping each

oriented simplex $(v_0, \dots, v_n)$ to

$$\partial_n(v_0, \dots, v_n) := \sum_{i=0}^{n} (-1)^i (v_0, \dots, v_{i-1}, v_{i+1}, \dots, v_n)$$

and extending the map linearly. The boundary map is a well-defined map on the quotient, which can easily be checked by the reader[1]. We call the sequence

$$(C_\bullet(\mathcal{K}), \partial_\bullet) = (\ \dots \longrightarrow C_i(\mathcal{K}) \xrightarrow{\partial_i} C_{i-1}(\mathcal{K}) \xrightarrow{\partial_{i-1}} \dots \xrightarrow{\partial_1} C_0(\mathcal{K}) \xrightarrow{\partial_0} 0 \xrightarrow{\partial_{-1}} \dots\ )$$

the *simplicial chain complex* of $\mathcal{K}$.

Definition 2.7 (Singular homology). Let $\mathcal{K}$ be an (abstract) simplicial complex. For all $n \in \mathbb{Z}$, we define its n-th *simplicial homology* as

$$H_n(\mathcal{K}) := \ker(\partial_n) \big/ \mathrm{im}(\partial_{n+1}) ,$$

where $(C_\bullet(\mathcal{K}), \partial_\bullet)$ is the corresponding simplicial chain complex.

Definition 2.8 (Geometric realization). For a simplicial complex K we define by

$$|K| := \bigcup_{\sigma \in K} \sigma$$

its *geometric realization*.

If A is an abstract simplicial complex obtained from K_A by replacing each simplex by its set of vertices, we define $|A| := |K|$ to be its *geometric realization*.

Lemma 2.9. *Let X be an (abstract) simplicial complex. Then the singular homology $H_n(|X|)$ of the geometric realization of the complex is equivalent to the simplicial homology $H_n(X)$ of the complex for all $n \in \mathbb{Z}$.*

Proof. We refer to [Hat02, Theorem 2.27] with $A = \varnothing$. □

Remark 2.10. We can define chain complexes and homology of simplicial and singular chain complexes with coefficients in a general ring or a field similarly. For our algorithms we will use coefficients in $\mathbb{F}_2$, since those chains are very easy to compute. We should be aware of the fact that this way some information gets lost as mentioned in [Cro05, Chapter 9.2].

In the following, let all rings be commutative rings with 1 if we do not specify them differently.

[1] If two oriented simplices differ by a permutation, then the permutation can be written as a composition of transpositions. It suffices to show that

$$\partial(v_0, \dots, v_i, v_{i+1}, \dots, v_n) = -\partial(v_0, \dots, v_{i+1}, v_i, \dots, v_n).$$

2.3 More on Homology and Chain Complexes

Singular and simplicial homology are just special cases of the general concept of homology obtained by chain complexes. We also want to introduce this general definition shortly.

Definition 2.11 (Chain complex). We define a *chain complex* $(A_\bullet, \partial_\bullet)$ as a sequence of modules

$$\ldots \longrightarrow A_{n+1} \xrightarrow{\partial_{n+1}} A_n \xrightarrow{\partial_n} A_{n-1} \longrightarrow \ldots$$

over some ring together with *boundary maps* $\{\partial_n\}_{n\in\mathbb{Z}}$ which satisfy $\partial_{n-1} \circ \partial_n = 0$ for all n. Elements $a \in A_n$ are said to have *degree* $\deg(a) = n$.

Definition 2.12 (Homology of a chain complex). For a chain complex $(A_\bullet, \partial_\bullet)$, we define its *homology* as

$$H_n(A_\bullet) := \ker(\partial_n) \big/ \operatorname{im}(\partial_{n+1})$$

for all $n \in \mathbb{Z}$. We note that elements in $\ker(\partial)$ are called *cycles* and elements in $\operatorname{im}(\partial)$ are called *boundaries* of the chain complex.

Remark 2.13. Simplicial and singular chain complexes are chain complexes in the sense of Definition 2.11 and their homology coincides with the homology of chain complexes:

$$H_n(X) = H_n(C_\bullet(X)) \qquad H_n(\mathcal{K}) = H_n(C_\bullet(\mathcal{K}))$$

To decide whether the homology modules of two chain complexes are equal we have to find maps between them whose composition is the identity. In the following, we explain how to obtain induced maps on homology by maps on chain complexes and how to decide whether these induced maps coincide.

Definition 2.14 (Chain map). A *chain map* $f_\bullet : A_\bullet \to B_\bullet$ between two chain complexes $(A_\bullet, \partial^A_\bullet)$ and $(B_\bullet, \partial^B_\bullet)$ is a sequence of morphisms $f_n : A_n \to B_n$ with the property $f_n \circ \partial^A_{n+1} = \partial^B_n \circ f_n$ for all $n \in \mathbb{Z}$.

A chain map sends kernels to kernels and images to images of the boundary maps. Hence, we obtain an induced map on homology

$$(f_n)_* : H_n(A_\bullet) \longrightarrow H_n(B_\bullet)$$

for all $n \in \mathbb{Z}$.

Definition 2.15 (Chain homotopy). Let $f_\bullet, g_\bullet : A_\bullet \to B_\bullet$ be chain maps.

(i) We call the sequence $\{D_n\}_{n\in\mathbb{Z}}$ consisting of morphisms $D_n : A_n \to B_{n+1}$ a *chain homotopy* from $f_\bullet$ to $g_\bullet$ if and only if $D_{n-1}\partial^A_n + \partial^B_{n+1} D_n = f_n - g_n$ for all n.

(ii) The chain maps $f_\bullet$ and $g_\bullet$ are said to be *chain homotopic* $f_\bullet \sim g_\bullet$ if such a chain homotopy exists.

Lemma 2.16. *If $f_\bullet, g_\bullet : A_\bullet \to B_\bullet$ are chain homotopic, then their maps on homology coincide.*

Proof. The proof can be found for example in [Hat02, Proposition 2.12]. □

2.4 Generate Simplicial Complexes from Data Sets

If we have a finite data set $S = \{x^i\}_{i\in I} \subseteq \mathbb{R}^n, n \in \mathbb{Z}_{\geqslant 0}$ and want to identify holes, we need to connect the points in some way. There are two popular approaches for this, which can be found for example in [EH10, Section III.2] or [Car09, Section 2.2].

Definition 2.17 (Čech complex). We construct the *Čech complex* $\check{\mathrm{C}}(S, r)$ for a set of points $S \subseteq \mathbb{R}^n$ and a prescribed radius $r \geqslant 0$ as a set of abstract simplices: Each finite subset $J \subseteq S$ forms an abstract simplex of the Čech complex if and only if

$$\bigcap_{j\in J} B_r(j) \neq \varnothing$$

for the closed balls $B_r(j)$, or equivalently if there exist some $x \in \mathbb{R}^n$ such that

$$\|x - j\|_2 \leqslant r \tag{2.1}$$

for all $j \in J$.

Definition 2.18 (Vietoris-Rips complex). For a set of points $S \subseteq \mathbb{R}^n$ and a given radius r we denote by $\mathrm{VR}(S, r)$ the *Vietoris-Rips complex*. It is an abstract simplicial complex, where each finite subset $J \subseteq S$ defines a simplex if and only if the closed balls of radius $r \geqslant 0$ and center $j \in J$ intersect pairwise, or equivalently if

$$\|j - j'\|_2 \leqslant 2r \tag{2.2}$$

holds for all $j, j' \in J$.

For both complexes we define a corresponding boundary map as the standard boundary map for simplicial chain complexes as in Definition 2.6.

Remark 2.19. Since the properties (2.1) and (2.2) are still satisfied for all subsets of J, the faces of each simplex are again in the complex. Hence, the Čech complex and Vietoris-Rips complex are well-defined abstract simplicial complexes.

As we will see in the following example, the Čech complex and the Vietoris-Rips complex can differ.

Example 2.20. Let $S = \{a, b, c\} \subset \mathbb{R}^2$ be a set containing three points of pairwise distance 1, for example the points $a = (0, 0)$, $b = (1, 0)$ and $c = (\frac{1}{2}, \frac{\sqrt{3}}{2})$.

For the construction of the Čech and Vietoris-Rips complex, intersection of balls around the points are essential. The center of the points is $M = (\frac{1}{2}, \frac{1}{2\sqrt{3}})$ and each of the points has distance $\frac{1}{\sqrt{3}}$ to M. Hence, the balls intersect pairwise if their radius is at least $\frac{1}{2}$ and all three balls intersect for a radius of $\frac{1}{\sqrt{3}}$ or higher. For $r_0 \in [0, \frac{1}{2})$, $r_1 \in [\frac{1}{2}, \frac{1}{\sqrt{3}})$ and $r_2 \in [\frac{1}{\sqrt{3}}, \infty)$ we obtain the following complexes:

$$\begin{aligned}
\check{\mathrm{C}}(X, r_0) &= \{\{a\}, \{b\}, \{c\}\} \\
\mathrm{VR}(X, r_0) &= \{\{a\}, \{b\}, \{c\}\} \\
\check{\mathrm{C}}(X, r_1) &= \{\{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}\} \\
\mathrm{VR}(X, r_1) &= \{\{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\} \\
\check{\mathrm{C}}(X, r_2) &= \{\{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\} \\
\mathrm{VR}(X, r_2) &= \{\{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}
\end{aligned}$$

This can be seen by visualizing the distances with balls as in Figure 2.2.

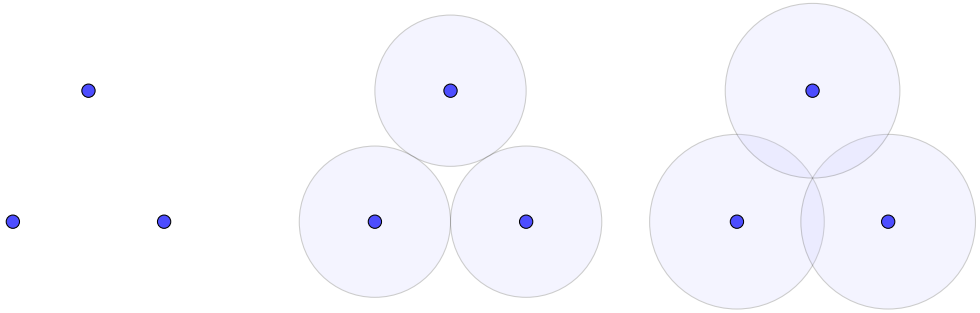


Figure 2.2: Balls at points of S with radii 0, $\frac{1}{2}$ and $\frac{1}{\sqrt{3}}$.

Even though the complexes are different, we still have a relation between them.

Proposition 2.21. *For the Čech complex and the Vietoris-Rips complex the inclusions*

$$\check{\mathrm{C}}(S,r) \subseteq \mathrm{VR}(S,r) \subseteq \check{\mathrm{C}}(S,2r)$$

hold for all $r \geqslant 0$.

Proof. Let $J \subseteq S$ be a simplex in $\check{\mathrm{C}}(S,r)$. By definition, there is some $x \in \mathbb{R}^n$ such that $\|j-x\|_2 \leqslant r$ for all $j \in J$. By using the triangle inequality we obtain

$$\|j-j'\|_2 \leqslant \|j-x\|_2 + \|x-j'\|_2 \leqslant 2r$$

for all $j, j' \in J$, which implies that J is also a simplex in $\mathrm{VR}(S,r)$. For a simplex $I \in \mathrm{VR}(S,r)$ we obtain

$$\|i-x\|_2 \leqslant 2r$$

for all $i \in I$ by setting x to be any of the points in I. Therefore I is also a simplex in the complex $\check{\mathrm{C}}(S,2r)$. □

The Vietoris-Rips complex is much easier to compute and therefore more suitable for our algorithm. Anyway, we will also refer to the Čech complex in Section 4.6.

Later on, we will work with *filtrations* of simplicial complexes, which are just sequences of inclusions of those complexes. The Čech complex and the Vietoris-Rips complex yield such filtrations.

Lemma 2.22. *We have inclusions* $\check{\mathrm{C}}(S,r) \subseteq \check{\mathrm{C}}(S,r')$ *and* $\mathrm{VR}(S,r) \subseteq \mathrm{VR}(S,r')$ *for all radii* $0 \leqslant r \leqslant r'$.

Proof. This follows directly from the properties of the complexes in Definition 2.17 and Definition 2.18. □

The complexes can only change for finitely many radii $0 = r_0 < r_1 < r_2 < \cdots < r_n$ since there are only finitely many points S to construct abstract simplices. For the Vietoris-Rips complex we obtain a filtration

$$\varnothing \hookrightarrow \mathrm{VR}(S,0) \hookrightarrow \mathrm{VR}(S,r_1) \hookrightarrow \ldots \hookrightarrow \mathrm{VR}(S,r_n) \tag{2.3}$$

and analogously, we obtain one for the Čech complexes.

These inclusions yield maps on homology since they are chain maps in the sense of Definition 2.14. In the proceeding chapter we will study how to track the change in homology during these filtrations.

3 Persistent Homology

If we have a filtration as in (2.3), then its inclusions are chain maps and induce maps on homology as in DEFINITION 2.14. *Persistent homology* is a tool to track the change in homology during this filtration. In this chapter we mainly follow [ELZ00, Section 3] and [EH10, Section VII.1].

3.1 What Is Persistent Homology?

If we have an inclusion $X \subseteq Y$ of simplicial complexes, then this yields an inclusion of chain complexes $\iota : C_\bullet(X) \to C_\bullet(Y)$, which induces a map on homology

$$\iota_* : H_k(X) \longrightarrow H_k(Y)$$

for all $k \in \mathbb{Z}$. The homology classes that persist under the map ι_* can be specified as the image $\mathrm{im}(\iota_*)$.

DEFINITION 3.1 (Persistent homology). We call $\mathrm{im}(\iota_*) \subseteq H_k(Y)$ the *k-th persistent homology* of the inclusion $\iota : X \to Y$.

It is easy to provide a basis of the persistent homology. We take a basis of $H_k(X)$ and apply ι_* to each of the basis elements to obtain a list of generators for $\mathrm{im}(\iota_*)$. Then we create a basis by removing those elements that make the list linearly dependent.

Now, we consider a filtration of simplicial complexes

$$\varnothing \hookrightarrow X_0 \xhookrightarrow{\iota^0} X_1 \xhookrightarrow{\iota^1} \ldots \xhookrightarrow{\iota^{N-1}} X_N \tag{3.1}$$

for $N \in \mathbb{Z}_{\geqslant 1}$, which is a sequence of simplicial complexes connected by inclusions. We obtain a sequence in homology

$$0 \longrightarrow H_k(X_0) \xrightarrow{\iota^0_*} H_k(X_1) \xrightarrow{\iota^1_*} \ldots \xrightarrow{\iota^{N-1}_*} H_k(X_N) \tag{3.2}$$

for any $k \in \mathbb{Z}$ and can define persistent homology in a similar way:

DEFINITION 3.2 (Persistent homology of a filtration). We consider that we have a filtration as in (3.1). Let $i, j \in \mathbb{Z}$ be integers with $0 \leqslant i < j \leqslant N$. For all $k \in \mathbb{Z}$ we define the *k-th persistent homology* from i to j as

$$H_k^{i,j} := \mathrm{im}(\iota_*^{j-1} \circ \cdots \circ \iota_*^i) \subseteq H_k(X_j).$$

We want to find bases $\mathcal{B}_{H_k(X_i)}$ for every homology $H_k(X_i)$ in the sequence, such that for all $b \in \mathcal{B}_{H_k(X_i)}$ one of the following properties is satisfied:

- $\iota_*^i(b) = 0$
- there exists $\widetilde{b} \in \mathcal{B}_{H_k(X_{i+1})}$ such that $\iota_*^i(b) = \widetilde{b}$

If we have such bases, it is easy to state a basis for the persistent homology

$$\mathcal{B}_{H_k^{i,j}} = \left\{ b \in \mathcal{B}_{H_k(X_j)} \;\middle|\; \exists b' \in \mathcal{B}_{H_k(X_i)} : \iota_*^{j-1} \circ \cdots \circ \iota_*^i(b') = b \right\}$$

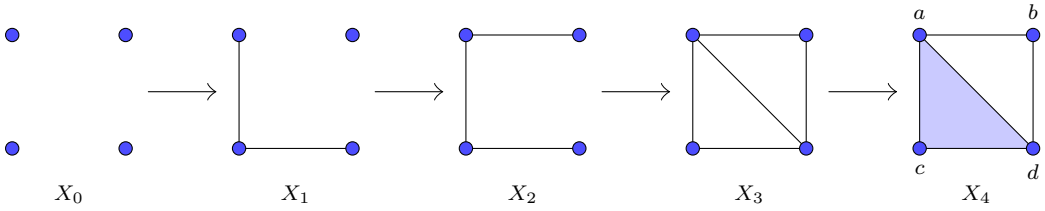
and this enables us to track the basis elements in the sequence (3.2).

Furthermore, if we assume that the map $\iota_*^l : \mathcal{B}_{H_k(X_l)} \to \mathcal{B}_{H_k(X_{l+1})}$ is injective for all $l \in \{0, \dots, N-1\}$, then for each basis element $b \in \mathcal{B}_{H_k(X_{j_b})}$ with $j_b \in \{0, \dots, N\}$ and with $\iota_*^{j_b}(b) = 0$ if $j_b \neq N$ we find a unique

$$\mathcal{B}_{H_k^{i_b, j_b}} \ni b$$

such that $j_b - i_b$ is maximal. This describes the fact that b is created at $H_k(X_{i_b})$ and destroyed at $H_k(X_{j_b+1})$ if $j_b \neq N$. The lifespans can be visualized in the form of *barcodes*, where we draw one interval $[i_b, j_b + 1)$ for each basis element b with $\iota_*^{j_b} b = 0$ and $[i_b, N_b]$ extended by a red line at the end for each basis element b in $\mathcal{B}_{H_k(X_N)}$.

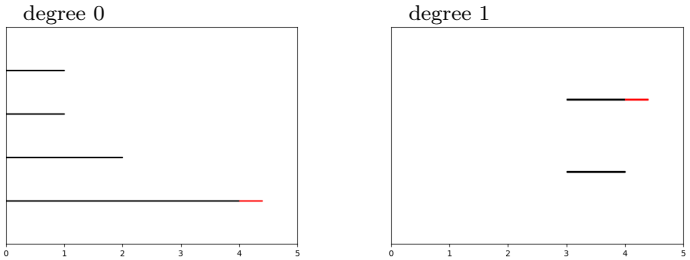
Example 3.3. For the filtration



we can find the following bases:

$$\begin{aligned}
\mathcal{B}_{H_0(X_0)} &= \{a, b-a, c-a, d-a\} \\
\mathcal{B}_{H_0(X_1)} &= \{a, b-a\} \\
\mathcal{B}_{H_0(X_2)} &= \{a\} = \mathcal{B}_{H_0(X_3)} = \mathcal{B}_{H_0(X_4)} \\
\mathcal{B}_{H_1(X_3)} &= \{ab + bd - ad, ac + cd - ad\} \\
\mathcal{B}_{H_1(X_4)} &= \{ab + bd - ad\}
\end{aligned}$$

Therefore we can assign the barcodes



to the filtration.

It is neither obvious if it is possible to create a basis with the properties needed to track the ways of the basis elements, nor if the lifetimes obtained by the choice of basis are unique in any sense.

In the next section we discuss the concept of barcodes. We will see that, under certain assumptions, we are always able to draw barcodes as above and that these barcodes are unique up to reordering of the bars.

3.2 Barcodes

If we have a sequence of homology modules, then we are interested in how the sequence changes at which steps. In this section we define *barcodes*, which describe this in a unique way. They represent bases of persisting homology classes through intervals. We want to draw barcodes for the homology of filtrations like those in Equation (2.3) obtained from data sets.

We consider a filtration of simplicial complexes, where at each step new simplices are added to the complex. Before we start tracking the holes through homology, at first we want to check whether this is even a reasonable idea. We will show that it is possible to obtain lifetimes and that they are unique up to reordering like indicated in the last section, if we consider the homology over fields.

We start with a filtration

$$\varnothing \hookrightarrow X_0 \xhookrightarrow{\iota_0} X_1 \xhookrightarrow{\iota_1} X_2 \xhookrightarrow{\iota_2} \dots \xhookrightarrow{\iota_{n-1}} X_n = X.$$

By computing homology for an arbitrary ring R, we obtain a sequence of R-modules

$$0 \longrightarrow H_k(X_0) \xrightarrow{(\iota_0)_*} H_k(X_1) \xrightarrow{(\iota_1)_*} \dots \xrightarrow{(\iota_{n-1})_*} H_k(X_n) = H_k(X). \tag{3.3}$$

If we consider R to be a field, we get a sequence of vector spaces, which we call a *directed space*. Since we assume that these vector spaces are finite dimensional, we can decompose this directed space into a direct sum of very simple directed spaces:

Theorem 3.4. *Every directed space* $(V, f) = \left(0 \to V_0 \xrightarrow{f_0} V_1 \xrightarrow{f_1} \dots \xrightarrow{f_{n-1}} V_n \xrightarrow{f_n} 0\right)$ *of finite dimensional vector spaces* V_i *over a field* $\mathbb{F}$ *is isomorphic*[1] *to a direct sum*[2] *of intervals*

$$V \cong \bigoplus_{i=0}^{s} \mathbb{F}[a_i, b_i]$$

with $a_i, b_i \in \{0, \dots, n\}$, $a_i \leqslant b_i$ *and* $s \in \mathbb{Z}_{\geqslant 0}$, *where*

$$\mathbb{F}[a_i, b_i] = \left(0 \to \underset{0}{0} \to \dots \to 0 \to \underset{a_i}{\mathbb{F}} \xrightarrow{\mathsf{id}} \mathbb{F} \xrightarrow{\mathsf{id}} \mathbb{F} \xrightarrow{\mathsf{id}} \dots \xrightarrow{\mathsf{id}} \underset{b_i}{\mathbb{F}} \to 0 \to \dots \to \underset{n}{0} \to 0\right).$$

The numbers underneath the vector spaces denote the positions with respect to V.

[1] An isomorphisms of directed spaces (V, f) and (W, g) is a sequence of isomorphisms $\phi_i : V_i \to W_i$ such that $\phi_{i+1} \circ f_i = g_i \circ \phi_i$ for all i.

[2] Taking the direct sum of directed spaces means taking the direct sum at each level of the sequence.

Proof. The authors Zomorodian and Wang prove this theorem in [ZC05], [ZC08] and [Wan12] by considering the directed space as a graded module

$$M = V_0 \oplus \cdots \oplus V_n$$

over the ring $\mathbb{F}[t]$ with $t \cdot (v_0, \ldots, v_n) := (0, f_0(v_0), \ldots, f_{n-1}(v_{n-1}))$ and using some structure theorem for graded modules. We want to follow a more elementary approach, tailor-made for the problem. Our strategy is to find and split off summands of the form $\mathbb{F}[a_i, b_i]$ inductively. This yields the required decomposition, since V only consists of finitely many finite dimensional vector spaces.

Without loss of generality, we can assume that $V_0 \neq 0$. Otherwise we would execute the same procedure but with an index shift. Let i be the minimal index with

$$\ker(f_i \circ \cdots \circ f_0) \neq 0.$$

This condition has to be fulfilled for some i since $\ker(f_n \circ \cdots \circ f_0) = V_0 \neq 0$ by assumption. For $W := \ker(f_i \circ \cdots \circ f_0)$ we obtain the commutative diagram

$$\begin{array}{ccccccccccc}
W & \xrightarrow[\sim]{f_0|_W} & f_0(W) & \xrightarrow[\sim]{f_1|_{f_0(W)}} & \cdots & \xrightarrow[\sim]{f_{i-1}|_{\ldots}} & f_{i-1} \circ \ldots \circ f_0(W) & \xrightarrow{f_i|_{\ldots}} & f_i \circ \ldots \circ f_0(W) = 0 & & \\
\downarrow & & \downarrow & & & & \downarrow & & \downarrow & & \\
V_0 & \xrightarrow{f_0} & V_1 & \xrightarrow{f_1} & \cdots & \xrightarrow{f_{i-1}} & V_i & \xrightarrow{\qquad\qquad f_i \qquad\qquad} & V_{i+1} & \xrightarrow{f_{i+1}} & \cdots,
\end{array}$$

where the maps in the upper row, except the last one, are isomorphisms of vector spaces since $f_0, \ldots, (f_{i-1} \circ \ldots \circ f_0)$ are injective by the definition of i. We can prove that

$$\widetilde{W} = (\ 0 \longrightarrow \underbrace{W}_{=:\widetilde{W}_0} \longrightarrow \ldots \longrightarrow \underbrace{f_{i-1} \circ \ldots \circ f_0(W)}_{=:\widetilde{W}_i} \longrightarrow 0\ )$$

can be decomposed into a direct sum of intervals. To do this we choose any basis $b_1^0, \ldots, b_n^0$ of $\widetilde{W}_0$. The image $b_1^j = (f_{j-1} \circ \ldots \circ f_0)(b_1^0), \ldots, b_n^j = (f_{j-1} \circ \ldots \circ f_0)(b_n^0)$ of the basis under the first $j \leqslant i$ maps is a basis of $\widetilde{W}_j$. We use this to decompose $\widetilde{W}$:

$$\begin{aligned}
\widetilde{W} &= \bigoplus_{l=1}^{n} \Big(\ 0 \longrightarrow \mathbb{F}\, b_l^0 \longrightarrow \ldots \longrightarrow \mathbb{F}\, b_l^i \longrightarrow 0\ \Big) \\
&= \bigoplus_{l=1}^{n} \mathbb{F}[0, i]
\end{aligned}$$

We want to split off $\widetilde{W}$ from V by finding $\widetilde{V} = \Big(0 \to \widetilde{V}_0 \to \cdots \to \widetilde{V}_n \to 0\Big)$ a directed space with

$$\widetilde{W} \oplus \widetilde{V} = V.$$

By extending the basis of $\widetilde{W}_0$ to a basis of V_0 we can find a vector space $\widetilde{V}_0$, which is the linear combination of the added basis elements. It holds

$$V_0 = \widetilde{W}_0 \oplus \widetilde{V}_0.$$

For all $j \leqslant i$, we define $\widetilde{V}_j$ inductively in the following way. Assume that V_{j-1} is of the form $\widetilde{W}_{j-1} \oplus \widetilde{V}_{j-1}$ for some $\widetilde{V}_{j-1}$. Then the sum of $\widetilde{W}_j$ and $f_{j-1}(\widetilde{V}_{j-1})$ is a direct sum since f_{j-1} is injective for all $j \leqslant i$. By extending the basis of $\widetilde{W}_j \oplus f_{j-1}(\widetilde{V}_{j-1})$, we obtain V'_j with

$$V_j = \widetilde{W}_j \oplus \underbrace{f_{j-1}(\widetilde{V}_{j-1}) \oplus V'_j}_{=:\widetilde{V}_j}.$$

For $i < j \leqslant n$, we define $\widetilde{V}_j := V_j$. The maps of the sequence

$$0 \longrightarrow \widetilde{V}_0 \xrightarrow{f_0|_{\widetilde{V}_0}} \widetilde{V}_1 \xrightarrow{f_1|_{\widetilde{V}_1}} \dots \xrightarrow{f_{n-1}|_{\widetilde{V}_{n-1}}} \widetilde{V}_n \longrightarrow 0$$

are well-defined morphisms since each $f_i|_{\widetilde{V}_i}$ has its image in $\widetilde{V}_{i+1}$ by definition. Therefore, the sequence is a directed space $(\widetilde{V}, f|_{\widetilde{V}})$.

In the following, we will see that the directed spaces (V, f) and $(\widetilde{W} \oplus \widetilde{V}, f|_{\widetilde{W}} \oplus f|_{\widetilde{V}})$ coincide. Since the vector spaces in each degree are the same, it remains to show that the maps of both sequences are equal, i.e. that the diagram

$$\begin{array}{ccc} V_j & \xrightarrow{\quad f_j \quad} & V_{j+1} \\ \wr\downarrow \mathsf{id} & & \wr\downarrow \mathsf{id} \\ \widetilde{W}_j \oplus \widetilde{V}_j & \xrightarrow{f_j|_{\widetilde{W}_j} \oplus f_j|_{\widetilde{V}_j}} & \widetilde{W}_{j+1} \oplus \widetilde{V}_{j+1} \end{array}$$

commutes. For $j \leqslant i$, we have by the linearity of f_j that

$$f_j(v) = f_j(\widetilde{w}) + f_j(\widetilde{v})$$

for $v = \widetilde{w} + \widetilde{v} \in V_j = \widetilde{W}_j \oplus \widetilde{V}_j$. Furthermore, the diagram

$$\begin{array}{ccc} V_j & \xrightarrow{\quad f_j \quad} & V_{i+1} \\ \wr\downarrow & & \wr\downarrow \\ W_j \oplus \widetilde{V}_j \\ = 0 \oplus V_j & \xrightarrow{0 \oplus f_j} & W_{j+1} \oplus \widetilde{V}_{j+1} \\ & & = 0 \oplus V_{j+1} \end{array}$$

commutes, too.

If $\widetilde{W}$ already equals the whole directed space V, we found a decomposition as required. Otherwise we have restricted the proof to a case where $\widetilde{V}$ has lower dimension. By induction, this finishes the proof. □

Proposition 3.5. *The direct sum decomposition from* Theorem *3.4 is unique up to reordering of the summands.*

Proof. Let $V = (V, f) = \left(0 \to V_0 \xrightarrow{f_0} V_1 \xrightarrow{f_1} \dots \xrightarrow{f_{n-1}} V_n \xrightarrow{f_n} 0\right)$ be a directed space as in Theorem 3.4. We assume that there are two different decompositions

$$\bigoplus_{i=0}^{s_1} \mathbb{F}[a_i, b_i] \cong V \cong \bigoplus_{j=0}^{s_2} \mathbb{F}[c_j, d_j] \tag{3.4}$$

with $s_1, s_2 \in \mathbb{Z}_{\geqslant 0}$, $a_i \leqslant b_i \in \{0, \dots, n\}$ for all $i \in \{0, \dots, s_1\}$ and $c_j \leqslant d_j \in \{0, \dots, n\}$ for all $j \in \{0, \dots, s_2\}$. Here, $\mathbb{F}[a,b] = (\mathbb{F}[a,b], f^{[a,b]})$ with

$$\mathbb{F}[a,b]_k = \begin{cases} \mathbb{F} & , k \in \{a, \dots, b\} \\ 0 & , \text{else} \end{cases} \quad \text{and} \quad f_k^{[a,b]} = \begin{cases} \mathsf{id} & , k \in \{a, \dots, b-1\} \\ 0 & , \text{else} \end{cases}$$

is the directed space known from THEOREM 3.4 for $a \leqslant b \in \{0, \dots, n\}$.

Then for $k \in \{0, \dots, n\}$ we have

$$\dim(\ker(f_k)) = \dim\left(\ker\left(\bigoplus_{i=0}^{s_1} f_k^{[a_i,b_i]}\right)\right) = |\{i \mid b_i = k\}|$$

and $\dim(\ker(f_k)) = |\{j \mid d_j = k\}|$. Furthermore, for $l \leqslant k \in \{0, \dots, n\}$ we obtain

$$\begin{aligned} &\dim\left(\left\{v \in V_k \;\middle|\; \begin{array}{l} v \in \ker(f_k) \\ v \in \operatorname{im}(f_{k-1} \circ \cdots \circ f_l) \end{array}\right\}\right) \\ &= \dim\left(\left\{v \in \bigoplus_{i=0}^{s_1} \mathbb{F}[a_i,b_i] \;\middle|\; \begin{array}{l} v \in \ker(\bigoplus_{i=0}^{s_1} f_k^{[a_i,b_i]}) \\ v \in \operatorname{im}(\bigoplus_{i=0}^{s_1} f_{k-1}^{[a_i,b_i]} \circ \cdots \circ \bigoplus_{i=0}^{s_1} f_l^{[a_i,b_i]}) \end{array}\right\}\right) \\ &= |\{i \mid b_i = k, a_i \leqslant l\}| \end{aligned}$$

and an analog statement for the other direct sum. By using

$$|\{i \mid b_i = k, a_i \leqslant l\}| - |\{i \mid b_i = k, a_i \leqslant l-1\}| = |\{i \mid b_i = k, a_i = l\}|$$

if $l \geqslant 1$ and by doing the same argument for the other direct sum we conclude

$$|\{i \mid b_i = k, a_i = l\}| = |\{j \mid d_j = k, c_j = l\}|.$$

Hence, both direct sums are equal up to reordering of the summands. □

Now we can define barcodes as in [ZC08, Section 3.5], [CZCG05, Section 5.3] or [Ghr08, Section 2.3].

DEFINITION 3.6 (Barcode). Let V be a directed space of finite dimensional vector spaces over a field $\mathbb{F}$. By THEOREM 3.4, we have $V = \bigoplus_{i=0}^{s} \mathbb{F}[a_i, b_i]$. The *barcode* for V is defined as the tuple of intervals

$$([a_i, b_i])_{i=0,\dots,s}.$$

By PROPOSITION 3.5, the barcode is unique up to reordering.

For each sequence of homology modules with field coefficients $\mathbb{F}$ as in (3.3) we obtain a unique barcode if we extend the sequence by the map $H_k(X) \to 0$.

We note that the construction and definition of a barcode work for directed spaces with arbitrary indices $0 < r_1 < \cdots < r_n$ instead of $0 < 1 < \cdots < n$. Moreover, we note that for an interval $[r_i, r_j]$ with $j < n$ in the barcode, we draw $[r_i, r_{j+1})$ as in EXAMPLE 3.3 since this describes our conception that the basis element is destroyed at the r_{j+1}-th level. For an interval $[r_i, r_n]$ we draw $[r_i, r_n]$ and extend this interval by a red line at the end to indicate the difference to $[r_i, r_n)$.

4 Computing Persistent Homology

In the preceding chapter we defined persistent homology of a sequence of simplicial complexes to describe the change of homology through the sequence. We learned that if we choose the coefficients for chains and homology to be a filed $\mathbb{F}$, we can describe which elements in homology are created and destroyed through this process by drawing barcodes. Now we want to deal with the computational aspect. We will state an algorithm that takes a filtration of finite simplicial complexes as input and returns a barcode together with a corresponding basis element for each interval.

The algorithm is a replication from [ZC08]. The proof is inspired by the paper [ZC05] but for quite a few points we chose to give our own arguments. It also provides us with the idea of the next section.

4.1 General Idea

At first we assume that we have a filtration

$$\varnothing = X_{-1} \hookrightarrow X_0 \hookrightarrow X_1 \hookrightarrow \ldots \hookrightarrow X_N = X \tag{4.1}$$

of a finite simplicial complex X where at each inclusion $X_{i-1} \hookrightarrow X_i$ one simplex $\sigma_i \notin X_{i-1}$ is added to the complex:

$$X_i = X_{i-1} \cup \{\sigma_i\}$$

We choose a field $\mathbb{F}$ as coefficients for the chain groups and homology. In the following, we will discuss how to find a basis

$$\mathcal{B}_l^N = \{b_1^l, \ldots, b_{m_l}^l, z_1^l, \ldots, z_{n_l}^l\}$$

of $C_l(X)$ for all $l \in \mathbb{Z}_{\geqslant 0}$, such that every basis element is either a cycle or maps to another basis element by the boundary map:

$$\begin{aligned} \partial_l(z_i^l) &= 0 \\ \partial_l(b_i^l) &= z_i^{l-1} \end{aligned} \tag{4.2}$$

The non-vanishing boundaries

$$\partial_l b_1^l, \ldots, \partial_l b_{m_l}^l$$

are linearly independent, since they map to different basis elements in $\mathcal{B}_{l-1}^N$. They form a basis of $\mathrm{im}(\partial_l)$. Those elements in $\mathcal{B}_l^N$ with vanishing boundary

$$z_1^l, \ldots, z_{n_l}^l$$

are a basis of the kernel $\ker(\partial_l)$ of the boundary map. The basis $\mathcal{B}_l^N$ enables us to write the homology of X as

$$\begin{aligned} H_l(X) = {}^{\ker(\partial_l)}\!/_{\mathrm{im}(\partial_{l+1})} &= \langle z_1^l, \dots, z_{n_l}^l \rangle / \langle \partial_{l+1} b_1^{l+1}, \dots, \partial_{l+1} b_{m_{l+1}}^{l+1} \rangle \\ &= \langle z_1^l, \dots, z_{n_l}^l \rangle / \langle z_1^l, \dots, z_{m_{l+1}}^l \rangle \\ &= \langle [z_{m_{l+1}+1}^l], \dots, [z_{n_l}^l] \rangle \end{aligned}$$

and obtain a basis $\{[z_{m_{l+1}+1}^l], \dots, [z_{n_l}^l]\}$ of $H_l(X)$ for all $l \in \mathbb{Z}_{\geqslant 0}$. In this setting $\langle \cdot \rangle$ denotes the span and $[\cdot]$ are equivalence classes of the quotient. The algorithm that we aim at yields bases $\mathcal{B}_l^N, l \in \mathbb{Z}_{\geqslant 0}$, which can be restricted to bases $\mathcal{B}_l^i \subseteq \mathcal{B}_l^N$ of $C_l(X_i)$ for all $0 \leqslant i \leqslant N$. This allows us to track the basis elements and obtain barcodes. We will discuss this further in Section 4.3.

Remark 4.1. In the algorithm we will not order the basis elements such that the first m_l elements of $z_1^{l-1}, \dots, z_{n_{l-1}}^{l-1}$ are in the image of the boundary map ∂_l. We did it here for simplicity.

To achieve this representation of the basis, we use an inductive approach. Let i be in $\{0, \dots, N-1\}$. We assume that the bases $\mathcal{B}_l^i = \{b_1^l, \dots, b_{m_l}^l, z_1^l, \dots, z_{n_l}^l\}$ of $C_l(X_i)$ for all $l \in \mathbb{Z}_{\geqslant 0}$ are already of the desired form. To obtain bases of all $C_l(X_{i+1}), l \in \mathbb{Z}_{\geqslant 0}$ we just have to add one basis element: If we add a k-simplex σ_{i+1} at the inclusion $X_i \hookrightarrow X_{i+1}$, then

$$\mathcal{B}_k^{i+1} := \mathcal{B}_k^i \cup \{\sigma_{i+1}\}$$

is a basis of of $C_k(X_{i+1})$. For all $l \neq k \in \mathbb{Z}_{\geqslant 0}$ we can adopt the basis $\mathcal{B}_l^{i+1} := \mathcal{B}_l^i$ of $C_l(X_i)$ to obtain a basis of $C_l(X_{i+1})$ since both chain groups coincide.

To modify the bases $\mathcal{B}_l^{i+1}, l \in \mathbb{Z}_{\geqslant 0}$ such that they are also in the desired form we perform two steps. In Step 1 we modify the newly added basis element σ_{i+1} by adding a linear combination of the other basis elements, such that either $\partial_k \sigma_{i+1} = 0$ or we detect that the basis elements $\partial_k b_1^k, \dots, \partial_k b_{m_k}^k, \partial_k \sigma_{i+1}$ are linearly independent. In the latter case we use Step 2 to modify one of the basis elements $z_{m_k+1}^{k-1}, \dots, z_{n_{k-1}}^{k-1}$, such that it equals $\partial_k \sigma_{i+1}$.

Remark 4.2. The two steps described above can be realized by using elementary transformations of the rows and columns of a matrix as in [ZC05]. For this purpose we consider the transformation matrix

$$M_{\mathcal{B}_{k-1}^{i+1}}^{\mathcal{B}_k^{i+1}}(\partial_k)$$

of the boundary map ∂_k for the bases as above. Each element in $\mathcal{B}_k^{i+1}$ represents a column and each element in $\mathcal{B}_k^i$ represents a row of the matrix. Since $\mathcal{B}_l^i, l \in \mathbb{Z}_{\geqslant 0}$ is of the desired form (4.2), the columns represented by $z_1^k, \dots, z_{m_k}^k$ have just 0-entries and the columns represented by $b_1^k, \dots, b_{n_k}^k$ each have one entry which is not 0.

In the first step we use Gaussian elimination for the column represented by σ_{i+1}, until all entries of the column are 0 or we obtain a pivot element which we can not

eliminate by Gaussian elimination. In this case we multiply the row with a factor such that the pivot element is 1:

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & \vdots & \vdots & 0 & \vdots \\ 0 & \vdots & \vdots & \vdots & 0 \\ 0 & 0 & 0 & 0 & 1 \\ \vdots & 1 & \vdots & \vdots & * \\ \vdots & 0 & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & * \end{pmatrix}$$

Afterwards, in step 2 we utilize the Gaussian elimination for the rows of the matrix to generate zeros below the pivot:

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & \vdots & \vdots & 0 & \vdots \\ 0 & \vdots & \vdots & \vdots & 0 \\ \vdots & 0 & \vdots & \vdots & 1 \\ \vdots & 1 & \vdots & \vdots & 0 \\ \vdots & 0 & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

This idea is essential for the algorithm which we will introduce in the next section.

4.2 The Algorithm

First of all, we discuss how to implement the idea from last section into an algorithm. Then we will prove that it indeed gives us the desired basis representation.

Although one can formulate the algorithm for chains with coefficients in any field, we state it for $\mathbb{F}_2$ since this makes it easier to implement the algorithm: There is no need to take care of the sign, especially when we use the boundary map, since each simplex has just one orientation. Furthermore, we can implement simplicial chains as sets of basis elements and their addition and subtraction can easily be achieved by joining the corresponding sets and removing their intersection. Another advantage is that we do not have to pay attention to the coefficients for the Gaussian elimination but just add the basis elements.

We consider filtration (4.1). For a simplex $\sigma \in X_m$ with $X_m = X_{m-1} \cup \{\sigma\}$ for some $m \in \{0, \dots, N\}$ we define its *index* to be $\text{index}(\sigma) := m$. In a list K we store all those simplices ordered by their index. We say $\tau \in K$ has a *lower index* than $\sigma \in K$ if $\text{index}(\tau) < \text{index}(\sigma)$. Equivalently, we say that σ has a *higher index* than τ.

Remark 4.3. We note that the boundary $\partial\sigma$ of each simplex $\sigma \in K$ is a linear combination of simplices $\tau \in K$ with a lower index than σ since each face of σ is in the simplicial complex $X_{\text{index}(\sigma)}$ and the $(\dim(\sigma) - 1)$-simplices in $X_{\text{index}(\sigma)}$ and $X_{\text{index}(\sigma)-1}$ coincide.

To each simplex $\sigma \in K$ we assign two new variables basisel(σ) and partner(σ). The variable basisel(σ) is a simplicial chain which is a sum of σ and a linear combination of simplices in K which have a lower index than σ:

$$\text{basisel}(\sigma) = \sigma + \sum_{\text{index}(\eta)<\text{index}(\sigma)} \lambda_\eta \cdot \eta \quad , \lambda_\eta \in \mathbb{F}_2 \tag{4.3}$$

Because of their form, the set

$$\mathcal{B}_k^m := \{\text{basisel}(\sigma) \mid \sigma \in X_m \ k\text{-simplex}\}$$

is a basis of $C_k(X_m)$ for all $m \in \{0, \dots, N\}$ and $k \in \mathbb{Z}_{\geqslant 0}$ at all times, which we will prove in Remark 4.7 (3). At the beginning we define basisel(σ) := σ for each $\sigma \in K$ but we will modify these values during the algorithm. We will use a for-loop over the simplices in K such that after iteration $\sigma \in K$ with index(σ) = m the bases $\{\mathcal{B}_k^m\}_{k \in \mathbb{Z}_{\geqslant 0}}$ are of the desired form (4.2). For this we consider the matrix

$$M_k^m := M_{\mathcal{B}_{k-1}^m}^{\mathcal{B}_k^m}(\partial_k)$$

if the simplex σ with index(σ) = m is a k-simplex and do the operations from Remark 4.2. We note that in [ZC08] Zomorodian calls the basis elements cascade(σ) for $\sigma \in K$ since by using Gaussian elimination we add chains to the basis element and it spreads over the complex like a cascade.

The variable partner(σ) for each $\sigma \in K$ is either empty or another simplex in K. It indicates relation (4.2) in the following way at the end of the algorithm:

$$\partial\, \text{basisel}(\sigma) = \text{basisel}(\eta) \iff \text{partner}(\sigma) = \eta \text{ and } \text{index}(\sigma) > \text{index}(\eta) \tag{4.4}$$

and $\partial\, \text{basisel}(\sigma) = 0$ otherwise.

For the procedure described in Remark 4.2 it is essential to find pivot elements for Gaussian elimination. In the algorithm we have to find the pivot of the column represented by basisel(σ) in M_k^m. We consider that the rows of M_k^m are ordered such that a row represented by basisel(τ) $\in \mathcal{B}_{k-1}^m$ is higher than a row represented by basisel(η) $\in \mathcal{B}_{k-1}^m$ if and only if index(τ) > index(η). If we describe $\partial\, \text{basisel}(\sigma)$ as linear combination

$$\partial\, \text{basisel}(\sigma) = \sum_{\tau \in \mathcal{B}_{k-1}^m} \lambda_\tau \cdot \text{basisel}(\tau) \quad , \lambda_\tau \in \mathbb{F}_2$$

then finding basisel(τ) such that τ is the simplex with the highest index in $\mathcal{B}_{k-1}^m$ with $\lambda_\tau \neq 0$ is the same as finding the pivot of the column represented by basisel(σ). We will show in Remark 4.7 (4) that since the basis elements are of the form (4.3), finding the pivot is equivalent to finding the simplex τ with the highest index which occurs in the chain $\partial\, \text{basisel}(\sigma) \in C_{k-1}(X_m)$. This can be done by the function

$$\begin{aligned} &\text{youngest} : C_l(X_m) - \{0\} \longrightarrow \mathcal{A}_l^m := \{l\text{-simplices in } X_m\} \subset X_m \\ &\qquad \xi = \sum_{\eta \in \mathcal{A}_l^m} \lambda_\eta \cdot \eta \longmapsto \tau \quad , \text{index}(\tau) = \max\{\text{index}(\eta) \mid \lambda_\eta \neq 0, \eta \in \mathcal{A}_l^m\} \end{aligned} \tag{4.5}$$

which is defined for all $m \in \{0, \dots, N\}$ and $l \in \mathbb{Z}_{\geqslant 0}$. We note that each function youngest : $C_l(X_m) - \{0\} \to \mathcal{A}_l^m$ is a restriction of youngest : $C_l(X_N) - \{0\} \to \mathcal{A}_l^N$.

Before the algorithm starts, we assign partners and basis elements

$$\text{partner}(\sigma) = \varnothing$$
$$\text{basisel}(\sigma) = \sigma$$

to all simplices σ in the list K. We execute the algorithm and pass the list K as parameter to it.

Algorithm 1 Persistent homology algorithm.

```
1:  def change_basis(K):
2:      for σ ∈ K:
3:          while True:.....Step 1
4:              if ∂ basisel(σ) = 0:
5:                  break
6:              else:
7:                  τ = youngest(∂ basisel(σ))
8:                  if partner(τ) = ∅:
9:                      assign_partner(τ, σ)
10:                     break
11:                 else:
12:                     basisel(σ) = basisel(σ) + basisel(partner(τ))
13:         if partner(σ) ≠ ∅: .....Step 2
14:             eliminate = basisel(partner(σ)) + ∂ basisel(σ)
15:             while eliminate ≠ 0:
16:                 τ = youngest(eliminate)
17:                 basisel(partner(σ)) = basisel(partner(σ)) + basisel(τ)
18:                 eliminate = eliminate + basisel(τ)
```

In this algorithm, the **break**-operator terminates the superordinate while-loop. In line 9 we use the function assign_partner() for two simplices τ and σ in K. It connects the two simplices by assigning them as partners to each other by Algorithm 2.

Algorithm 2 Assigning partners.

```
1: def assign_partner(τ, σ):
2:     partner(τ) = σ
3:     partner(σ) = τ
```

The first step includes lines 3 to 12 and can be visualized by Figure 4.1. In line 7 we determine the highest non-vanishing element of the column represented by basisel(σ) as described above for Gaussian elimination.

The addition in line 12 is the elementary operation for Gaussian elimination on columns. In the proof of Properties 4.6 (1) in Lemma 4.10 we will see that basisel(σ) and basisel(partner(τ)) are chains of the same degree and therefore their addition in this line is well-defined.

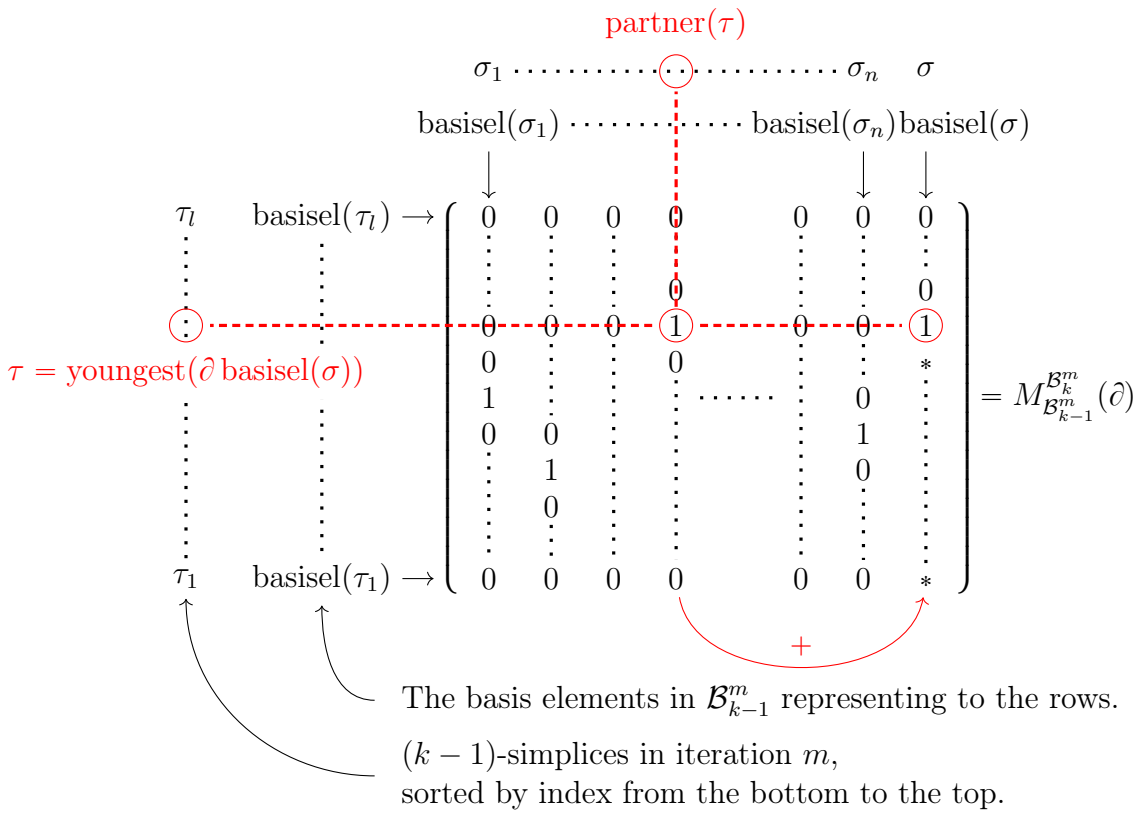


Figure 4.1: The first step of the algorithm.

There are two criterions to terminate the first step. If all entries of the column belonging to basisel(σ) in M_k^m are 0, then ∂ basisel(σ) = 0 and the basis $\mathcal{B}_k^m$ is of the form (4.2). We terminate the first step in line 5 and do not need a second step. The second criterion is partner(τ) = $\varnothing$ for τ = youngest(∂ basisel(σ)). In this case all entries in the row represented by basisel(τ) in M_k^{m-1} are 0 since the bases $\{\mathcal{B}_k^{m-1}\}_{k\in\mathbb{Z}_{\geqslant 0}}$ are of the form (4.2) which is indicated by the assignment of partners as in (4.4). But the column represented by basisel(σ) has a non-zero entry in the row represented by basisel(τ) in M_k^m. Therefore ∂ basisel(σ) is not a linear combination of the other boundaries ∂ basisel(σ_i) with basisel(σ_1), ..., basisel(σ_n) $\in \mathcal{B}_k^{m-1}$. We assign partner(σ) = τ and vice versa to mark the highest non-vanishing row represented by basisel(τ) and terminate step 1 in line 10.

If step 1 was terminated by the second criterion, we use step 2 in lines 13 to 18 to obtain

$$\text{basisel}(\text{partner}(\sigma)) = \partial\,\text{basisel}(\sigma). \tag{4.6}$$

In Remark 4.4 we show that after each iteration of the while-loop in step 2 we have

$$\textit{eliminate} = \text{basisel}(\text{partner}(\sigma)) + \partial\,\text{basisel}(\sigma)$$

which describes the entries below the row represented by basisel(partner(σ)) in the column basisel(σ). It is used to modify the value of basisel(partner(σ)). If *eliminate* is 0, then (4.6) holds and the second step is finished. Step 2 from Algorithm 1 is the procedure described in step 2 of Remark 4.2: In line 16 we use the function youngest($\cdot$)

to find the highest row in M_k^m below the row represented by basisel(partner(σ)) which is has a non-zero entry in column represented by basisel(σ). We create a zero at this entry by modifying the basis $\mathcal{B}_{k-1}^m$ in line 17 using the elementary operation from Gaussian elimination on the rows. In the proof of PROPERTIES 4.6 (2) in LEMMA 4.10 we show that the additions of chains in step 2 are well-defined since all these chains have the same degree.

REMARK 4.4. Instead of the expression in line 18 we could also use

$$\mathit{eliminate} = \text{basisel}(\text{partner}(\sigma)) + \partial\, \text{basisel}(\sigma).$$

Proof. We write $\mathit{eliminate}_k$, τ_k and $\text{basisel}_k(\text{partner}(\sigma))$ for the chains defined in the k-th iteration of the while-loop. The variables $\mathit{eliminate}_0$ and $\text{basisel}_0(\text{partner}(\sigma))$ are the ones defined before we start the while-loop. We use induction. In iteration 0, before the while-loop starts, we have $\mathit{eliminate}_0 = \text{basisel}_0(\text{partner}(\sigma)) + \partial\, \text{basisel}(\sigma)$ because of line 14. We assume that the property holds for all iterations $0, \dots, k$. If $\mathit{eliminate}_k \neq 0$, then it is changed in line 18 at iteration $k+1$:

$$\begin{aligned} \mathit{eliminate}_{k+1} &\overset{\text{line 18}}{=} \mathit{eliminate}_k + \text{basisel}(\tau_{k+1}) \\ &\overset{\text{induction}}{=} \partial\, \text{basisel}(\sigma) + \underbrace{\text{basisel}_k(\text{partner}(\sigma)) + \text{basisel}(\tau_{k+1})}_{\overset{\text{line 17}}{=} \text{basisel}_{k+1}(\text{partner}(\sigma))} \end{aligned}$$

This proves the remark. □

REMARK 4.5. If we want to extend the algorithm to chains with coefficients in some arbitrary field, we have to think about the differences of Gaussian elimination for a matrix with coefficients in this field instead of $\mathbb{F}_2$.

For step 1 we need to add a multiple of the column represented by basisel(partner(τ)) to the column represented by basisel(σ) to generate a zero at the row represented by the basis element of $\tau = \text{youngest}(\partial\, \text{basisel}(\sigma))$ in M_k^m. Therefore, we have to change line 12. Furthermore, we need to modify basisel(σ) by a multiplication with a factor after step 1 to obtain a 1 as pivot element of the column represented by basisel(σ) in the matrix M_k^m.

To make step 2 compatible for arbitrary fields, we need to change the plus in line 14 to a minus and we need to add a multiple of basisel(τ) in lines 17 and 18 such that we generate a 0 in the row represented by basisel(youngest($\mathit{eliminate}$)) of the column represented by basisel(σ).

We want to give a formal proof that the algorithm indeed yields a basis of the desired form. To do this at first we state the following properties. Later on, we will see that these hold in every iteration $\sigma \in K$ of the for-loop.

PROPERTIES 4.6. For Algorithm 1 the following properties hold:

(1) We have $\text{basisel}(\eta) \neq 0 \in C_{\dim(\eta)}(X)$ for all simplices $\eta \in K$ at all times.

(2) We have $\text{youngest}(\text{basisel}(\eta)) = \eta$ for all simplices $\eta \in K$ at all times.

(3) In step 1 of iteration $\sigma \in K$ only the values basisel(σ), partner(σ) and partner(τ) for a simplex $\tau \in K$ with index(τ) < index(σ) can be modified. Simplices in K with a higher index than σ are not even relevant for this step.

(4) If in line 9 the function assign_partner() is called, then we have partner(τ) = $\varnothing$ and partner(σ) = $\varnothing$ at that time.

(5) In step 2 of iteration $\sigma \in K$ only the value basisel(partner(σ)) can be changed, where partner(σ) has a lower index than σ. Simplices in K with a higher index than σ are not even relevant for this step.

We point out that properties (1) and (2) hold at each time in the algorithm. The properties (3) and (4) are formulated specifically for step 1 and (5) for step 2.

Remark 4.7. For Properties 4.6 we make the following observations:

(1) If property (1) holds, then youngest($\cdot$) can be used for each basisel(η), $\eta \in K$. This means that by property (1) we can formulate property (2).

(2) From properties (1) and (2) follows that $\mathcal{B}_k^N$ is a basis of $C_k(X)$ for all $k \in \mathbb{Z}_{\geqslant 0}$.

(3) Properties (1) and (2) imply that each basis element keeps its order within the filtration in the sense that

$$\text{basisel}(\sigma) \in C_k(X_i) \iff \sigma \in X_i \text{ is a } k\text{-simplex.}$$

We conclude that for all $i \in \{0, \dots, N\}$ and $k \in \mathbb{Z}_{\geqslant 0}$ the set $\mathcal{B}_k^i$ is a basis of $C_k(X_i)$ at all times.

(4) Let $\sigma \in X_m$ be a k-simplex. If we assume that properties (1) and (2) hold and τ = youngest(∂(basisel(σ))) is defined, then we have

$$\partial\, \text{basisel}(\sigma) = \text{basisel}(\tau) + \sum_{\substack{\tau' \in \mathcal{A}_{k-1}^{m-1} \\ \text{index}(\tau') < \text{index}(\tau)}} \lambda_{\tau'} \cdot \text{basisel}(\tau')$$

with coefficients $\lambda_{\tau'} \in \mathbb{F}_2$ for all $\tau' \in \mathcal{B}_{k-1}^m$. As in the definition of the function youngest() in (4.5) we denote by $\mathcal{A}_{k-1}^{m-1}$ the set of all $(k-1)$-simplices in X_{m-1}.

(5) Property (4) implies that the assignment of a partner cannot be changed once it is made during the algorithm. We have partner(η) = τ for $\eta, \tau \in K$ if and only if partner(τ) = η.

Proof of Remark *4.7.* The function youngest($\cdot$) is defined for all chains in $C_l(X) - \{0\}$ and each basisel(η), $\eta \in K$ is such a chain at all times by property (1). $\square_{(1)}$

For the proof of (2) we consider the matrix

$$M_{\mathcal{A}_k^N}^{\mathcal{A}_k^N}(\text{basisel}(\cdot))$$

and order the columns and rows by the index of the corresponding simplex. Then the matrix is a triangular matrix with non-zero entries on the main diagonal by properties (1) and (2). We conclude that the matrix $M_{\mathcal{A}_k^N}^{\mathcal{A}_k^N}(\text{basisel}(\cdot))$ is invertible and the set $\mathcal{B}_k^N = \{\text{basisel}(\eta) \mid \eta \in \mathcal{A}_k^N\}$ is a basis of $C_k(X)$. □(2)

For the proof of the direction "⇒" of (3) we let $\text{basisel}(\sigma) \in C_k(X_i)$ be a k-chain. By remark (1) we can use the function youngest$(\cdot)$. We have $\sigma = \text{youngest}(\text{basisel}(\sigma)) \in \mathcal{A}_k^i$ by property (2) and therefore σ is a k-simplex in X_i. The other direction "⇐" follows directly from property (1).

To prove that $\mathcal{B}_k^m$ is a basis of $C_k(X_m)$ for $k \in \mathbb{Z}_{\geqslant 0}$ and $m \in \{0, \dots, N\}$ we let l be the number of k-simplices in X_i. The set $\mathcal{B}_k^i \subseteq \mathcal{B}_k^N$ is $\mathbb{F}_2$-linearly independent by remark (2). The number of k-chains in $\mathcal{B}_k^i \subseteq C_k(X_i)$ is l by the equivalence already proven. Hence, $\mathcal{B}_k^i$ is a basis of the l-dimensional vector space $C_k(X_i)$. □(3)

Now we want to prove (4). Let $\sigma \in X_m$ be a k-simplex and let $\tau_1, \dots, \tau_l \in X_{m-1}$ be the simplices of dimension $k-1$ with $\text{index}(\tau_i) < \text{index}(\tau_j)$ for all $i < j \in \{1, \dots, l\}$. By remark (3) we know that $\text{basisel}(\sigma)$ is in $C_k(X_m)$ and $\partial\,\text{basisel}(\sigma) \in C_{k-1}(X_{m-1})$. Let $i \in \{0, \dots, l\}$ such that $\tau_i = \text{youngest}(\partial\,\text{basisel}(\sigma))$. Also by remark (3) the set $\mathcal{B}_{k-1}^{m-1} = \{\text{basisel}(\tau_1), \dots, \text{basisel}(\tau_l)\}$ is a basis of $C_{k-1}(X_{m-1})$. We obtain

$$\partial\,\text{basisel}(\sigma) = \sum_{j=1}^{l} \lambda_j \cdot \text{basisel}(\tau_j)$$

for some coefficients $\lambda_j \in \mathbb{F}_2$. Let $p = \max\{j \mid \lambda_j \neq 0\}$. Then

$$\partial\,\text{basisel}(\sigma) = 1 \cdot \text{basisel}(\tau_p) + \sum_{j=1}^{p-1} \lambda_j \cdot \text{basisel}(\tau_j)$$

holds. By property (2) the simplex youngest$(\sum_{j=1}^{p-1} \lambda_j \cdot \text{basisel}(\tau_j))$ has a lower index than τ_p, and we know that $\text{youngest}(\text{basisel}(\tau_p)) = \tau_p$. Hence, we conclude that

$$\tau_p = \text{youngest}(\partial\,\text{basisel}(\sigma)) = \tau_i.$$

This proves (4). □(4)

To prove remark (5), we observe that partners can only be changed in line 9. Because of property (4), a partner can only be assigned to simplices, which do not already have a partner. Hence, the partner cannot be changed once it is assigned to a simplex in the algorithm. Furthermore, partners can only be assigned by using Algorithm 2. Therefore, the equivalence holds. □(5)

We conclude that the remark holds. □

By using the properties from above, we will prove the following

Proposition 4.8.

(a) In every iteration of the for-loop step 1 as well as step 2 terminate at some time.

(b) After each iteration $\sigma \in K$ of the for-loop with $m := \text{index}(\sigma)$, the bases

$$\mathcal{B}_k^m = \text{basisel}(\{\eta \in K \mid \dim(\eta) = k, \text{index}(\eta) \leqslant m\}),$$

for $k \in \mathbb{Z}_{\geqslant 0}$ are of the desired form as in equation (4.2)*: Let $\eta \in K$ with* $\operatorname{index}(\eta) \leqslant m$*. If* $\operatorname{partner}(\eta) \neq \varnothing$ *and* $\operatorname{index}(\operatorname{partner}(\eta)) < \operatorname{index}(\eta)$*, we have*

$$\partial \operatorname{basisel}(\eta) = \operatorname{basisel}(\operatorname{partner}(\eta))$$

and otherwise we have

$$\partial \operatorname{basisel}(\eta) = 0.$$

The partners are pairwise different in the sense that $\eta = \eta'$ for all $\eta, \eta' \in K$ with $\operatorname{partner}(\eta) = \operatorname{partner}(\eta')$*.*

Remark 4.9.

(1) Proposition 4.8 (a) causes the whole algorithm to terminate, since the list K is finite.

(2) Assume that for some $m \in \{0, \dots, N\}$ the bases $\mathcal{B}_k^m, k \in \mathbb{Z}_{\geqslant 0}$ are of the form as in Proposition 4.8 (b). Then similarly as for (4.2) the set

$$\{\partial \operatorname{basisel}(\eta) \mid \operatorname{basisel}(\eta) \in \mathcal{B}_k^m, \partial \operatorname{basisel}(\eta) \neq 0\}$$

is a basis of $\operatorname{im}(\partial_k : C_k(X_m) \to C_{k-1}(X_m))$ and

$$\{\operatorname{basisel}(\eta) \mid \operatorname{basisel}(\eta) \in \mathcal{B}_k^m, \partial \operatorname{basisel}(\eta) = 0\}$$

is a basis of $\ker(\partial_k : C_k(X_m) \to C_{k-1}(X_m))$.

(3) If the bases $\mathcal{B}_k^N, k \in \mathbb{Z}_{\geqslant 0}$ are of the form as described above in Proposition 4.8, then also for each $m \in \{0, \dots, N\}$ the bases $\mathcal{B}_k^m, k \in \mathbb{Z}_{\geqslant 0}$ are of this form.

We want to prove the proposition inductively. To do this we first note that Properties 4.6 as well as Proposition 4.8 hold for the first iteration:

The first element σ in the list K has to be a point, since by Remark 4.3 we have $\partial\sigma = 0$. The first step terminates directly without even looking at another simplex. No partner is assigned and therefore step 2 is skipped. We have $\operatorname{basisel}(\eta) = \eta$ for all $\eta \in K$ for the whole time.

To perform the induction step, we introduce the following

Lemma 4.10. *If* Proposition *4.8 and* Properties *4.6 hold for all iterations $\eta \in K$ of the for-loop with* $\operatorname{index}(\eta) \leqslant (m-1)$*, then* Properties *4.6 also hold for iteration $\sigma \in K$ with* $\operatorname{index}(\sigma) = m$*.*

Proof of Lemma *4.10.* Let $\sigma \in K$ be the simplex with $\operatorname{index}(\sigma) = m$. We assume that the proposition and the properties hold for all simplices with a lower index than σ.

For step 1 we prove properties (1) - (4) inductively over the while-loop. Before the first iteration of the while-loop starts, (1) and (2) hold by induction over the for-loop. Since we did not consider any other simplices yet and line 9 was not reached, also (3) and (4) hold.

To do the induction step, we assume that properties (1) - (4) hold for the first $k \in \mathbb{Z}_{\geqslant 0}$ iterations of the while-loop. We want to prove that they also hold for the $(k+1)$-th iteration. Let $\operatorname{basisel}_l(\sigma)$ and $\tau_l = \operatorname{youngest}(\partial \operatorname{basisel}_{l-1}(\sigma))$ be the variables

that are defined in the l-th iteration of the while-loop for $l \in \{1, \dots, k\}$. Furthermore, let $\text{basisel}_0(\sigma)$ be the value of $\text{basisel}(\sigma)$ before the while-loop starts. If we set new values for $\text{basisel}(\sigma)$ and τ in the $(k+1)$-th iteration, we denote them by $\text{basisel}_{k+1}(\sigma)$ and τ_{k+1}.

If in the $(k+1)$-th iteration of the while-loop $\partial\,\text{basisel}_k(\sigma) = 0$, nothing is changed and properties (1) - (4) are fulfilled. In the other case $\partial\,\text{basisel}_k(\sigma) \neq 0$ we have to put in a bit more effort. We note that (1) and (2) hold by induction up to line 12 since basis elements are not modified before in this iteration of the while-loop.

By examining each line of the algorithm we see that (3) holds. We just need that τ_{k+1} has a lower index than σ. REMARK 4.7 (3) can be used before line 12 and yields that $\text{basisel}_k(\sigma) \in C_{\dim(\sigma)(X_m)}$. We obtain

$$\partial\,\text{basisel}_k(\sigma) \in C_{\dim(\sigma)-1}(X_m) = C_{\dim(\sigma)-1}(X_{m-1})$$

and $\tau_{k+1} = \text{youngest}(\partial\,\text{basisel}_k(\sigma)) \in X_{m-1}$. Hence, $\text{index}(\tau_{k+1}) < m = \text{index}(\sigma)$.

To prove property (4), we consider line 9 of the algorithm. Because of line 8, we have $\text{partner}(\tau_{k+1}) = \varnothing$. The partner of σ was not changed in the first k iterations of the while-loop since otherwise step 1 would have terminated. By using properties (3) and (5) for all iterations given by simplices with a lower index than σ, we conclude that $\text{partner}(\sigma) = \varnothing$. Therefore property (4) holds.

Now we will prove property (1) for this iteration of the while-loop. We recall that basis elements can only be changed in line 12 and therefore property (1) holds by induction up to this point. Since line 12 can only be reached if $\text{partner}(\tau_{k+1}) \neq \varnothing$, we assume that τ_{k+1} has a partner. We know that $\text{basisel}_k(\sigma) \neq 0$ and $\text{basisel}(\text{partner}(\tau_{k+1})) \neq 0$.

In the following, will show that $\text{basisel}_k(\sigma)$ and $\text{basisel}(\text{partner}(\tau_{k+1}))$ have the same degree since then

$$\text{basisel}_{k+1}(\sigma) = \text{basisel}_k(\sigma) + \text{basisel}(\text{partner}(\tau_{k+1})) \in C_{\dim(\sigma)}(X)$$

by induction. REMARK 4.7 (5) yields that the simplices τ_{k+1} and $\text{partner}(\tau_{k+1})$ are both assigned as partners to each other. The assignment must have happened in one of the preceding iterations of the for-loop. By property (3) for this iteration, τ_{k+1} and $\text{partner}(\tau_{k+1})$ both have a lower index than σ. We want to show that $\partial\,\text{basisel}(\text{partner}(\tau_{k+1})) = \text{basisel}(\tau_{k+1})$ since then we can conclude

$$\begin{aligned}
\deg(\text{basisel}(\text{partner}(\tau_{k+1}))) &= \deg(\partial\,\text{basisel}(\text{partner}(\tau_{k+1}))) + 1 \\
&= \dim(\tau_{k+1}) + 1 \\
&= \dim(\text{youngest}(\partial\,\text{basisel}_k(\sigma))) + 1 \\
&= \deg(\partial\,\text{basisel}_k(\sigma)) + 1 \\
&= \deg(\text{basisel}_k(\sigma)).
\end{aligned}$$

Since $\text{basisel}_k(\sigma)$ is a simplicial chain, we know that $\partial\,\text{basisel}_k(\sigma) \in \ker(\partial)$. We can use REMARK 4.7 (4) since properties (1) and (2) hold up to line 12, to obtain a unique representation

$$\partial\,\text{basisel}_k(\sigma) = \text{basisel}(\tau_{k+1}) + \sum_{\substack{\eta \in \mathcal{A}^{m-1}_{\dim(\sigma)-1} \\ \text{index}(\eta) < \text{index}(\tau_{k+1})}}$$

of $\partial\,\mathrm{basisel}_k(\sigma)$ in the basis $\mathcal{B}^{m-1}_{\dim(\sigma)-1}$. Remark 4.9 (2) yields that $\partial\,\mathrm{basisel}(\tau_{k+1}) = 0$. Now we use Proposition 4.8 (b) at the iteration $\eta \in K$ with $\mathrm{index}(\eta) = m-1$ of the for-loop. Since $\mathrm{index}(\tau_{k+1}), \mathrm{index}(\mathrm{partner}(\tau_{k+1})) \leqslant m-1$, the basis elements of τ_{k+1} and $\mathrm{partner}(\tau_{k+1})$ are of the desired form. We obtain $\mathrm{index}(\mathrm{partner}(\tau_{k+1})) > \mathrm{index}(\tau_{k+1})$ and $\partial\,\mathrm{basisel}(\mathrm{partner}(\tau_{k+1})) = \mathrm{basisel}(\tau_{k+1})$.

It remains to show that

$$\mathrm{basisel}_{k+1}(\sigma) = \mathrm{basisel}_k(\sigma) + \mathrm{basisel}(\mathrm{partner}(\tau_{k+1})) \neq 0.$$

But his holds since $\mathrm{partner}(\tau_{k+1}) = \mathrm{youngest}(\mathrm{basisel}(\mathrm{partner}(\tau_{k+1})))$ has a lower index than $\sigma = \mathrm{youngest}(\mathrm{basisel}_k(\sigma))$ as proven above. Hence, (1) holds for this iteration of the while-loop.

The preceding argument also shows that

$$\mathrm{youngest}(\mathrm{basisel}_{k+1}(\sigma)) = \mathrm{youngest}(\mathrm{basisel}_k(\sigma) + \mathrm{basisel}(\mathrm{partner}(\tau_{k+1}))) = \sigma$$

which proves (2) for this iteration.

It is left to show that properties (1), (2) and (5) hold for step 2. We note that step 2 terminates immediately if step 1 did not terminate after assigning partners. Hence, we can assume $\mathrm{partner}(\sigma) \neq \varnothing$.

We prove the properties by induction over the while-loop. Before the while-loop starts no value of the simplices in K is modified. Therefore (1) and (2) hold since the basis coincides with the basis from step 1. The simplex $\mathrm{partner}(\sigma)$ has a lower index than σ since it was assigned in step 1 and property (3) holds. Hence, also (5) holds.

We do the induction step by assuming that properties (1), (2) and (5) hold for the first $k \in \mathbb{Z}_{\geqslant 0}$ iterations of the while-loop and do the proof for the $(k+1)$-th iteration. Let τ_l, $\mathrm{basisel}_l(\mathrm{partner}(\sigma))$ and $\mathit{eliminate}_l$ be the variables which are defined in the l-th iteration of the while-loop for $l \in \{1, \ldots, k+1\}$ and let $\mathrm{basisel}_0(\mathrm{partner}(\sigma))$ and $\mathit{eliminate}_0$ be the values before the while-loop starts. Properties (1) and (2) hold until line 17 by induction since the basis elements cannot be changed before.

To prove property (1) for this iteration we just have to check how the basis behaves in line 17. If we can show that $\mathit{eliminate}_k \neq 0$ is a chain in $C_{\dim(\mathrm{partner}(\sigma))}(X)$, then also $\mathrm{basisel}(\tau_{k+1}) \in C_{\dim(\mathrm{partner}(\sigma))}(X)$ since

$$\begin{aligned}\deg(\mathrm{basisel}(\tau_{k+1})) &= \dim(\tau_{k+1}) \\ &= \dim(\mathrm{youngest}(\mathit{eliminate}_k)) \\ &= \deg(\mathit{eliminate}_k)\end{aligned}$$

by property (1) before line 17. Also by property (1) we know that $\mathrm{basisel}_k(\mathrm{partner}(\sigma))$ is a chain in $C_{\dim(\mathrm{partner}(\sigma))}(X)$ and therefore

$$\mathrm{basisel}_{k+1}(\mathrm{partner}(\sigma)) = \mathrm{basisel}_k(\mathrm{partner}(\sigma)) + \mathrm{basisel}(\tau_{k+1}) \in C_{\dim(\mathrm{partner}(\sigma))}(X).$$

Now we show that $\mathit{eliminate}_k \neq 0$ indeed is a chain with degree $\dim(\mathrm{partner}(\sigma))$. By Remark 4.4 or line 14 if $k = 0$ the variable $\mathit{eliminate}_k$ is of the form

$$\mathit{eliminate}_k = \mathrm{basisel}_k(\mathrm{partner}(\sigma)) + \partial\,\mathrm{basisel}(\sigma).$$

By property (1) we have:

$$\begin{aligned}\deg(\text{basisel}_k(\text{partner}(\sigma))) &= \dim(\text{partner}(\sigma)) \\ &= \dim(\text{youngest}(\partial\,\text{basisel}(\sigma))) \\ &= \deg(\partial\,\text{basisel}(\sigma))\end{aligned}$$

Therefore, $\mathit{eliminate}_k$ is a chain and has the same degree as $\text{basisel}_k(\text{partner}(\sigma))$ and $\partial\,\text{basisel}(\sigma)$ which is $\dim(\text{partner}(\sigma))$ by property (1). Furthermore, we know that $\mathit{eliminate}_k \neq 0$ by the condition of the while-loop in line 15.

We prove $\text{basisel}_{k+1}(\text{partner}(\sigma)) \neq 0$ by showing that $\text{youngest}(\text{basisel}(\tau_{k+1}))$ has a lower index than $\text{youngest}(\text{basisel}_k(\text{partner}(\sigma)))$. By property (2) before line 17 we have $\text{youngest}(\text{basisel}_k(\text{partner}(\sigma))) = \text{partner}(\sigma)$ and $\text{youngest}(\text{basisel}(\tau_{k+1})) = \tau_{k+1}$. By Remark 4.4 we have

$$\tau_{k+1} = \text{youngest}(\text{basisel}_k(\text{partner}(\sigma)) + \partial\,\text{basisel}(\sigma)).$$

We know that $\text{youngest}(\partial\,\text{basisel}(\sigma)) = \text{partner}(\sigma)$, which is the same simplex as $\text{youngest}(\text{basisel}_k(\text{partner}(\sigma)))$ by induction. Hence, we obtain that τ_{k+1} has a lower index than $\text{partner}(\sigma)$. This also proves property (2)

In the algorithm we see that only $\text{basisel}(\text{partner}(\sigma))$ can be modified. To prove property (5), we have to show that $\text{partner}(\sigma)$ and τ_{k+1} have a lower index than σ. Since the simplex $\text{partner}(\sigma)$ was assigned in step 1 of this iteration of the for-loop and property (3) holds, it has a lower index than σ. We have already proven above that τ_{k+1} has a lower index than σ. This finishes the proof of the lemma. □

Now, we want to prove Proposition 4.8.

Proof of Proposition *4.8.* We use induction over the iterations of the for-loop. After Remark 4.9 on page 27 we already discussed the base case. Let $m \in \{1, \dots, N\}$. For the induction step we assume that the proposition holds for all iterations $\eta \in K$ of the for-loop with $\text{index}(\eta) \leqslant m-1$. Let $\sigma \in K$ be the simplex with $\text{index}(\sigma) = m$. By using Lemma 4.10 we conclude that Properties 4.6 hold for all iterations $\eta \in K$ of the for-loop with $\text{index}(\eta) \leqslant m$, i.e. they even hold for the iteration σ.

At first we want to prove (a), which states that both steps of the algorithm terminate. We assume that step 1 does not terminate and denote by τ_l and $\text{basisel}_l(\sigma)$ with $l \in \mathbb{Z}_{\geqslant 1}$ the values of τ and $\text{basisel}(\sigma)$ which are defined in the l-th iteration of the while-loop. Furthermore, we denote by $\text{basisel}_0(\sigma)$ the value of $\text{basisel}(\sigma)$ before the while-loop. We will show that

$$\text{index}(\tau_{l+1}) < \text{index}(\tau_l)$$

for all $l \in \mathbb{Z}_{\geqslant 1}$. This contradicts the assumption that there are only finitely many simplices in K and step 1 does not terminate.

Let $k \in \mathbb{Z}_{\geqslant 0}$ be an arbitrary positive integer. By its definition in line 7 and line 12 we have

$$\begin{aligned}\tau_{k+1} &= \text{youngest}(\partial\,\text{basisel}_k(\sigma)) \\ &= \text{youngest}(\partial\,\text{basisel}_{k-1}(\sigma) + \partial\,\text{basisel}(\text{partner}(\tau_k))).\end{aligned}$$

In line 7 the simplex τ_k is defined by $\tau_k = \text{youngest}(\partial\,\text{basisel}_{k-1}(\sigma))$. Furthermore, in the proof of property (1) in LEMMA 4.10 we have shown that

$$\partial\,\text{basisel}(\text{partner}(\tau_k)) = \text{basisel}(\tau_k).$$

By property (2) we know that $\text{youngest}(\text{basisel}(\tau_k)) = \tau_k$. We conclude that τ_{k+1} has a lower index than τ_k.

Now we assume that step 2 does not terminate. We denote by τ_l and $\mathit{eliminate}_l$ for $l \in \mathbb{Z}_{\geqslant 1}$ the values defined in the l-th iteration of the while-loop. Furthermore, let $\mathit{eliminate}_0$ be the value of *eliminate* before the first iteration of the while-loop. As in step 1, we will show that

$$\text{index}(\tau_{l+1}) < \text{index}(\tau_l)$$

for all $l \in \mathbb{Z}_{\geqslant 1}$, which contradicts our assumption.

Let $k \in \mathbb{Z}_{\geqslant 1}$ be some positive integer. By lines 16 and 18 we have

$$\begin{aligned}\tau_{k+1} &= \text{youngest}(\mathit{eliminate}_k)\\ &= \text{youngest}(\mathit{eliminate}_{k-1} + \text{basisel}(\tau_k)).\end{aligned}$$

In line 16 τ_k is defined by $\text{youngest}(\mathit{eliminate}_{k-1}) = \tau_k$. Furthermore, we know that $\text{youngest}(\text{basisel}(\tau_k)) = \tau_k$ by property (2). Hence, $\text{index}(\tau_{k+1}) < \text{index}(\tau_k)$. □(a)

Now we prove that the new basis still has the desired form like in (b). We know by PROPERTIES 4.6 (3) and (5) that only the values of $\text{partner}(\sigma)$, $\text{partner}(\text{partner}(\sigma))$, $\text{basisel}(\sigma)$ and $\text{basisel}(\text{partner}(\sigma))$ can be modified during the algorithm. Furthermore, by REMARK 4.7 (3) the basis elements of the simplices still form a basis.

If $\partial\,\text{basisel}(\sigma) = 0$ at the end of this iteration, then by property (5) we even have $\partial\,\text{basisel}(\sigma) = 0$ after step 1. This means that step 1 terminated in line 5. Hence, no other value than $\text{basisel}(\sigma)$ is modified. The other basis elements are in $\mathcal{B}_k^{m-1}, k \in \mathbb{Z}_{\geqslant 0}$ and are already of the desired form by induction. In this case (b) holds.

If $\partial\,\text{basisel}(\sigma) \neq 0$ at the end of this iteration of the for-loop, then by property (5) $\partial\,\text{basisel}(\sigma) \neq 0$ even holds after step (1). Step 1 still has to terminate by (a) which we have proven above. It terminates in line 10 after the assignment of partners. By property (3) and REMARK 4.7 we have

$$\text{index}(\text{partner}(\sigma)) < \text{index}(\sigma) = \text{index}(\text{partner}(\text{partner}(\sigma))).$$

Since a partner was assigned, the condition in line 13 is fulfilled and the while-loop in step 2 is executed. Again, by using (a) we know that step 2 terminates. But this can only happen if $\mathit{eliminate} = 0$. By REMARK 4.4 we know that

$$\mathit{eliminate} = \partial\,\text{basisel}(\sigma) + \text{basisel}(\text{partner}(\sigma)).$$

We obtain $\partial(\text{basisel}(\sigma)) = \text{basisel}(\text{partner}(\sigma))$ since the coefficients of the chains in this algorithm are in $\mathbb{F}_2$. Furthermore, we note that the basis element $\text{partner}(\sigma)$ still has vanishing boundary since

$$\partial\,\text{basisel}(\text{partner}(\sigma)) = \partial\partial\,\text{basisel}(\sigma) = 0.$$

Two different simplices $\eta, \eta' \in K$ can not have the same partner since REMARK 4.7 (5) holds: If $\text{partner}(\eta) = \xi = \text{partner}(\eta')$ for some $\xi \in K$, then $\eta = \text{partner}(\xi) = \eta'$. By induction, all bases $\mathcal{B}_k^m, k \in \mathbb{Z}_{\geqslant 0}$ are of the desired form. □(b)

This completes the proof of the proposition. □

Remark 4.11. We stated Algorithm 1 for filtrations of simplicial complexes, but many other complexes are also suitable to be used as input. To execute the algorithm we actually just need a sequence of chain complexes

$$C_\bullet^0 \xrightarrow{f_0} C_\bullet^1 \xrightarrow{f_1} \dots \xrightarrow{f_{N-1}} C_\bullet^N$$

with the following properties: Each C_k^i is a finite dimensional vector space over $\mathbb{F}_2$ with basis $\mathcal{B}_k^i$, which can be included by the map f_i into the basis of a higher level:

$$f_i|_{\mathcal{B}_k^i} : \mathcal{B}_k^i \hookrightarrow \mathcal{B}_k^{i+1}$$

For each $i \in \{0, \dots, N-1\}$ there exists $k_i \in \mathbb{Z}_{\geqslant 0}$ and $b \in \mathcal{B}_{k_i}^{i+1}$ such that

$$b \notin \operatorname{im}(f_i|_{\mathcal{B}_{k_i}^i}) \quad \text{and} \quad \operatorname{im}(f_i|_{\mathcal{B}_{k_i}^i}) \cup \{b\} = \mathcal{B}_{k_i}^{i+1} \tag{4.7}$$

and for all $l \neq k_i$

$$\operatorname{im}(f_i|_{\mathcal{B}_l^i}) = \mathcal{B}_l^{i+1} \tag{4.8}$$

holds.

4.3 Tracking Lifetimes

Now we address the problem of figuring out at which steps of the filtration which basis elements of the homology occur and vanish. The algorithm already yields this information. We just have to interpret it in the right way. As noticed in Remark 4.7 (3) we have

$$\text{basisel}(\eta) \in C_k(X_i) \iff \text{index}(\eta) \leqslant i,\ \eta\ k\text{-simplex}.$$

Therefore, we are able to specify a basis of the homology at every stage of the filtration

$$\mathcal{B}_{H_k(X_i)} = \left\{ \text{basisel}(\eta) \;\middle|\; \begin{array}{l} \eta \in K\ k\text{-simplex},\ \text{index}(\eta) \leqslant i \\ \text{and} \begin{cases} \text{partner}(\eta) = \varnothing \\ \text{or index}(\text{partner}(\eta)) > i \end{cases} \end{array} \right\}$$

by using the basis $\mathcal{B}_k^N$ of the last chain complex $C_k(X_N)$ in the filtration (4.1) from Proposition 4.8 (b) and Remark 4.9 (2) and (3), where K is the list of all simplices added by this filtration. We are able to specify the lifetime of each basisel(σ) with vanishing boundary in the homology-sequence of (4.1) by

$$\left[\text{index}(\sigma),\ \begin{array}{l} \text{index}(\text{partner}(\sigma)) \\ \text{or } \infty,\ \text{if partner}(\sigma) = \varnothing \end{array} \right)$$

where ∞ indicates that the basis element still exists in the homology of X_N. We draw $[\text{index}(\sigma), N]$ and extend the interval by a red line instead of $[\text{index}(\sigma), \infty)$.

Up to this point we always considered filtrations (4.1) where one simplex is added at each step. Now we assume that we have a filtration

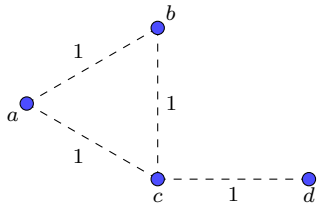
$$\varnothing \hookrightarrow X_0 \hookrightarrow X_1 \hookrightarrow \dots \hookrightarrow X_n, \tag{4.9}$$

of finite simplicial complexes where an arbitrary number of simplices can be added at each step. We note that the indices do not need to have integer values but should have an increasing order. Such filtrations can be obtained from data as in Section 2.4. We still want to be able to track the lifetimes of the generators in the homology sequence. To do this we construct a filtration where just one simplex is added at each step

$$\varnothing \hookrightarrow \widetilde{X}_0 \hookrightarrow \widetilde{X}_1 \hookrightarrow \ldots \hookrightarrow \widetilde{X}_N \tag{4.10}$$

by adding the simplices of each inclusion successively ordered by their dimension.

Example 4.12. We consider the data set consisting of four points $\{a, b, c, d\} = S \subseteq \mathbb{R}^2$, where a, b, c have pairwise distance 1, $\operatorname{dist}(c, d) = 1$, $\operatorname{dist}(a, d) > 1$ and $\operatorname{dist}(b, d) > 1$:



The Vietoris-Rips complexes for the radii 0 and $\frac{1}{2}$ look like

$$\begin{aligned}\operatorname{VR}(S, 0) &= \{\{a\}, \{b\}, \{c\}, \{d\}\} \\ \operatorname{VR}(S, \tfrac{1}{2}) &= \{\{a\}, \{b\}, \{c\}, \{d\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}, \{c, d\}\}\,.\end{aligned}$$

We obtain the sequence $\varnothing \hookrightarrow X_0 = \operatorname{VR}(S, 0) \hookrightarrow X_1 = \operatorname{VR}(S, \frac{1}{2})$. At the first inclusion, there are added $\{a\}, \{b\}, \{c\}$ and $\{d\}$. They all have the same dimension. Therefore, we can add them in arbitrary order:

$$\varnothing \xhookrightarrow{\operatorname{add}_{\{a\}}} \widetilde{X}_0 \xhookrightarrow{\operatorname{add}_{\{b\}}} \widetilde{X}_1 \xhookrightarrow{\operatorname{add}_{\{c\}}} \widetilde{X}_2 \xhookrightarrow{\operatorname{add}_{\{d\}}} \widetilde{X}_3$$

The complex that we obtain for $\widetilde{X}_3$ is the same as $\operatorname{VR}(S, 0)$. In the second inclusion we add the simplices

$$\{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}, \{c, d\}.$$

At first, we order them by their dimension and then arbitrarily:

1. $\{a, b\}, \{a, c\}, \{b, c\}, \{c, d\}$
2. $\{a, b, c\}$

We obtain

$$\widetilde{X}_3 \xhookrightarrow{\operatorname{add}_{\{a,b\}}} \widetilde{X}_4 \xhookrightarrow{\operatorname{add}_{\{a,c\}}} \widetilde{X}_5 \xhookrightarrow{\operatorname{add}_{\{b,c\}}} \widetilde{X}_6 \xhookrightarrow{\operatorname{add}_{\{c,d\}}} \widetilde{X}_7 \xhookrightarrow{\operatorname{add}_{\{a,b,c\}}} \widetilde{X}_8 = \operatorname{VR}(S, \tfrac{1}{2}),$$

and the whole sequence $\varnothing \hookrightarrow \widetilde{X}_1 \hookrightarrow \cdots \hookrightarrow \widetilde{X}_8$ is of the form like (4.10).

By using the algorithm for the new filtration (4.10), we obtain a basis of the form

$$\mathcal{B}_{C_k(X_i)} = \{\text{basisel}(\eta) \mid \eta \in X_i \ k\text{-simplex}\}$$

for each chain group of (4.9) in each step. We assign a new value to each simplex, which indicates its index in the original filtration (4.9):

$$\text{order}(\eta) := \min\{i \mid \eta \in X_i\}$$

It indicates that $\eta \in X_i$ if and only if $\text{order}(\eta) \leqslant i$. This yields a basis

$$\mathcal{B}_{H_k(X_i)} = \left\{ \text{basisel}(\eta) \;\middle|\; \begin{array}{l} \eta \in K, \dim(\eta) = k, \text{order}(\eta) \leqslant i \\ \text{and} \begin{cases} \text{partner}(\eta) = \varnothing \\ \text{or order}(\text{partner}(\eta)) > i \end{cases} \end{array} \right\}$$

for the homology in each step. Instead of stating the lifetimes through indices of filtration (4.10) we are able to specify them in terms of the order:

$$\left[\text{order}(\sigma), \begin{array}{l} \text{order}(\text{partner}(\sigma)) \\ \text{or } \infty, \text{ if } \text{partner}(\sigma) = \varnothing \end{array} \right)$$

These intervals represent the lifetimes of the chains with vanishing boundary in the homology-sequence of filtration (4.9).

We note that the construction of (4.10) is not unique, but the lifetimes in the homology of the original filtration (4.9) are going to be uniquely determined since Proposition 3.5 says that barcodes of directed spaces are unique up to reordering of the intervals. Furthermore, we want to mention that if the sequence is already of the form (4.10) we can assign their indices as orders anyway. Therefore, it suffices to specify the intervals only in terms of the order.

Remark 4.13. By a similar procedure we can even make the algorithm available for sequences of vector spaces as in Remark 4.11 without properties (4.7) and (4.8). In [ZC08, Definition 11 and Theorem 6] such a sequence of chain complexes is called *based persistence complex*.

4.4 Simplification of the Algorithm

In this section we will show that we are able to omit step 2 of Algorithm 1 to draw the barcode but save calculation time.

At first, we state the algorithm which uses only step 1 in each iteration:

Algorithm 3 Simplified persistent homology algorithm.

```
1: def change_basis_without2(K):
2:     for σ ∈ K:
3:         while True:.....Step 1
4:             if ∂ basisel(σ) = 0:
5:                 break
6:             else:
7:                 τ = youngest(∂ basisel(σ))
8:                 if partner(τ) = ∅:
9:                     assign_partner(τ, σ)
10:                    break
11:                else:
12:                    basisel(σ) = basisel(σ) + basisel(partner(τ))
```

Lemma 4.14. *Step 1 in Algorithm 1 and in Algorithm 3 execute the exact same operations.*

Proof. Let $\sigma \in K$ with $\text{index}(\sigma) = n$ be an arbitrary iteration of the for-loop. At the iterations $\eta \in K$ with $\text{index}(\eta) \leqslant n-1$ of Algorithm 1 step 2 changes only basis elements with vanishing boundary of simplices with a lower index than η as stated in Properties 4.6 (5). In step 1 of iteration σ the only simplices which have a lower index than σ and basis elements with vanishing boundary are the simplices described by the variable τ. But only their partner assignment is relevant in step 1. □

We use this lemma to compare the algorithms with and without step 2.

Proposition 4.15. *By comparing the results of Algorithm 1 and Algorithm 3, we observe the following:*

(a) For each $\sigma \in K$ the values of $\text{partner}(\sigma)$ *after using both algorithms coincide.*

(b) For each $\sigma \in K$ with $\text{partner}(\sigma) = \varnothing$ *the values of* $\text{basisel}(\sigma)$ *after using both algorithms coincide.*

(c) For each $\sigma \in K$ with $\text{partner}(\sigma) \neq \varnothing$ *and* $\text{index}(\text{partner}(\sigma)) < \text{index}(\sigma)$ *the values of* $\text{basisel}(\sigma)$ *after using both algorithms coincide.*

(d) For each $\sigma \in K$ with $\text{partner}(\sigma) \neq \varnothing$ *and* $\text{index}(\text{partner}(\sigma)) > \text{index}(\sigma)$ *the values of* $\text{basisel}(\sigma)$ *after using both algorithms can be different.*

Proof. By Lemma 4.14 both algorithms execute the same operations in step 1. In step 2 of iteration $\sigma \in K$ only $\text{basisel}(\text{partner}(\sigma))$ with $\text{index}(\text{partner}(\sigma)) < \text{index}(\sigma)$ can be modified by Properties 4.6 (5). These are the basis elements described in (d). All other variables have to coincide. □

Remark 4.16. To draw barcodes of a filtration like (4.1) we just need the assignment of the partners and the order of each simplex. Since the order cannot be changed by Algorithm 1 and Algorithm 4 and because of Proposition 4.15 (a), Algorithm 4 suffices to draw the barcodes.

When we use Algorithm 3 instead of Algorithm 1 only the values of basisel(σ) for $\sigma \in K$ with partner(σ) = $\varnothing$ and index(partner(σ)) > index(σ) do not coincide. But they are uniquely determined by basisel(partner(σ)) since PROPOSITION 4.8 (b) holds for Algorithm 1. We can adjust these elements by using Algorithm 4 after Algorithm 3 to obtain the same output as Algorithm 1.

Algorithm 4 Adjust basis elements after the simplified persistent homology algorithm.

```
1: def adjust_basisel(K):
2:     for σ ∈ K:
3:         if partner(σ) ≠ ∅ and index(partner(σ)) > index(σ):
4:             basisel(σ) = ∂̂ basisel(partner(σ))
```

4.5 The Implementation

The algorithm is implemented in *Python3* and can be found in the appendix in Section 6.4 and at [Gün19]. The file called *simpcells.py* describes the class of simplicial cells. Each cell has the attributes basisel, partner, index, etc. as described. The file *homology.py* contains the algorithms stated in Section 4.2 and 4.4.

To compute the homology of a complex with these algorithms, we at first need to generate a list K of simplicial cells. We have to add the order and dimension of the cells to ensure good behavior of the functions. Then we use

$$\text{compute\_homology}(K)$$

from homology.py to execute Algorithm 3. Alternatively, we can add the parameter $step2 = True$ to adjust the basis elements afterwards by using Algorithm 4. We can store the barcode and the generators of the homology after using the algorithm by

$$bar, gen = \text{get\_barcodes}(K)$$

from homology.py. At the end the barcodes of the d-th homology can be drawn by using

$$\text{draw\_barcode}(bar, d, K)$$

from homology.py.

An example using this procedure for a simple simplicial complex is

Example_simple_barcodes_of_cell_list.py

which can be found at [Gün19].

4.6 Persistent Homology for Evenly Distributed Points on a Circle

In the paper [AA17] the authors study Vietoris-Rips and Čech complexes of the circle S^1. They state that the Vietoris-Rips complex and the Čech complex yield the homology of all odd-dimensional spheres until finally the complex is contractible if the

radius for the construction of the complexes is increased. In DEFINITION 2.17 and DEFINITION 2.18 we can find the construction of those two complexes.

We want to check by an experiment if we obtain a similar result in the case of a finite subset by studying the homology of the corresponding complexes of n *evenly distributed points* on a circle. For an increasing number of points we could expect to recognize a growing amount of odd-dimensional spheres in homology.

DEFINITION 4.17 (Evenly distributed points). Let $\partial B_R((0,0))$ be a circle with radius $R > 0$ in $\mathbb{R}^2$. Then we have a parametrization of the circle given by

$$\begin{aligned} \phi_s : [s, s+2\pi) &\longrightarrow \partial B_R((0,0)) \\ t &\longmapsto R \cdot \begin{pmatrix} \cos(t) \\ sin(t) \end{pmatrix}. \end{aligned}$$

The points $x_1, \ldots, x_n$ on the circle are *evenly distributed* if they are evenly distributed in each parametrization. Equivalently, they are evenly distributed in one parametrization, where adjacent points have distance $\frac{2\pi}{n}$:

$$\exists s \in \mathbb{R} : \phi_s^{-1}(\{x_1, \ldots, x_n\}) = \left\{ s + \frac{2\pi}{n} \cdot i \;\middle|\; i \in \{0, \ldots, n-1\} \right\}$$

The points are implemented in the two dimensional real vector space $\mathbb{R}^2$ by using sine and cosine. Algorithmically, we generate the Čech complex and the Vietoris-Rips complex for these points. We can easily compute the smallest radius for a simplex $\sigma = \{\sigma_0, \ldots, \sigma_k\}$ in the Vietoris-Rips complex, such that the balls centered at $\sigma_0, \ldots, \sigma_k$ intersect pairwise by

$$r_{min}^{\sigma} = \max \left\{ \tfrac{1}{2} \left\| \sigma_i - \sigma_j \right\|_2 \;\middle|\; \sigma_i, \sigma_j \in \sigma \right\}. \tag{4.11}$$

It is more difficult to compute the Čech complex since we have to check whether all balls intersect

$$\bigcap_{i=0}^{k} B_r(\sigma_i) \neq \varnothing. \tag{4.12}$$

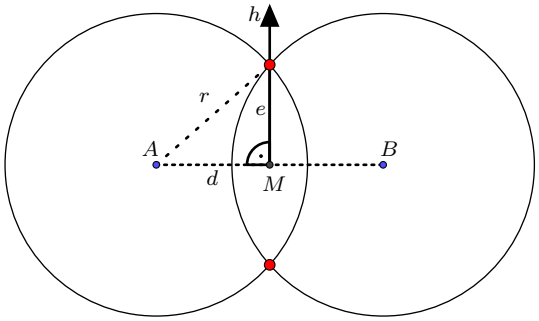
It is not obvious how to determine the lowest radius $r \geqslant 0$ with this property, therefore we follow a numerical approach: If all balls intersect, then they also intersect pairwise. We start with the minimal radius r from (4.11) and incrementally increase r until the intersection property is satisfied. To check whether the property holds, we use an algorithm from [LMDV15] and adapt it to our needs. The authors state that the intersection of all $B_r(\sigma_0), \ldots, B_r(\sigma_k) \subseteq \mathbb{R}^2$ is not empty if and only if

(1) one ball is contained in all of the others or

(2) at least one point in the intersection set of the boundaries $\partial B_r(\sigma_i) \cap \partial B_r(\sigma_j)$ for $i \neq j$ is contained in all balls.

We note that (1) cannot happen for $k \geqslant 1$, since all balls have the same radius and different center. Since we only deal with finitely many balls, we also have only finitely

many intersection points. For our needs it suffices to compute all intersection points and then check successively if one of the points is contained in all balls. We know that all balls have the same radius. Therefore, it is easy to compute their intersection points pairwisely. Let A and B be the centers of two balls with radius $r \geqslant 0$. Furthermore, let $M = \frac{A+B}{2}$ be the midpoint of the line between A and B.



If the distance d between A and B is bigger then $2r$, we have no intersection points for these two balls. Otherwise, we define a vector h, which is orthonormal to the vector from A to M. By using the Pythagorean theorem we can compute $e = \sqrt{r^2 - (\frac{d}{2})^2}$ and obtain the intersection points $M + e \cdot h$ and $M - e \cdot h$.

By implementing this approach for evenly distributed points on the circle with radius 1 we obtain the following results for complexes with different prescribed radii and numbers of points:

(1) in the Vietoris-Rips complex:
   - homology of the points
   - homology of S^1
   - if we have 6 or more points: higher homologies but not only the odd-dimensional spheres.

(2) in the Čech complex:
   - homology of the points
   - homology of S^1
   - many homologies only in small intervals that we cannot classify

The exact results can be found in the appendix in Section 6.1. Since the intervals for the Čech complex are very small and even exceed 1, we can conclude that numerical errors distort the homology crucially. To obtain more precise results we calculate the order of the simplices analytically. For this we need to find a more precise description for the condition with the intersections (4.12).

In the following, let $x_1, \ldots, x_n$ be evenly distributed points on the circle and $p_1, \ldots, p_l$ be a subset of $x_1, \ldots, x_n$ consisting of l elements. The points $p_1, \ldots, p_l$ form an abstract simplex in the filtration of Čech complexes. Its order is the minimal radius r for which the property

$$\bigcap_{i=1}^{l} B_r(p_i) \neq \varnothing \tag{4.13}$$

is fulfilled. We recall that we defined $B_r(p_i)$ to be closed balls for the construction of Čech and Vietoris-Rips complexes. To describe the minimal radius such that all balls at $p_1, \dots, p_l \in \partial B_R((0,0))$ intersect, we distinguish whether all points are *on an open half* of the circle.

Definition 4.18. Let $p_1, \dots, p_l \in \partial B_R((0,0))$ be points on the circle. We define them to be *on an open half* of the circle by the following equivalent properties:

(1) There exists $t \in [0, 2\pi]$ such that $p_1, \dots, p_l \in \{(\sin(x), \cos(x)) \mid x \in (t, t+\pi)\}$.

(2) There is $\alpha \in [0, 2\pi]$ such that $p_1, \dots, p_l \in R_\alpha \, \mathbb{H}$ where

$$R_\alpha = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix}$$

is the rotation matrix for the angle α and $\mathbb{H} = \left\{(x, y) \in \mathbb{R}^2 \;\middle|\; y > 0\right\}$.

(3) We have $R_\alpha p_1, \dots, R_\alpha p_l \in \mathbb{H}$ for some $\alpha \in [0, 2\pi]$.

We can directly state the following

Proposition 4.19. *Let $x_1, \dots, x_n \in \partial B_R((0,0)) \subseteq \mathbb{R}^2$ be evenly distributed points on the circle with radius R. Furthermore, let $\{p_1, \dots, p_l\}$ be a subset of $\{x_1, \dots, x_n\}$ consisting of $l \geqslant 2$ elements.*

(a) If all points $p_1, \dots, p_l$ are on an open half of the circle, then (4.13) *is fulfilled if and only if $r \geqslant \frac{1}{2} \max_{i,j}(\operatorname{dist}(p_i, p_j))$ where $\operatorname{dist}(p_i, p_j) := \|p_i - p_j\|_2$.*

(b) If $p_1, \dots, p_l$ are not on an open half of the circle, then (4.13) *is fulfilled if and only if $r \geqslant R$.*

The property (a) is the same as in the construction of the Vietoris-Rips complex. We can prove Proposition 4.19 (a) directly:

Proof of Proposition *4.19 (a).* At first we prove the direction "⇒". We assume that $\bigcap_{i=1}^{l} B_r(p_i) \neq \varnothing$. Then we have $B_r(p_i) \cap B_r(p_j) \neq \varnothing$ for all $i, j \in \{1, \dots, l\}$. We conclude $\operatorname{dist}(p_i, p_j) \leqslant 2r$ for all i, j and obtain $\max_{i,j}(\operatorname{dist}(p_i, p_j)) \leqslant 2r$.

To prove "⇐" we let p_1 and p_l be the left and right points of the set $\{p_1, \dots, p_l\}$ on

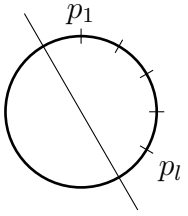


Figure 4.2: Left and right points of $\{p_1, \dots, p_l\}$ on an open half.

the open half in the following sense: If $P \subseteq (t, t+\pi)$ represents the points on the open half by

$$\{p_1, \dots, p_l\} = \{(\sin(x), \cos(x)) \mid x \in P \subseteq (t, t+\pi)\},$$

then p_1 and p_l are given by $(\sin(x), \cos(x))$ for $x = \min P$ and $x = \max P$, respectively. We define $t := \frac{1}{2}\operatorname{dist}(p_1, p_l)$. Let $r \geqslant \frac{1}{2}\max_{i,j}(\operatorname{dist}(p_i, p_j))$ be an arbitrary radius. Then we have $B_r(p_1) \cap B_r(p_l) \neq \varnothing$ since the center $M := \frac{p_1+p_2}{2}$ is in the intersection of both balls. We want to prove that even

$$M \overset{!}{\in} \bigcap_{i=1}^{l} B_r(p_i) \tag{4.14}$$

holds. It suffices to show that $p_i \in B_t(M) \subseteq B_r(M)$ for all i.

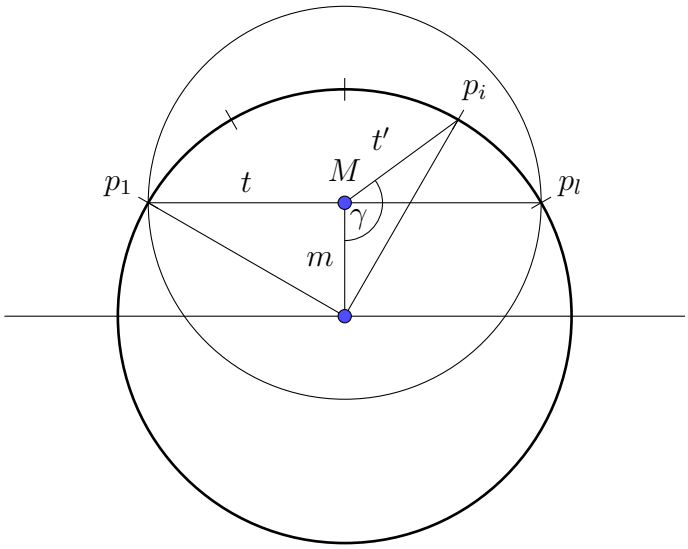


Figure 4.3: Setting for the proof of PROPOSITION 4.19 (a).

Let t' be the distance from M to some point p_i and $m = |M|$ the distance from M to $(0, 0)$ as in Figure 4.3. For a triangle with points a, b, c and angle γ opposite to c, we have by the law of cosines

$$c^2 = a^2 + b^2 - 2ab\cos(\gamma).$$

We can use this for our construction to obtain

$$\begin{aligned} R^2 &= t^2 + m^2 \\ R^2 &= t'^2 + m^2 - 2t'm\cos(\gamma). \end{aligned}$$

Since the angle γ is in $[\frac{\pi}{2}, \frac{3\pi}{2}]$, we obtain

$$R^2 \geqslant t'^2 + m^2 = t'^2 - t^2 + R^2$$

and therefore $t \geqslant t'$. This proves the result. □

For the second part (b), we want to prove that

$$\bigcap_i B_r(p_i) \neq \varnothing \iff (0,0) \in \bigcap_i B_r(p_i)$$

if not all points p_i are on an open half of the circle. To do this we state the following

Lemma 4.20. *The points $p_1, \dots, p_l$ are not on an open half of the circle $\partial B_R((0,0))$ if and only if their convex hull contains the center of the circle:*

$$(0,0) \in \operatorname{conv}(p_1, \dots, p_l)$$

Proof. To prove "$\Leftarrow$" indirectly, we assume that all points are on one side of the circle. We can assume that $p_1, \dots, p_l \in \mathbb{H}$ without loss of generality. But this means that also their convex hull $\operatorname{conv}(p_1, \dots, p_l)$ is in $\mathbb{H}$, which does not contain $(0,0)$.

To prove the other direction "$\Rightarrow$" we assume that not all points are on an open half of the circle. Successively, we remove points with the highest indices until all remaining points $p_1, \dots, p_s$ are on an open half. By rotating the circle and renumbering the points we can assume that $p_1, \dots, p_s \in \mathbb{H}$ and the points p_1 and p_s are the left and right points[1] of $p_1, \dots, p_s$. In order that $p_1, \dots, p_{s+1}$ cannot be on one side of the circle, the property

$$-p_{s+1} \in \operatorname{cone}(p_1, p_s) = \operatorname{cone}(p_1, \dots, p_s)$$

has to hold, where $\operatorname{cone}(p_1, \dots, p_s)$ is the conical hull of the points $p_1, \dots, p_s$. We can find coefficients $\lambda_1, \lambda_s \geqslant 0$, such that $\lambda_1 p_1 + \lambda_s p_s = -p_{s+1}$. By adding p_{s+1} and multiplying with $\lambda = \frac{1}{1+\lambda_1+\lambda_s}$, we obtain

$$(0,0) = \lambda\lambda_1 p_1 + \lambda\lambda_s p_s + \lambda p_{s+1}$$

with $\lambda + \lambda\lambda_1 + \lambda\lambda_s = 1$ and $\lambda, \lambda\lambda_1, \lambda\lambda_s \geqslant 0$. Therefore, $(0,0)$ is contained in the convex hull $\operatorname{conv}(p_1, p_s, p_{s+1}) \subseteq \operatorname{conv}(p_1, \dots, p_l)$ of the points $p_1, \dots, p_l$. □

Remark 4.21. In the proof of the lemma we have shown that if the point $(0,0)$ is in $\operatorname{conv}(p_1, \dots, p_l)$, then we can choose three points $p_{i_1}, p_{i_2}, p_{i_3}$ from $p_1, \dots, p_l$ such that $(0,0)$ is also contained in $\operatorname{conv}(p_{i_1}, p_{i_2}, p_{i_3})$.

We will need a particular choice of such three points $p_{i_1}, p_{i_2}, p_{i_3}$ in the proof of Proposition 4.19 (b) with which we deal now.

Proof of Proposition *4.19 (b).* To prove "$\Leftarrow$" we consider some $r \geqslant R$. We obtain $(0,0) \in B_r(p_i)$ for all i and therefore $(0,0) \in \bigcap_{i=1}^{l} B_r(p_i)$. For this direction we did not even use that $p_1, \dots, p_l$ are not on an open half.

Now we prove "$\Rightarrow$". Let $p_1, \dots, p_l$ be points on the circle that are not on an open half. By Remark 4.21 we are able to choose three points $p_{i_1}, p_{i_2}, p_{i_3}$, such that

$$(0,0) \in \operatorname{conv}(p_{i_1}, p_{i_2}, p_{i_3}) =: D.$$

Without loss of generality we can assume $p_1 = p_{i_1}, p_2 = p_{i_2}$ and $p_3 = p_{i_3}$.

If $(0,0)$ is on one of the edges of the triangle D, two points have to be opposite on the circle. Their distance is $2R$ and balls at these points can only intersect if their radius r is greater or equal R. We assume that $(0,0)$ is in the interior of the triangle and consider the following problem:

$$\text{Find } p \in B_{r'}(p_1) \cap B_{r'}(p_2) \cap B_{r'}(p_3) \text{ for a minimal } r'.$$

[1] The points are left and right as described in the proof of Proposition 4.19 (a).

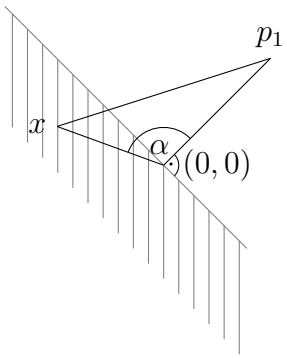
If we manage to solve this, we can conclude that $B_r(p_1) \cap B_r(p_2) \cap B_r(p_3) \neq \varnothing$ if and only if $r \geqslant \max_{i=1,2,3} \operatorname{dist}(p_i, p)$. In the following we will prove that $p = (0,0)$ is this point. We reformulate the problem to: Find $p \in \mathbb{R}^2$ such that it minimizes the function

$$f(x) = \max\{\operatorname{dist}(p_1, x), \operatorname{dist}(p_2, x), \operatorname{dist}(p_3, x)\}.$$

For the origin $(0,0)$, it holds $f(0,0) = \max\{R, R, R\} = R$. Since $(0,0)$ is in the interior of the triangle we can write $\mathbb{R}^2 - \{(0,0)\}$ as

$$\mathbb{R}^2 - \{(0,0)\} = \left\{ x \in \mathbb{R}^2 \;\middle|\; \langle x, p_1 \rangle < 0 \text{ or } \langle x, p_2 \rangle < 0 \text{ or } \langle x, p_3 \rangle < 0 \right\},$$

where $\langle \cdot, \cdot \rangle$ denotes the standard scalar product in $\mathbb{R}^2$. Let $x \in \mathbb{R}^2 - \{(0,0)\}$ be some point which is not the origin. Without loss of generality, we can assume that $\langle x, p_1 \rangle < 0$.



By the law of cosines we obtain

$$\operatorname{dist}(p_1, x)^2 = \operatorname{dist}(x, 0)^2 + \operatorname{dist}(p_1, 0)^2 - 2 \operatorname{dist}(x, 0) \operatorname{dist}(p_1, 0) \cos(\alpha),$$

where $\alpha \in (\frac{\pi}{2}, \frac{3\pi}{2})$ since $\langle x, p_1 \rangle < 0$. The cosine is negative for $\alpha \in (\frac{\pi}{2}, \frac{3\pi}{2})$. Hence, we obtain

$$\operatorname{dist}(p_1, x)^2 > \operatorname{dist}(x, 0)^2 + \operatorname{dist}(p_1, 0)^2 > \operatorname{dist}(p_1, 0)^2$$

and therefore $\operatorname{dist}(p_1, x) > \operatorname{dist}(p_1, 0)$. By the definition of f, we have $f(x) > f(0,0)$ and since $x \in \mathbb{R}^2 - \{(0,0)\}$ was chosen arbitrarily this holds for all those x. We conclude that $(0,0) \in \bigcap_{i=1}^{l} B_r(p_i) \subseteq B_r(p_1)$. Hence $r \geqslant R$. □

We use Proposition 4.19 to improve our algorithm for the construction of the Čech complexes. By implementing the cells directly without the actual position of the vertices and computing the distance of points only once at the beginning, we obtain improved results for the Čech and Vietoris-Rips complex. They can be found in the appendix in Section 6.2.

In the experiment the homology of the Čech complexes for at least 3 points is just the homology of the single points, of S^1 and of one point for different radii. We want to verify this observation by an analytic discussion of the Čech complexes.

Let $S = \{x_1, \ldots, x_n\}$ be the set of n evenly distributed points on the circle with radius R and center in the origin. We denote them in a way such that x_{i+1} is next to x_i for all i and x_1 is next to x_n. Furthermore, let $\check{\mathrm{C}}(X, r)$ be the corresponding Čech complex for the radius r as in Definition 2.17. By Proposition 4.19 we have the following description:

For $0 \leqslant r < R$ we know that a subset $\{x_{i_1}, \ldots, x_{i_l}\} \subseteq \{x_1, \ldots, x_n\}$ consisting of l elements is an abstract l-simplex in $\check{\mathrm{C}}(X, r)$ if and only if $x_{i_1}, \ldots, x_{i_l}$ are on an open half of the circle and their maximal pairwise distance $\max_{j,j' \in \{i_1,\ldots,i_l\}} \operatorname{dist}(x_j, x_{j'})$ is $2r$ at the most. In the case $r \geqslant R$ all subsets $\{x_{i_1}, \ldots, x_{i_l}\} \subseteq \{x_1, \ldots, x_n\}$ are simplices in $\check{\mathrm{C}}(X, r)$. We have the following realization:

$$|\check{\mathrm{C}}(X, R)| = \Delta^{n-1} = \operatorname{conv}(e^1, \ldots, e^n) = \left\{ y \in \mathbb{R}^n \;\middle|\; \textstyle\sum_i y_i = 1, y_i \geqslant 0 \right\} \subseteq \mathbb{R}^n$$

Let r be some radius with $0 \leqslant r < R$. Moreover, let A^r be the set of all sets $\{e^{i_1}, \ldots, e^{i_l}\}$ such that $x_{i_1}, \ldots, x_{i_l}$ are on an open half of the circle and the condition $\max_{j,j' \in \{i_1,\ldots,i_l\}} \operatorname{dist}(x_j, x_{j'}) \leqslant 2r$ is satisfied. We have

$$\Delta^{n-1} \supseteq |\check{\mathrm{C}}(S, r)| = \bigcup_{\{e^{i_1},\ldots,e^{i_l}\} \in A^r} \operatorname{conv}(e^{i_1}, \ldots, e^{i_l}). \tag{4.15}$$

In the following, let $0 < r_1 < r_2 < \cdots < r_{\lfloor \frac{n-1}{2} \rfloor} < R$ be the radii at which the complex changes. We have $r_i = \frac{1}{2} \operatorname{dist}(x_j, x_{j+i})$ for all j. The realization from (4.15) can be written as

$$|\check{\mathrm{C}}(S, r_i)| = \bigcup_{j=1}^{n} \operatorname{conv}(e^j, \ldots, e^{j+i}), \tag{4.16}$$

where here and in the following the indices should be understood modulo n for better readability[2]. Our observations up to this point yield

$$\begin{aligned} |\check{\mathrm{C}}(S,\, 0\,)| &= \coprod_{j=1}^{n} \{e^j\} \\ |\check{\mathrm{C}}(S, r_1)| &= \bigcup_{j=1}^{n} \operatorname{conv}(e^j, e^{j+1}) \\ |\check{\mathrm{C}}(S, R)| &= \Delta^{n-1}. \end{aligned} \tag{4.17}$$

The first complex has the homology of n distinct points and the last one is contractible. The second complex has the homology of the 1-sphere, since it is just one connected component and the only chains with vanishing boundary are multiples of the chain $\sum_{j=1}^{n} (e^j, e^{j+1}) \in C_1(\check{\mathrm{C}}(X, r_1))$. For the remaining complexes we state the following

Lemma 4.22. *Let S and $\{r_i\}_i$ be defined as above. The realization of $\check{\mathrm{C}}(S, r_i)$ like in (4.16) is homotopy equivalent to $|\check{\mathrm{C}}(S, r_1)|$ for all $i \in \{1, \ldots, \lfloor \frac{n-1}{2} \rfloor\}$:*

$$|\check{\mathrm{C}}(S, r_i)| \simeq |\check{\mathrm{C}}(S, r_1)|$$

Hence, all those complexes have the homology of the 1-sphere.

Proof. We do the proof by induction. For $i = 1$ the statement is true. We assume that the lemma holds for all $r_1, \ldots r_{i-1}$ and consider the complex $\check{\mathrm{C}}(S, r_i)$. There is an inclusion

$$\begin{array}{ccc} |\check{\mathrm{C}}(S, r_i)| & = & \bigcup_{j=1}^{n} \operatorname{conv}(e^j, \ldots, e^{j+i}) \\ \cup\!| & & \cup\!| \\ |\check{\mathrm{C}}(S, r_{i-1})| & = & \bigcup_{j=1}^{n} \operatorname{conv}(e^j, \ldots, e^{j+i-1}) \end{array}$$

[2]This means that we define e_j to be $e_{j+k\cdot n}$ for some $k \in \mathbb{Z}$, such that $e_{j+k\cdot n}$ is defined.

of the realizations for different radii. We have a strong deformation retract[3] of

$$\widetilde{\Delta}_j^i := \operatorname{conv}(e^j, \ldots, e^{j+i-1}) \cup \operatorname{conv}(e^{j+1}, \ldots, e^{j+i}) \subseteq \Delta_j^i := \operatorname{conv}(e^j, \ldots, e^{j+i}),$$

which we state on the isomorphic spaces

$$\widetilde{\Delta}_j^i \cong \left\{ y \in \mathbb{R}^{i-1} \;\middle|\; \begin{array}{l} \sum_l y_l \leqslant 1 \\ y_l \geqslant 0 \text{ for all } l \\ y_1 = 0 \text{ or } y_2 = 0 \end{array} \right\} \subset \left\{ y \in \mathbb{R}^{i-1} \;\middle|\; \begin{array}{l} \sum_l y_l \leqslant 1 \\ y_l \geqslant 0 \text{ for all } l \end{array} \right\} \cong \Delta_j^i.$$

It is defined by

$$H_j^i : \Delta_j^i \times [0,1] \longrightarrow \Delta_j^i$$

mapping $((y_1, \ldots, y_{i-1}), t)$ to $(y_1, \ldots, y_{i-1}) + t \cdot \min\{y_1, y_2\} \cdot (-1, -1, 0, \ldots, 0)$. It is left to the reader to check that indeed $H_j^i(y, 0) = y$, $H_j^i(y, 1) \in \widetilde{\Delta}_j^i$ and $H_j^i(a, t) = a$ for all $y \in \Delta_j^i$, $t \in [0, 1]$ and $a \in \widetilde{\Delta}_j^i$. The properties $\bigcup_j \Delta_j^i = |\check{\mathrm{C}}(S, r_i)|$, $\bigcup_j \widetilde{\Delta}_j^i = |\check{\mathrm{C}}(S, r_{i-1})|$ and $\Delta_j^i \cap \Delta_{j'}^i \subseteq \widetilde{\Delta}_j^i \cap \widetilde{\Delta}_{j'}^i$ for all $j \neq j'$ hold. Therefore, we can define the strong deformation retract

$$\begin{aligned} H^i : |\check{\mathrm{C}}(S, r_i)| \times [0,1] &\longrightarrow |\check{\mathrm{C}}(S, r_i)| \\ (x, t) &\longmapsto H_j^i(x, t), \text{ for } x \in \Delta_j^i \end{aligned}$$

on the whole Čech complex. This forms a strong deformation retract from $|\check{\mathrm{C}}(S, r_i)|$ onto $|\check{\mathrm{C}}(S, r_{i-1})|$. Hence, both realizations are homotopy equivalent and their homology coincides. □

Remark 4.23. For the trick in the proof of the last lemma we did not explicitly use that the points are evenly distributed or that the dimension of the sphere containing the points is 1. Therefore, it should be possible to extend our results to non-evenly distributed points on spheres in higher dimensions.

[3] Its definition can be found in [Hat02, Chapter 0]. There it is just called deformation retract and the weak version is stated in exercise 4.

5 Localizing Holes

The algorithm in the preceding chapter yields descriptions for the holes of a simplicial complex in form of representatives of the non-vanishing homology classes. They can be very inconvenient in the sense that they enclose points that are far away from the hole.

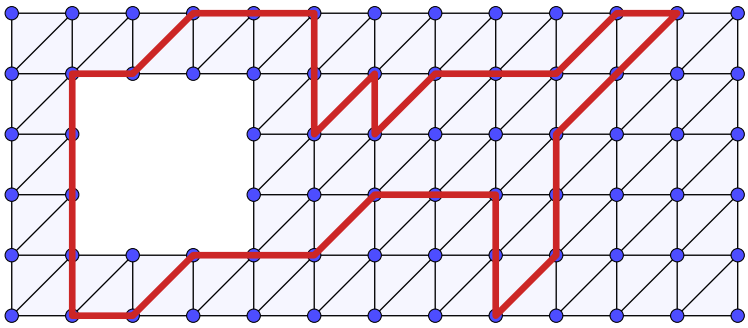


Figure 5.1: Bad description of a hole.

We want to avoid that by finding better local descriptions. The idea is to partition the complex and start searching for representatives there.

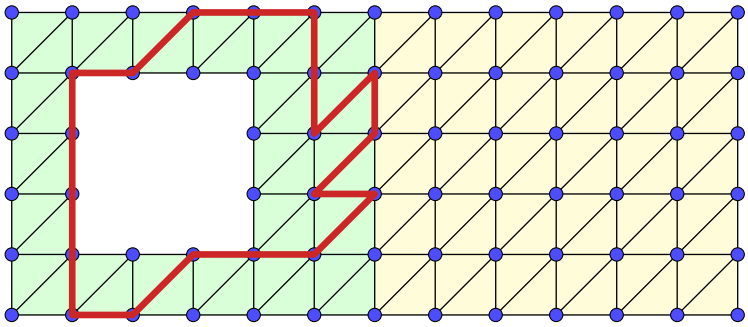


Figure 5.2: Improved description of a hole.

To do this we introduce the *Mayer-Vietoris blowup* in Section 5.2, which lives in the product of the simplicial complexes X and Δ^n. This product enables us to separate local parts and glue them together at a later point. By tracking the homology through this gluing process we obtain good local descriptions. The following statements and proofs hold for homology modules over any commutative ring with 1.

5.1 Handling Products of Simplicial Complexes

The *Mayer-Vietoris blowup*, which is the key construction for the localization algorithm and will be discussed in the next section, makes use of products of simplicial complexes.

In this section we recall the construction of a product and explain the relation between the chain complex of a product and the product of chain complexes.

Let X and Y be simplicial complexes. We know that the product of their realizations $|X|\times|Y|$ is also a topological space. It can be triangulated in a natural way if we assume that the simplices of both complexes X and Y come equipped with total orderings $<_X$ and $<_Y$ on their vertex sets V_X and V_Y. Note that this extra datum is also essential in our algorithm. On the next pages we follow the construction from [ZC08].

Let the set of all 0-dimensional simplices of X and Y be the *vertex sets* V_X and V_Y, respectively. We define a vertex set

$$V_{X\times Y} := V_X \times V_Y$$

of the product $X \times Y$. A subset $\sigma \subseteq V_{X\times Y}$ is defined to be an abstract simplex of $X \times Y$ if and only if there are simplices $\sigma_X \in X$ and $\sigma_Y \in Y$ such that $\sigma \subseteq \sigma_X \times \sigma_Y$ and the restriction of the total orderings

$$(p_1, p_2) <_{X\times Y} (q_1, q_2) \iff \begin{array}{c} p_1 <_X q_1 \text{ and } p_2 \leqslant_Y q_2 \\ \text{or} \\ p_1 \leqslant_X q_1 \text{ and } p_2 <_Y q_2 \end{array}$$

forms a total ordering on σ.

Definition 5.1. For (abstract) simplicial complexes X and Y with total vertex orderings we define the abstract simplicial complex $X \times Y$ to be

$$X \times Y := \left\{ \sigma \subseteq V_{X\times Y} \;\middle|\; \begin{array}{l} \sigma \text{ receives a total ordering} \\ \text{induced by those on } V_X \text{ and } V_Y \end{array} \right\}.$$

The faces of each simplex in $X \times Y$ are the typical ones for abstract simplices and are again in $X \times Y$. This makes $X \times Y$ a well-defined complex.

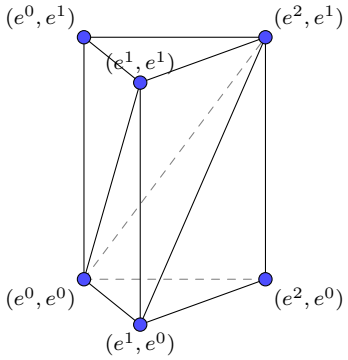


Figure 5.3: Product of Δ^2 and Δ^1.

Remark 5.2. Let Δ^{k_1} and Δ^{k_2} with $k_1, k_2 \in \mathbb{Z}_{\geqslant 0}$ be two simplices. With the definition of a product from above, we obtain that $|\Delta^{k_1}| \times |\Delta^{k_2}|$ is a geometric realization of the product $\Delta^{k_1} \times \Delta^{k_2}$. This is a standard argument, which can be found for example in [GZ67, Chapter 3, Section 3.4] for simplicial sets. It can be shown that this also holds for general simplicial complexes:

$$|X| \times |Y| \cong |X \times Y|$$

Next we will show that the homology of a product can be computed by the product of the chain complexes. This is useful, as the latter is usually a much smaller complex.

Definition 5.3 (Tensor product of chain complexes). Let $(V_\bullet, \partial^V_\bullet)$ and $(W_\bullet, \partial^W_\bullet)$ be chain complexes. Then their *tensor product* is defined by

$$(V_\bullet \otimes W_\bullet)_k := \bigoplus_{p+q=k} (V_p \otimes W_q)$$

for all $k \in \mathbb{Z}$ with boundary maps given by $\partial_k(a \otimes b) = \partial^V_p a \otimes b + (-1)^{\deg a} a \otimes \partial^W_q b$ for $a \in V_p$ and $b \in W_q$.

We can easily verify that $C_\bullet(X) \otimes C_\bullet(Y)$ is smaller than $C_\bullet(X \times Y)$ by constructing an injective map from the basis set

$$\mathcal{B}_{(C_\bullet(X) \otimes C_\bullet(Y))_k} = \left\{ \sigma \otimes \eta \;\middle|\; \begin{array}{l} \sigma \in X p\text{-simplex}, \eta \in Y q\text{-simplex} \\ \text{with } p+q=k \text{ and orientation} \\ \text{given by total vertex ordering} \end{array} \right\}$$

to $\mathcal{B}_{C_k(X \times Y)} = \{k\text{-simplices of } X \times Y\}$ where each simplex in the basis is oriented by the total vertex ordering, sending $(\sigma_0, \ldots, \sigma_p) \otimes (\eta_0, \ldots, \eta_q)$ to the basis element $((\sigma_0, \eta_0), \ldots, (\sigma_p, \eta_0), (\sigma_p, \eta_1), \ldots, (\sigma_p, \eta_q))$.

To prove that both complexes yield the same homology, we compare the modules of k-chains by the *Eilenberg-Zilber theorem*, which is also known as the *Alexander-Whitney theorem*, as in [ZC08].

Theorem 5.4 (Eilenberg-Zilber). *Let X and Y be simplicial complexes with total vertex orderings $<_X$ and $<_Y$. Like in the construction above this gives rise to a product $X \times Y$ of simplicial complexes. Then we can find chain maps*

$$C_\bullet(X \times Y) \underset{\mathcal{S}=\mathcal{S}^{X,Y}}{\overset{\mathcal{A}=\mathcal{A}^{X,Y}}{\rightleftarrows}} C_\bullet(X) \otimes C_\bullet(Y)$$

on chain complexes, which are natural in the sense of the following remark and induce an isomorphism on homology modules.

Remark 5.5. Having natural chain maps means that they commute with inclusions of simplicial complexes. Formally speaking, let $A(\cdot,\cdot), B(\cdot,\cdot) \in \{C_\bullet(\cdot \times \cdot), C_\bullet(\cdot) \otimes C_\bullet(\cdot)\}$ be two maps which assign some chain complex to each pair of simplicial complexes with total vertex orderings. For inclusions of simplicial complexes with total vertex ordering $\iota_0 : X \hookrightarrow X'$ and $\iota_1 : Y \hookrightarrow Y'$ we have induced maps[1] on the chain complexes, which we call

$$\iota_A : A(X,Y) \longrightarrow A(X',Y')$$
$$\iota_B : B(X,Y) \longrightarrow B(X',Y').$$

If we have a chain map $f^{X,Y} : A(X,Y) \to B(X,Y)$ for all simplicial complexes X and Y with total vertex ordering, then we call f *natural* if for all inclusions $X \subseteq X'$

[1] The induced maps for $C_\bullet(\cdot \times \cdot)$ and $C_\bullet(\cdot) \otimes C_\bullet(\cdot)$ are $\iota_0 \times \iota_1$ and $\iota_0 \otimes \iota_1$ respectively.

and $Y \subseteq Y'$ the diagram

$$\begin{array}{ccc} A(X,Y) & \xrightarrow{f^{X,Y}} & B(X,Y) \\ \downarrow{\scriptstyle \iota_A} & & \downarrow{\scriptstyle \iota_B} \\ A(X',Y') & \xrightarrow{f^{X',Y'}} & B(X',Y') \end{array}$$

commutes.

For the Eilenberg-Zilber theorem we adapt the proof of [Bre97, Chapter 6, Section 1] and [SZ88, Satz 12.2.6]. We will do the construction of $\mathcal{A}$ and $\mathcal{S}$ in a simpler case where X and Y are simplices and then we will use the naturality to extend the definition to arbitrary simplicial complexes.

Remark 5.6. Let X and Y be simplicial complexes with total vertex orderings.

(1) Let $\sigma \in X \times Y$ be a k-simplex. Then there are inclusions of simplicial complexes

$$\Delta^r \xrightarrow{\iota^0_\sigma} X \qquad \text{and} \qquad \Delta^s \xrightarrow{\iota^1_\sigma} Y$$

with $r, s \in \mathbb{Z}_{\geqslant 0}$ and a simplex $e_\sigma \in \Delta^r \times \Delta^s$, such that $(\iota^0_\sigma \times \iota^1_\sigma)(e_\sigma) = \sigma$.

(2) Let $\eta \otimes \tau \in C_p(X) \otimes C_q(Y)$ with $p, q \in \mathbb{Z}_{\geqslant 0}$ be a basis element. Then we can find inclusions of simplicial complexes

$$\Delta^p \xrightarrow{\iota_\eta} X \qquad \text{and} \qquad \Delta^q \xrightarrow{\iota_\tau} Y,$$

such that $(\iota_\eta \otimes \iota_\tau) : C_p(\Delta^p) \otimes C_q(\Delta^q) \to C_p(X) \otimes C_q(Y)$ maps $\Delta^p \otimes \Delta^q$ to $\eta \otimes \tau$.

We note that (1) also yields a map $(\iota^0_\sigma \times \iota^1_\sigma) : C_k(\Delta^r \times \Delta^s) \to C_k(X \times Y)$ which maps e_σ to σ. Since we can describe bases of $C_k(X \times Y)$ and $C_p(X) \otimes C_q(Y)$, we can construct any element in those chain groups. It is not always possible to choose r and s in part (1) of the remark such that $r + s = k$ but we always have $p + q = k$ in (2).

Proof. For (1) let $\sigma = \{(\eta_0, \tau_0), \ldots, (\eta_k, \tau_k)\} \in X \times Y$ be any k-simplex. By removing duplicates, we obtain simplices $\{\eta_{i_1}, \ldots, \eta_{i_r}\} \in X$ and $\{\tau_{j_1}, \ldots, \tau_{j_s}\} \in Y$. We define the maps $\iota^0_\sigma : \Delta^r \to X, e^t \mapsto \eta_{i_t}$ for all $t \in \{0, \ldots, r\}$ and $\iota^1_\sigma : \Delta^s \to X, e^t \mapsto \tau_{i_t}$ for all $t \in \{0, \ldots, s\}$. Let further

$$p_\eta : \{0, \ldots, k\} \longrightarrow \{i_0, \ldots, i_r\} \qquad \text{and} \qquad p_\tau : \{0, \ldots, k\} \longrightarrow \{j_0, \ldots, j_s\}$$

be the unique maps with $\eta_l = \eta_{p_\eta(l)}$ and $\tau_l = \tau_{p_\tau(l)}$ for all $l \in \{0, \ldots, k\}$. Then we obtain a k-simplex

$$e_\sigma = \{(e^{p_\eta(0)}, e^{p_\tau(0)}), \ldots, (e^{p_\eta(k)}, e^{p_\tau(k)})\} \in \Delta^r \times \Delta^s$$

with $(\iota^0_\sigma \times \iota^1_\sigma)(e_\sigma) = \{(\eta_{p_\eta(0)}, \tau_{p_\tau(0)}), \ldots, (\eta_{p_\eta(k)}, \tau_{p_\tau(k)})\} = \sigma$. □(1)

Let $\eta = \{\eta_0, \ldots, \eta_p\} \in X$ and $\tau = \{\tau_0 \ldots, \tau_q\} \in Y$ be simplices. Then the simplicial maps

$$\begin{array}{ccc} \iota_\eta : \Delta^p \longrightarrow X & \qquad \text{and} \qquad & \iota_\tau : \Delta^p \longrightarrow X \\ e^t \longmapsto \eta_t & & e^t \longmapsto \tau_t \end{array}$$

satisfy $\iota_\eta(\Delta^p) = \eta$ and $\iota_\tau(\Delta^q) = \tau$. We obtain a well-defined chain map

$$\iota_\eta \otimes \iota_\tau : C_p(\Delta^p) \otimes C_q(\Delta^q) \longrightarrow C_p(X) \otimes C_q(Y)$$

with $(\iota_\eta \otimes \iota_\tau)(\Delta^p \otimes \Delta^q) = (\eta \otimes \tau)$ in $C_p(X) \otimes C_q(Y)$. □(2)

This finishes the proof of the lemma. □

Lemma 5.7. *For all $l \in \mathbb{Z}_{\geqslant 1}$ the following homology modules vanish:*

(a) $H_l(C_\bullet(\Delta^r \times \Delta^s)) = 0$ for all $r, s \in \mathbb{Z}_{\geqslant 0}$

(b) $H_l(C_\bullet(\Delta^p) \otimes C_\bullet(\Delta^q)) = 0$ for all $p, q \in \mathbb{Z}_{\geqslant 0}$

Proof. By Remark 5.2 we know that

$$|\Delta^r \times \Delta^s| = |\Delta^r| \times |\Delta^s| \simeq \{pt\}$$

is contractible. □(a)

To prove part (b) of the lemma, we adapt the idea of the proof of the *Künneth theorem* from [Hat02, Theorem 3B.5] and customize it for our needs. For $k \in \mathbb{Z}_{\geqslant 0}$ let

$$\ldots \longrightarrow C_{i+1}(\Delta^k) \xrightarrow{\partial^k_{i+1}} C_i(\Delta^k) \xrightarrow{\partial^k_i} C_{i-1}(\Delta^k) \longrightarrow \ldots$$

be the simplicial chain complex. For better readability we define $B^k_i := \mathrm{im}(\partial^k_{i+1})$, $Z^k_i := \ker(\partial^k_i)$ and $C^k_i := C_i(\Delta^k)$. Furthermore, we will omit indices of the boundary map if their domain and codomain is obvious in the context. For any k, we have chain complexes $C^k_\bullet$, $Z^k_\bullet$ and $B^k_\bullet$:

$$\begin{array}{ccccccc}
\ldots & \longrightarrow & C^k_i & \xrightarrow{\partial^k_i} & C^k_{i-1} & \longrightarrow & \ldots \\
 & & \cup\!| & & \cup\!| & & \\
\ldots & \longrightarrow & Z^k_i & \xrightarrow{\partial^k_i = 0} & Z^k_{i-1} & \longrightarrow & \ldots \\
 & & \cup\!| & & \cup\!| & & \\
\ldots & \longrightarrow & B^k_i & \xrightarrow{\partial^k_i = 0} & B^k_{i-1} & \longrightarrow & \ldots
\end{array}$$

For p as in the statement of the lemma and any $s \in \mathbb{Z}$ we obtain a short exact sequence of modules:

$$0 \longrightarrow Z^p_s \xhookrightarrow{\iota} C^p_s \xrightarrow{\partial} B^p_{s-1} \longrightarrow 0$$

Tensoring the sequence with the free module C^q_t for any $t \in \mathbb{Z}$ yields a sum of short exact sequences which is a short exact sequence itself. We refer to [Rot09, Theorem 3.1 and Proposition 3.46] for a formal proof that by tensoring with C^q_t we obtain a short exact sequence. The new short exact sequence is of the form:

$$0 \longrightarrow Z^p_s \otimes C^q_t \xrightarrow{\iota \otimes \mathsf{id}} C^p_s \otimes C^q_t \xrightarrow{\partial \otimes \mathsf{id}} B^p_{s-1} \otimes C^q_t \longrightarrow 0$$

For all $i \in \mathbb{Z}$, using $\bigoplus_{s+t=i-1} B^p_s \otimes C^q_t = \bigoplus_{s+t=i} B^p_{s-1} \otimes C^q_t$ and the sequence from above, we can define the following short exact sequence of modules:

$$\begin{array}{ccccccccc}
0 & \longrightarrow & (Z^p_\bullet \otimes C^q_\bullet)_i & \longrightarrow & (C^p_\bullet \otimes C^q_\bullet)_i & \longrightarrow & (B^p_\bullet \otimes C^q_\bullet)_{i-1} & \longrightarrow & 0 \\
 & & \| & & \| & & \| & & \\
 & & \bigoplus_{s+t=i} Z^p_s \otimes C^q_t & & \bigoplus_{s+t=i} C^p_s \otimes C^q_t & & \bigoplus_{s+t=i} B^p_{s-1} \otimes C^q_t & &
\end{array}$$

To prove that this even defines a short exact sequence of chain complexes we have to check whether the maps in each degree commute with the boundary maps. We consider the following diagram:

$$\begin{array}{ccccccccc} 0 & \longrightarrow & (Z^p_\bullet \otimes C^q_\bullet)_i & \xrightarrow{\iota\otimes\mathsf{id}} & (C^p_\bullet \otimes C^q_\bullet)_i & \xrightarrow{\partial\otimes\mathsf{id}} & (B^p_\bullet \otimes C^q_\bullet)_{i-1} & \longrightarrow & 0 \\ & & \downarrow\partial & (1) & \downarrow\partial & (2) & \downarrow\partial & & \\ 0 & \longrightarrow & (Z^p_\bullet \otimes C^q_\bullet)_{i-1} & \xrightarrow{\iota\otimes\mathsf{id}} & (C^p_\bullet \otimes C^q_\bullet)_{i-1} & \xrightarrow{\partial\otimes\mathsf{id}} & (B^p_\bullet \otimes C^q_\bullet)_{i-2} & \longrightarrow & 0 \end{array}$$

The commutativity of (1) holds, since the maps on the rows are just the inclusions and both boundary maps coincide. To check whether (2) commutes, we take some $c \otimes c' \in (C^p_\bullet \otimes C^q_\bullet)_i$ and use the maps of the diagram:

$$\begin{array}{ccc} c \otimes c' & \xmapsto{\partial\otimes\mathsf{id}} & \partial c \otimes c' \\ \downarrow\partial & & \downarrow\partial \\ \partial c \otimes c' & & \partial\partial c \otimes c' \\ +(-1)^{|c|} c \otimes \partial c' & \mapsto\!\!\!\!\cdots & +(-1)^{|\partial c|} \partial c \otimes \partial c' \end{array}$$

Since $\partial\partial c$ vanishes, the diagram commutes up to a sign. This can be fixed by considering the map

$$(-1)^i \partial \otimes \mathsf{id} : (C^p_\bullet \otimes C^q_\bullet)_i \longrightarrow (C^p_\bullet \otimes B^q_\bullet)_{i-1}$$

instead of $\partial \otimes \mathsf{id}$. We obtain a short exact sequence of chain complexes given by:

$$0 \longrightarrow (Z^p_\bullet \otimes C^q_\bullet)_i \xrightarrow{\iota\otimes\mathsf{id}} (C^p_\bullet \otimes C^q_\bullet)_i \xrightarrow{(-1)^i\partial\otimes\mathsf{id}} (B^p_\bullet \otimes C^q_\bullet)_{i-1} \longrightarrow 0$$

As in the *Snake Lemma* [HS97, Chapter III, Lemma 5.1] this yields a long exact sequence in homology

$$\ldots \longrightarrow H_i(Z^p_\bullet \otimes C^q_\bullet) \longrightarrow H_i(C^p_\bullet \otimes C^q_\bullet) \longrightarrow H_{i-1}(B^p_\bullet \otimes C^q_\bullet) \longrightarrow \ldots,$$

where the *connecting homomorphism* $H_i(B^p_\bullet \otimes C^q_\bullet) \to H_i(Z^p_\bullet \otimes C^q_\bullet)$ is just the inclusion $\iota \otimes \mathsf{id}$ multiplied with a sign. We will see this by the following diagram chase:

If we have bases $\mathcal{B}_{B^p_s}$ and $\mathcal{B}_{C^q_t}$ for B^p_s and C^q_t, then $\mathcal{B}^{p,q}_{s,t} := \left\{ b \otimes c \;\middle|\; b \in \mathcal{B}_{B^p_s}, c \in \mathcal{B}_{C^q_t} \right\}$ is a basis of $B^p_s \otimes C^q_s$. Let

$$\sum_{b\otimes c \in \mathcal{B}^{p,q}_{s,t}} \lambda_{b,c} b \otimes c = \sum_{b \in \mathcal{B}_{B^p_s}} b \otimes c_b$$

be any element in $B^p_s \otimes C^q_s$. Its boundary is $0 + \sum_{b\in\mathcal{B}_{B^p_s}} (-1)^{|c|} b \otimes \partial c_b$. If the boundary vanishes, then $\partial c_b = 0$ for all $b \in \mathcal{B}_{B^p_s}$ and the summand $b \otimes c_b$ is already a chain with vanishing boundary. Hence, it suffices to define the map $H_i(B^p_\bullet \otimes C^q_\bullet) \to H_i(Z^p_\bullet \otimes C^q_\bullet)$ just for elements of the form $c \otimes c' \neq 0 \in B^p_s \otimes C^q_t$ with vanishing boundary. We have $c \neq 0$ and $c' \neq 0$. Since the boundaries $\partial(c\otimes c') = \partial c \otimes c' + (-1)^{|c|} c \otimes \partial c'$ and ∂c vanish, also the boundary of c' has to vanish. The map $(-1)^{s+t+1}\partial \otimes \mathsf{id} : C^p_{s+1} \otimes C^q_t \to B^p_s \otimes C^q_t$

is surjective. Hence, there exists a chain $d \in C^p_{s+1}$ such that $(-1)^{s+t+1}\partial(d) = c$. We obtain:

$$\begin{array}{ccc} d \otimes c' & \xmapsto{(-1)^{s+t+1}\partial\otimes\mathsf{id}} & c \otimes c' \\ \downarrow{\scriptstyle \partial} & & \downarrow{\scriptstyle \partial} \\ \partial d \otimes c' & \longmapsto & \partial d \otimes c' \qquad\quad 0 \end{array}$$

Therefore, the map $H_i(B^p_\bullet \otimes C^q_\bullet) \to H_i(Z^p_\bullet \otimes C^q_\bullet)$ in the long exact sequence is defined by $(-1)^{i+1}\iota \otimes \mathsf{id}$, where $i = s + t$.

We will show that the homology modules $H_i(C^p_\bullet \otimes C^q_\bullet)$ for $i \geqslant 1$ vanish, since they are enclosed by zero maps in the long exact sequence. The map

$$\begin{aligned} \partial_{Z^p_s \otimes C^q_t} : Z^p_s \otimes C^q_t &\longrightarrow Z^p_{s-1} \otimes C^q_t \oplus Z^p_s \otimes C^q_{t-1} \\ z \otimes c &\longmapsto \partial_{Z^p_s} z \otimes c + (-1)^p z \otimes \partial_{C^q_t} c = (-1)^p z \otimes \partial_{C^q_t} c \end{aligned}$$

has its image in $Z^p_s \otimes C^q_{t-1}$ since $\partial_{Z^p_s} z = 0$. In the following, we write $\partial_{s,t}$ for $\partial_{Z^p_s \otimes C^q_t}$ and ∂_i for $\partial_{(Z^p_\bullet \otimes C^q_\bullet)_i}$. Since

$$\partial_i : \bigoplus_{s+t=i} Z^p_s \otimes C^q_t \longrightarrow \bigoplus_{s+t=i} Z^p_s \otimes C^q_{t-1}$$

is defined by $\partial_{s,t}$ on each summand, we obtain

$$\begin{aligned} \ker(\partial_i) &= \bigoplus_{s+t=i} \ker(\partial_{s,t}) \\ \operatorname{im}(\partial_{i+1}) &= \bigoplus_{s+t=i+1} \operatorname{im}(\partial_{s,t}). \end{aligned}$$

We can specify how the boundary maps look like:

$$\begin{aligned} \ker(\partial_{s,t}) &= Z^p_s \otimes \ker(\partial_{C^q_t}) \\ \operatorname{im}(\partial_{s,t}) &= Z^p_s \otimes \operatorname{im}(\partial_{C^q_t}) \end{aligned}$$

This yields

$$\begin{aligned} \ker(\partial_i) &= \bigoplus_{s+t=i} Z^p_s \otimes \ker(\partial_{C^q_t}) \\ \operatorname{im}(\partial_{i+1}) &= \bigoplus_{s+t=i+1} Z^p_s \otimes \operatorname{im}(\partial_{C^q_t}) = \bigoplus_{s+t=i} Z^p_s \otimes \operatorname{im}(\partial_{C^q_{t+1}}). \end{aligned}$$

We have inclusions $Z^p_s \otimes \operatorname{im}(\partial_{C^q_{t+1}}) \subseteq Z^p_s \otimes \ker(\partial_{C^q_t})$. By using the description for the kernel and image and the inclusions it follows that

$$\begin{aligned} H_i(Z^p_\bullet \otimes C^q_\bullet) &= \ker(\partial_i) \big/ \operatorname{im}(\partial_{i+1}) \\ &= \bigoplus_{s+t=i} \left((Z^p_s \otimes \ker(\partial_{C^q_t})) \big/ (Z^p_s \otimes \operatorname{im}(\partial_{C^q_{t+1}})) \right). \end{aligned}$$

Tensoring with Z_s^p is right exact. Hence, we conclude

$$\begin{aligned} H_i(Z_\bullet^p \otimes C_\bullet^q) &= \bigoplus_{s+t=i} Z_s^p \otimes \left(\ker(\partial_{C_t^q}) / \operatorname{im}(\partial_{C_{t+1}^q}) \right) \\ &= \bigoplus_{s+t=i} Z_s^p \otimes H_t(C_\bullet^q). \end{aligned}$$

By using that Δ^q is contractible, we obtain that $H_t(C_\bullet^q)$ is the coefficient ring for $t = 0$ and 0 for each other t. This yields $H_i(Z_\bullet^p \otimes C_\bullet^q) = Z_i^p$. The proof from above can be copied for $B_\bullet^p$ instead of $Z_\bullet^p$ to show that $H_i(B_\bullet^p \otimes C_\bullet^q) = B_i^p$. We conclude that the long exact sequence in homology is of the following form:

$$\ldots \longrightarrow B_i^p \xrightarrow{(-1)^{i+1}\iota} Z_i^p \longrightarrow H_i(C_\bullet^p \otimes C_\bullet^q) \longrightarrow B_{i-1}^p \xrightarrow{(-1)^i \iota} Z_{i-1}^p \longrightarrow \ldots$$

Since $H_i(C_\bullet^p) = Z_i^p / B_i^p = 0$ for all $i \geqslant 1$, the map from B_i^p to Z_i^p is an isomorphism. Then $H_i(C_\bullet^p \otimes C_\bullet^q)$ is enclosed by zero-maps and we obtain $H_i(C_\bullet^p \otimes C_\bullet^q) = 0$ for all $i \geqslant 2$. Furthermore, in degree zero the map $B_0^p \hookrightarrow Z_0^p$ is an inclusion. Therefore we have

$$B_1^p \xrightarrow{\sim} Z_1^p \xrightarrow{0} H_1(C_\bullet^p \otimes C_\bullet^q) \xrightarrow{0} B_0^p \hookrightarrow Z_0^p$$

and even the homology in degree 1 of $C_\bullet^p \otimes C_\bullet^q$ vanishes. □(b)

This proves the lemma. □

Now we want to prove the Eilenberg-Zilber theorem. We will do so in several steps.

Proposition 5.8. *For all simplicial complexes X and Y with total vertex orderings let*

$$\phi^{X,Y}, \psi^{X,Y} : C_\bullet(X \times Y) \longrightarrow C_\bullet(X \times Y)$$

be two chain maps. If they are natural in the sense of Remark *5.5 and if they coincide in degree* 0*, then they are chain homotopic $\phi^{X,Y} \sim \psi^{X,Y}$ for all X, Y.*

Proof. We follow the proof of [Bre97, Chapter VI, Theorem 1.3] but do it for simplicial complexes. At first, we will show the statement for all $X = \Delta^r, Y = \Delta^s$ with $r, s \in \mathbb{Z}_{\geqslant 0}$ and then we will obtain a result for arbitrary simplicial complexes by using the naturality and Remark 5.6.

Let $\phi^{X,Y}$ and $\psi^{X,Y}$ be as in the statement of the proposition. We consider the chain maps $\phi^{r,s} := \phi^{\Delta^r,\Delta^s}$ and $\psi^{r,s} := \psi^{\Delta^r,\Delta^s}$ for some $r, s \in \mathbb{Z}_{\geqslant 0}$. Inductively, we will construct a chain homotopy

$$D_k^{r,s} : C_k(\Delta^r \times \Delta^s) \longrightarrow C_{k+1}(\Delta^r \times \Delta^s)$$

such that

$$D_k^{r,s}\partial + \partial D_k^{r,s} = \phi_k^{r,s} - \psi_k^{r,s} \tag{5.1}$$

for all $k \in \mathbb{Z}_{\geqslant 0}$.

In degree 0, we define $D_0^{r,s} := 0$. Then property (5.1) holds

$$0 + 0 = \phi_0^{r,s} - \psi_0^{r,s}$$

since $\phi^{r,s}$ and $\psi^{r,s}$ coincide in degree 0 by assumption.

To define $D^{r,s}$ in higher degrees, we use induction over k. Let k be in $\mathbb{Z}_{\geqslant 1}$. We consider the map

$$\phi_k^{r,s} - \psi_k^{r,s} - D_{k-1}^{r,s}\partial \tag{5.2}$$

on k-chains, which is already defined by induction. The canonical basis of $C_k(\Delta^r \times \Delta^s)$ is the set of all k-simplices equipped with the orientation given by the total vertex ordering on $\Delta^r \times \Delta^s$. It suffices to define $D_k^{r,s}$ on the basis. Let $\sigma \in C_k(\Delta^r \times \Delta^s)$ be a basis element. If we evaluate (5.2) at σ and use the boundary map, we obtain:

$$\partial_k(\phi_k^{r,s} - \psi_k^{r,s} - D_{k-1}^{r,s}\partial_k)(\sigma) = \partial_k\phi_k^{r,s}(\sigma) - \partial_k\psi_k^{r,s}(\sigma) - \partial_k D_{k-1}^{r,s}\partial_k(\sigma) \tag{5.3}$$

If $D_{k-2}^{r,s}$ is defined, we obtain that (5.3) is 0 since $\phi^{r,s}$ and $\psi^{r,s}$ can be interchanged with the boundary map and by induction

$$\partial_k D_{k-1}^{r,s} = \phi_{k-1}^{r,s} - \psi_{k-1}^{r,s} - D_{k-2}^{r,s}\partial_{k-1}$$

holds. In the case that $D_{k-2}^{r,s}$ is not defined we have $D_{k-1}^{r,s} = D_0^{r,s} = 0$ and $\phi_0^{r,s} = \psi_0^{r,s}$ which also concludes that (5.3) vanishes. Hence, the chain $(\phi_k^{r,s} - \psi_k^{r,s} - D_{k-1}^{r,s}\partial_k)(\sigma)$ is in the kernel of ∂_k. Since the homology of $H_k(\Delta^r \times \Delta^s)$ vanishes for all $k \geqslant 1$ by Lemma 5.7, the chain is also in the image of ∂_{k+1}. There exists some $\eta \in C_{k+1}(\Delta^r \times \Delta^s)$ such that $\partial_{k+1}\eta = (\phi_k^{r,s} - \psi_k^{r,s} - D_{k-1}^{r,s}\partial_k)(\sigma)$. We define $D_k^{r,s}(\sigma) = \eta$. Then we have

$$(\partial_{k+1}D_k^{r,s} - D_{k-1}^{r,s}\partial_k)(\sigma) = (\phi_k^{r,s} - \psi_k^{r,s})(\sigma).$$

This defines $D_k^{r,s}$ on all k-chains by extending linearly.

Now let X, Y be arbitrary simplicial complexes with total vertex orderings. We will state $D^{X,Y} : C_\bullet(X \times Y) \to C_\bullet(X \times Y)$ by defining it on each basis element. Let $\sigma \in X \times Y$ be a k-simplex with orientation induced by the total vertex ordering. By Remark 5.6 there are $r, s \in \mathbb{Z}_{\geqslant 0}$, an oriented simplex $e_\sigma \in \Delta^r \times \Delta^s$ and inclusions $\iota_\sigma^0 : \Delta^r \to X, \iota_\sigma^1 : \Delta^s \to Y$ such that $(\iota_\sigma^0 \times \iota_\sigma^1)(e_\sigma) = \sigma$. We define

$$D_k^{X,Y}(\sigma) = D_k^{X,Y}((\iota_\sigma^0 \times \iota_\sigma^1)(e_\sigma)) := (\iota_\sigma^0 \times \iota_\sigma^1)D_k^{r,s}(e_\sigma).$$

By extending linearly, we obtain $D_k^{X,Y}$.

The maps $D^{X,Y} = \{D_k^{X,Y}\}_{k\in\mathbb{Z}_{\geqslant 0}}$ form indeed a chain homotopy for all simplicial complexes X and Y, which we want to check in the following. It suffices to show that we have

$$\partial D^{X,Y} + D^{X,Y}\partial = \phi^{X,Y} - \psi^{X,Y}$$

on a basis. Let $\sigma \in X \times Y$ be a k-simplex with orientation induced by the total vertex ordering. Then we can write $\sigma = (\iota_\sigma^0 \times \iota_\sigma^1)(e_\sigma)$ for the simplex and the inclusions that we used for the definition of $D^{X,Y}$. Since the inclusion is a chain map, we have $\partial(\iota_\sigma^0 \times \iota_\sigma^1) = (\iota_\sigma^0 \times \iota_\sigma^1)\partial$ and conclude

$$(\partial_{k+1}D_k^{X,Y} + D_{k-1}^{X,Y}\partial_k)(\iota_\sigma^0 \times \iota_\sigma^1)(e_\sigma) = (\iota_\sigma^0 \times \iota_\sigma^1)(\partial_{k+1}D_k^{r,s} + D_{k-1}^{r,s}\partial_k)(e_\sigma).$$

By (5.1), we have $(\partial_{k+1}D_k^{r,s} + D_{k-1}^{r,s}\partial_k)(e_\sigma) = (\phi_k^{r,s} - \psi_k^{r,s})(e_\sigma)$. We use the naturality of ϕ and ψ to obtain

$$(\iota_\sigma^0 \times \iota_\sigma^1)(\phi_k^{r,s} - \psi_k^{r,s})(e_\sigma) = (\phi_k^{X,Y} - \psi_k^{X,Y})(\iota_\sigma^0 \times \iota_\sigma^1)(e_\sigma) = (\phi_k^{X,Y} - \psi_k^{X,Y})(\sigma).$$

This concludes the proof. □

Remark 5.9. There is an analog statement for $C_\bullet(X) \otimes C_\bullet(Y)$ instead of $C_\bullet(X \times Y)$, whose proof uses the second part of Remark 5.6. We just have to do the same construction of the chain homotopy D for all $C_p(\Delta^p) \otimes C_q(\Delta^q)$ and then copy the proof with the basis elements $\eta \otimes \tau = (\iota_\eta \otimes \iota_\tau)(\Delta^p \otimes \Delta^q)$ of $C_p(X) \otimes C_q(Y)$ instead of $\sigma = (\iota^0_\sigma \times \iota^1_\sigma)(e_\sigma)$.

The proposition states that the proof of the theorem can be reduced to finding the maps $\mathcal{A}$ and $\mathcal{S}$, which we will do in the following two lemmas.

Lemma 5.10. *For all simplicial complexes X and Y with total vertex orderings there is a map*

$$\mathcal{S} = \mathcal{S}^{X,Y} : C_\bullet(X) \otimes C_\bullet(Y) \longrightarrow C_\bullet(X \times Y)$$

such that $\mathcal{S}$

(1) is natural in the sense of Remark *5.5.*

(2) is the canonical isomorphism

$$\begin{aligned} \mathcal{S}_0 : C_0(X) \otimes C_0(Y) &\xrightarrow{\sim} C_0(X \times Y) \\ ((p_0), (p_1)) &\longmapsto ((p_0, p_1)) \end{aligned}$$

in degree 0.

Proof. The proof mainly follows [Bre97, Chapter IV, Section 16 and Chapter VI, Section 1] and [Wei94, Section 6.5 and Section 8.5]. At first we want to decompose $\Delta^p \times \Delta^q$ into a sum of simplices Δ^{p+q}'s to define $\mathcal{S}$ for all $p, q \in \mathbb{Z}_{\geqslant 0}$. A tuple $t = (t_0, \dots, t_{p+q}) \subseteq \{0, \dots, p\} \times \{0, \dots, q\}$ which is ordered in both coordinates yields a simplex $s_t = \{e^{t_0}, \dots, e^{t_{p+q}}\}$ in $\Delta^p \times \Delta^q$. For the index set

$$I_{p,q} := I_{[p],[q]} := \left\{ t \subseteq \{0, \dots, p\} \times \{0, \dots, q\} \;\middle|\; \begin{array}{l} |t| = p + q + 1, \\ t \text{ ordered in both coordinates} \end{array} \right\}$$

we obtain simplices $\{s_t\}_{t \in I_{p,q}}$, which add up to $\Delta^p \times \Delta^q$. We just have to choose the correct orientation of each simplex, such that they glue together: All indices $t \in I_{p,q}$ have first entry $(0, 0)$, last entry (p, q) and in each step a value $f_i = t_i - t_{i-1} \in \{(1, 0), (0, 1)\}$ is added since $t \in [p] \times [q]$ has maximal length. We can define a permutation π_t of $\{1, \dots, p + q\}$ with $\pi_t(1) < \dots < \pi_t(p)$ and $\pi_t(p + 1) < \dots < \pi_t(p + q)$ such that

$$f_{\pi_t(i)} = \begin{cases} (1, 0) & , i \in \{1, \dots, p\} \\ (0, 1) & , i \in \{p + 1, \dots, p + q\}. \end{cases}$$

We note that these permutations are (p, q)-*shuffles* in the sense of [Wei94, Section 6.5] and describe each $t \in I_{p,q}$ uniquely. They yield the correct orientation in form of a sign $\mathsf{sign}(\pi_t)$ for each s_t with $t \in I_{p,q}$.

To define $\mathcal{S}$ it suffices to do this on the basis elements $\eta \otimes \tau \in (C_\bullet(X) \otimes C_\bullet(Y))_k$, where η is a p-simplex in X and τ a q-simplex in Y with orientation induced by the

total vertex ordering and $p+q=k$. At first we observe that we can define a chain map[2]

$$\begin{aligned}\times : C_\bullet(X)\times C_\bullet(Y) &\longrightarrow C_\bullet(X\times Y)\\ (\eta,\tau) &\longmapsto (\iota_\eta\times\iota_\tau)\left(\sum_{t\in I_{p,q}} \mathsf{sign}(\pi_t)\cdot s_t\right).\end{aligned}$$

We will only give a sketch of the proof that this is indeed a chain map. We make use of the fact that all s_t sum up to $\Delta^p\times\Delta^q$, as mentioned in REMARK 5.2. Therefore the boundary of $\sum_{t\in I_{p,q}}\mathsf{sign}(\pi_t)s_t$ is a sum of the faces of $\Delta^p\times\Delta^q$, which are the chains

$$\Delta^{[p]-\{i\}}\times\Delta^q=\sum_{t\in I_{[p]-\{i\},[q]}}\mathsf{sign}(\pi_t)s_t \qquad\text{and}\qquad \Delta^p\times\Delta^{[q]-\{j\}}=\sum_{t\in I_{[p],[q]-\{j\}}}\mathsf{sign}(\pi_t)s_t$$

with $i\in[p]$ and $j\in[q]$. By calculating which chains cancel out in $\sum_{t\in I_{p,q}}\mathsf{sign}(\pi_t)\partial s_t$ and reorder the sum one can obtain

$$\mathcal{S}\,\partial(\eta\times\tau)=(\iota_\eta\times\iota_\tau)\left(\sum_{i=0}^{p}(-1)^i\sum_{t\in I_{[p]-\{i\},[q]}}\mathsf{sign}(\pi_t)s_t+(-1)^p\sum_{j=0}^{q}(-1)^j\sum_{t\in I_{[p],[q]-\{j\}}}\mathsf{sign}(\pi_t)s_t\right)$$

which is equal to $(\iota_\eta\times\iota_\tau)(\partial\sum_{t\in I_{p,q}}\mathsf{sign}(\pi_t)s_t)=\partial\,\mathcal{S}(\eta\times\tau)$ for all $\eta\in X$ p-simplices and $\tau\in Y$ q-simplices.

Since $\times$ is bilinear, we obtain a map on the tensor product

$$\begin{aligned}\mathcal{S} : (C_\bullet(X)\otimes C_\bullet(Y))_k &\longrightarrow C_k(X\times Y)\\ (\eta\otimes\tau) &\longmapsto (\iota_\eta\times\iota_\tau)\left(\sum_{t\in I_{p,q}}\mathsf{sign}(\pi_t)s_t\right),\end{aligned}$$

which is also a chain map since $\times$ is.

Now we just have to check that both required properties (1) and (2) hold. The naturality can be proven just by using the definitions: Let $\iota_0 : X\to X'$ and $\iota_1 : Y\to Y'$ be inclusions of simplicial complexes with total vertex orderings. Then we obtain chain maps $\iota_0\times\iota_1$ and $\iota_0\otimes\iota_1$. The map $\mathcal{S}$ commutes with the inclusion:

$$\begin{aligned}\mathcal{S}(\iota_0\otimes\iota_1)(\eta\otimes\tau)&=\mathcal{S}(\iota_0(\eta)\otimes\iota_1(\tau))\\ &=(\iota_{\iota_0(\eta)}\times\iota_{\iota_1(\tau)})\left(\sum_{t\in I_{p,q}}\mathsf{sign}(\pi_t)s_t\right)\\ &=(\iota_0\times\iota_1)(\iota_\eta\times\iota_\tau)\left(\sum_{t\in I_{p,q}}\mathsf{sign}(\pi_t)s_t\right)\\ &=(\iota_0\times\iota_1)\,\mathcal{S}(\eta\otimes\tau)\end{aligned}$$

To check the second property, we consider the map $\mathcal{S}$ in degree 0. Each basis element in the tensor product $C_0(X)\otimes C_0(Y)$ is of the form $(\eta_0)\otimes(\tau_0)$ and is mapped by $\mathcal{S}$ to $((\eta_0,\tau_0))\in C_0(X\times Y)$. This is the canonical isomorphism in degree 0. □

[2] $C_\bullet(X)\times C_\bullet(Y)$ is chain complex by $(C_\bullet(X)\times C_\bullet(Y))_i=\bigoplus_{p+q=i}C_p(X)\times C_q(Y)$ with boundary map $\partial_{C_p(X)\times C_q(Y)}(\eta\times\tau)=\partial_{C_p(X)}\eta\times\tau+(-1)^p\eta\times\partial_{C_q(Y)}\tau$.

Lemma 5.11. *For all simplicial complexes X and Y with total vertex orderings there exists a chain map*

$$\mathcal{A} = \mathcal{A}^{X,Y} : C_\bullet(X \times Y) \longrightarrow C_\bullet(X) \otimes C_\bullet(Y),$$

such that $\mathcal{A}$ is

(1) a natural chain map.

(2) the canonical isomorphism

$$\mathcal{A}_0 : C_0(X \times Y) \xrightarrow{\sim} C_0(X) \otimes C_0(Y)$$

in degree 0.

Proof. We follow [Bre97, Chapter VI, Section 1]. The same trick as in the proof of Proposition 5.8 will be used here. At first we construct the chain maps $\mathcal{A}^{r,s} = \mathcal{A}^{\Delta^r,\Delta^s}$ for $r, s \in \mathbb{Z}_{\geqslant 0}$ and then we use the required naturality of $\mathcal{A}$ to define it for arbitrary complexes.

Let $r, s \in \mathbb{Z}_{\geqslant 0}$ be two non-negative integers. Inductively, we define $\mathcal{A}_k^{r,s}$: In the base case $k = 0$ the map $\mathcal{A}_0^{r,s}$ has to be the canonical isomorphism mapping a 0-simplex (e^i, e^j) to $e^i \otimes e^j$. There is no choice, since the map has to satisfy property (2) of the lemma.

Now we do the induction step. Let $k \in \mathbb{Z}_{\geqslant 1}$ be a positive integer and $\sigma \in \Delta^r \times \Delta^s$ be some k-simplex oriented by the total vertex orderings. We consider the $(k-1)$-chain $\mathcal{A}_{k-1}^{r,s}\,\partial(\sigma)$. If $k = 1$, then the boundary vanishes since the negative chain groups of a simplicial complex are all 0. For $k \geqslant 2$, we have

$$\partial\,\mathcal{A}_{k-1}^{r,s}\,\partial(\sigma) = \mathcal{A}_{k-2}^{r,s}\,\partial\partial(\sigma) = 0$$

by induction. Hence, the $(k-1)$-chain is in the kernel of the boundary map. Since the homology $H_{k-1}(C_\bullet(\Delta^r) \otimes C_\bullet(\Delta^s))$ is 0 by Lemma 5.7, the $(k-1)$-chain is also in the image of ∂_k. There exists some $\eta \in (C_\bullet(\Delta^r) \otimes C_\bullet(\Delta^s))_k$ such that

$$\partial\eta = \mathcal{A}_{k-1}^{r,s}\,\partial(\sigma)$$

and we define $\mathcal{A}_k^{r,s}(\sigma)$ to be η. In the same way, we define $\mathcal{A}_k^{r,s}$ for all basis elements $\sigma \in \Delta^r \times \Delta^s$. We obtain a well-defined map on the whole chain group by extending linearly.

Now let X and Y be arbitrary simplicial complexes with total vertex orderings. We want to define $\mathcal{A}^{X,Y}$ on the basis of each $C_k(X \times Y)$. Let $\sigma \in X \times Y$ be any k-simplex with orientation induced by the total vertex orderings. By Remark 5.6 we can find a simplex $e_\sigma \in C_k(\Delta^r \times \Delta^s)$ and an inclusion $\iota_\sigma^0 \times \iota_\sigma^1 : \Delta^r \times \Delta^s \to X \times Y$ with $\sigma = (\iota_\sigma^0 \times \iota_\sigma^1)(e_\sigma)$. We define $\mathcal{A}_k^{X,Y}(\sigma) := (\iota_\sigma^0 \otimes \iota_\sigma^1)\,\mathcal{A}_k^{r,s}(e_\sigma)$. By extending linearly, we again obtain a map $\mathcal{A}^{X,Y}$ on the whole chain complex.

It remains to check that $\mathcal{A}$ is a chain map, that it is natural and that it is the canonical isomorphism in degree 0. Let $\sigma \in X \times Y$ be any simplex oriented by the total vertex orderings. Then $\mathcal{A}$ commutes with the boundary maps since inclusions of

simplicial complexes commute with boundary maps and on $C_\bullet(\Delta^r \times \Delta^s)$ the map $\mathcal{A}$ is defined such that is commutes with the boundary map:

$$\begin{aligned}\partial\,\mathcal{A}^{X,Y}(\sigma) &= \partial(\iota_\sigma^0 \otimes \iota_\sigma^1)\,\mathcal{A}^{r,s}(e_\sigma) = (\iota_\sigma^0 \otimes \iota_\sigma^1)\partial\,\mathcal{A}^{r,s}(e_\sigma) = (\iota_\sigma^0 \otimes \iota_\sigma^1)\,\mathcal{A}^{r,s}\,\partial(e_\sigma)\\ &= \mathcal{A}^{X,Y}(\iota_\sigma^0 \times \iota_\sigma^1)\partial(e_\sigma) = \mathcal{A}^{X,Y}\,\partial(\iota_\sigma^0 \times \iota_\sigma^1)(e_\sigma) = \mathcal{A}^{X,Y}\,\partial(\sigma)\end{aligned}$$

Let $\iota_0 : X \to X'$ and $\iota_1 : Y \to Y'$ be inclusions of simplicial complexes with total vertex orderings and $\sigma \in X \times Y$ be a simplex with $(\iota_\sigma^0 \times \iota_\sigma^1)(e_\sigma) = \sigma$ and $e_\sigma \in \Delta^r \times \Delta^s$. By going through the proof of Remark 5.6, we see that it yields

$$(\iota_0 \times \iota_1)(\sigma) = ((\iota_0 \circ \iota_\sigma^0) \times (\iota_1 \circ \iota_\sigma^1))(e_\sigma)$$

as representation for $(\iota_0 \times \iota_1)(\sigma)$. Therefore, by definition we have the equation $\mathcal{A}^{X',Y'}((\iota_0 \times \iota_1)(\sigma)) = ((\iota_0 \circ \iota_\sigma^0) \times (\iota_1 \circ \iota_\sigma^1))\,\mathcal{A}^{r,s}(e_\sigma)$. We obtain:

$$\begin{aligned}\mathcal{A}^{X',Y'}(\iota_0 \times \iota_1)(\sigma) &= \mathcal{A}^{X',Y'}(\iota_0\iota_\sigma^0 \times \iota_1\iota_\sigma^1)(e_\sigma) = (\iota_0\iota_\sigma^0 \otimes \iota_1\iota_\sigma^1)\,\mathcal{A}^{r,s}(e_\sigma)\\ &= (\iota_0 \otimes \iota_1)(\iota_\sigma^0 \otimes \iota_\sigma^1)\,\mathcal{A}^{r,s}(e_\sigma) = (\iota_0 \otimes \iota_1)\,\mathcal{A}^{X,Y}(\iota_\sigma^0 \otimes \iota_\sigma^1)(e_\sigma)\\ &= (\iota_0 \times \iota_1)\,\mathcal{A}^{X,Y}(\sigma)\end{aligned}$$

The 0-simplex $((p_0, p_1)) \in C_0(X \times Y)$ is indeed mapped to $(p_0) \otimes (p_1)$ by $\mathcal{A}$:

$$\mathcal{A}_0^{X,Y}(((p_0, p_1))) = (\iota_{p_0} \otimes \iota_{p_1})\,\mathcal{A}_0^{0,0}(((e^0, e^0))) = (\iota_{p_0} \otimes \iota_{p_1})((e^0) \otimes (e^0)) = ((p_0) \otimes (p_1))$$

Hence, $\mathcal{A}$ as constructed above has all required properties. □

After all this preliminary work, the proof of Theorem 5.4 can be realized by gathering the lemmas and proposition we already have.

Proof of Theorem *5.4.* We use Lemma 5.10, Lemma 5.11 and Proposition 5.8 from above to show that

$$\begin{aligned}\mathcal{A} \circ \mathcal{S} &\sim \mathsf{id}\\ \mathcal{S} \circ \mathcal{A} &\sim \mathsf{id},\end{aligned}$$

where $\sim$ denotes that they are chain homotopic. We conclude that the induced maps of $\mathcal{A}$ and $\mathcal{S}$ on homology are isomorphisms. □

For our purposes we also need an Eilenberg-Zilber theorem but for relative homology.

Theorem 5.12. *For all simplicial complexes $X_0 \subseteq X$, $Y_0 \subseteq Y$ with total vertex orderings the maps $\mathcal{A}$ and $\mathcal{S}$ induce chain maps on the relative chain complexes*

$$C_\bullet(X \times Y, (X \times Y_0) \cup (X_0 \times Y)) \underset{\mathcal{S}}{\overset{\mathcal{A}}{\rightleftarrows}} C_\bullet(X, X_0) \otimes C_\bullet(Y, Y_0),$$

which again induce isomorphisms on homology modules.

Proof. At first we want to show that $\mathcal{A}$ and $\mathcal{S}$ are well-defined in each degree k. The expression $(C_\bullet(X, X_0) \otimes C_\bullet(Y, Y_0))_k$ on the right hand side is a direct sum of all $C_p(X, X_0) \otimes C_q(Y, Y_0)$ with $p + q = k$. We will prove that each of them can be written as a quotient:

$$\begin{aligned} C_p(X, X_0) \otimes C_q(Y, Y_0) &= \left({}^{C_p(X)}\!/_{C_p(X_0)} \right) \otimes \left({}^{C_q(X)}\!/_{C_q(X_0)} \right) \\ &= {}^{(C_p(X) \otimes C_q(Y))}\!/_{(C_p(X) \otimes C_q(Y_0) + C_p(X_0) \otimes C_q(Y))} \end{aligned}$$

If we treat the modules of the chain complexes as vector spaces, this would be true immediately since for vector spaces $V = V_1 \oplus V_0$ and $W = W_1 \oplus W_0$ we have

$$(V_1 \oplus V_0) \otimes (W_1 \otimes W_0) = (V_1 \otimes W_1) \oplus \underbrace{(V_1 \otimes W_0) \oplus (V_0 \otimes W_1) \oplus (V_0 \otimes W_0)}_{= V \otimes W_0 + V_0 \otimes W},$$

which implies

$${}^{V}\!/_{V_0} \otimes {}^{W}\!/_{W_0} = V_1 \otimes W_1 = {}^{V \otimes W}\!/_{(V \otimes W_0 + V_0 \otimes W)} \,.$$

We note that $V \otimes W_0 + V_0 \otimes W$ is not a direct sum, since $V_0 \otimes W_0$ occurs two times.

The result also holds for free modules over a commutative ring R with 1 as given in our case. We denote by $\bigoplus_X^l R$ the direct sum where we have one copy of R for each l-simplex in the simplicial complex X. Then we have $C_l(X) = \bigoplus_X^l R$ and ${}^{C_l(X)}\!/_{C_l(X_0)} = \bigoplus_{X - X_0}^l R$. We can copy the proof from above by using

$$\begin{aligned} &C_p(X) \otimes C_q(Y) \\ &= \left(\bigoplus_{X - X_0}^{p} R \otimes \bigoplus_{Y - Y_0}^{q} R \right) \oplus \underbrace{\left(\bigoplus_{X - X_0}^{p} R \otimes \bigoplus_{Y_0}^{q} R \right) \left(\oplus \bigoplus_{X_0}^{p} R \otimes \bigoplus_{Y - Y_0}^{q} R \right) \oplus \left(\bigoplus_{X_0}^{p} R \otimes \bigoplus_{Y_0}^{q} R \right)}_{\left(\oplus_X^p R \otimes \oplus_{Y_0}^q R\right) + \left(\oplus_{X_0}^p R \otimes \oplus_Y^q R\right)} . \end{aligned}$$

Furthermore, we can take direct sums of quotients of modules by using the rule $\bigoplus_i ({}^{A_i}\!/_{B_i}) \cong {}^{\bigoplus_i A_i}\!/_{\bigoplus_i B_i}$ to obtain

$$\begin{aligned} &(C_\bullet(X, X_0) \otimes C_\bullet(Y, Y_0))_k \\ &\cong {}^{(C_\bullet(X) \otimes C_\bullet(Y))_k}\!/_{(C_\bullet(X) \otimes C_\bullet(Y_0))_k + (C_\bullet(X_0) \otimes C_\bullet(Y))_k} \,. \end{aligned}$$

Now, since we can write both sides as quotients, we have to check that the restrictions map to each other, explicitly that the diagram

$$\begin{array}{ccc} C_k(X \times Y_0 \cup X_0 \times Y) & \xrightarrow{\mathcal{A}} & (C_\bullet(X) \otimes C_\bullet(Y_0))_k + (C_\bullet(X_0) \otimes C_\bullet(Y))_k \\ \downarrow & & \downarrow \\ C_k(X \times Y) & \xrightarrow{\qquad\qquad \mathcal{A} \qquad\qquad} & (C_\bullet(X) \otimes C_\bullet(Y))_k \end{array} \tag{5.4}$$

and its counterpart for $\mathcal{S}$ commute, where the vertical maps are those induced by the inclusions $Y_0 \subseteq Y$, $X_0 \subseteq X$ and $X \times Y_0 \cup X_0 \times Y \subseteq X \times Y$. If we manage to prove this, we can conclude that the maps $\mathcal{A}$ and $\mathcal{S}$ are well-defined on the quotients.

The upper left entry is a sum $C_k(X \times Y_0 \cup X_0 \times Y) = C_k(X \times Y_0) + C_k(X_0 \times Y)$, where each summand maps to one on the right side and vice versa:

$$C_k(X \times Y_0) \underset{\mathcal{S}}{\overset{\mathcal{A}}{\rightleftarrows}} (C_\bullet(X) \otimes C_\bullet(Y_0))_k$$

$$C_k(X_0 \times Y) \underset{\mathcal{S}}{\overset{\mathcal{A}}{\rightleftarrows}} (C_\bullet(X_0) \otimes C_\bullet(Y))_k$$

By using the naturality of the maps $\mathcal{A}$ and $\mathcal{S}$, we argue that the diagrams

$$\begin{array}{ccc} C_k(X \times Y_0) & \xrightarrow{\mathcal{A}} & (C_\bullet(X) \otimes C_\bullet(Y_0))_k \\ \downarrow & & \downarrow \\ C_k(X \times Y) & \xrightarrow{\mathcal{A}} & (C_\bullet(X) \otimes C_\bullet(Y))_k \end{array} \qquad \begin{array}{ccc} C_k(X_0 \times Y) & \xrightarrow{\mathcal{A}} & (C_\bullet(X_0) \otimes C_\bullet(Y))_k \\ \downarrow & & \downarrow \\ C_k(X \times Y) & \xrightarrow{\mathcal{A}} & (C_\bullet(X) \otimes C_\bullet(Y))_k \end{array}$$

and their counterparts for $\mathcal{S}$ commute. Therefore, the whole diagrams (5.4) for $\mathcal{A}$ and $\mathcal{S}$ commute.

Now, knowing that $\mathcal{A}$ and $\mathcal{S}$ are well defined in all degrees, we still need to show that $\mathcal{A}$ and $\mathcal{S}$ are chain maps. Since $\mathcal{A}$ is a chain map on $C_\bullet(X \times Y)$, we obtain

$$\partial \circ \mathcal{A}([\sigma]) = \partial([\mathcal{A}(\sigma)]) = [\partial \circ \mathcal{A}(\sigma)] = [\mathcal{A} \circ \partial(\sigma)] = \mathcal{A} \circ \partial([\sigma])$$

for all $[\sigma] \in C_k(X \times Y)/C_k(X \times Y_0 \cup X_0 \times Y)$ by using our definitions. We conclude that $\mathcal{A}$ is a chain map on the quotient. The proof for $\mathcal{S}$ works analogously.

Now we will show that $\mathcal{A}$ and $\mathcal{S}$ on the quotients induce isomorphisms in homology. In the following, we will do the proof just for $\mathcal{A}$, since for $\mathcal{S}$ we can use analog arguments. We consider the commutative diagram

$$\begin{array}{ccccccccc} 0 \to & C_\bullet(X \times Y_0 \cup X_0 \times Y) & \longrightarrow & C_\bullet(X \times Y) & \longrightarrow & C_\bullet(X \times Y, (X \times Y_0) \cup (X_0 \times Y)) & \to & 0 \\ & \downarrow \mathcal{A} & & \downarrow \mathcal{A} & & \downarrow \mathcal{A} & & \\ 0 \to & \begin{array}{c} C_\bullet(X) \otimes C_\bullet(Y_0) \\ + C_\bullet(X_0) \otimes C_\bullet(Y) \end{array} & \to & C_\bullet(X) \otimes C_\bullet(Y) & \longrightarrow & C_\bullet(X, X_0) \otimes C_\bullet(Y, Y_0) & \longrightarrow & 0, \end{array}$$

where the $\mathcal{A}$ in the middle is an isomorphism in homology by Theorem 5.4. If we can show that the left one is an isomorphism in homology, we can look at the long exact sequence in homology and deduce that all induced maps are isomorphisms by the 5-*Lemma*[3]. To use the lemma we need that the diagram in homology commutes, which is true since both horizontal sequences are the long exact sequences of quotients and all vertical maps are $\mathcal{A}$. For the proof that the left $\mathcal{A}$ is an isomorphism in homology we consider

$$\begin{array}{ccccccccc} 0 & \longrightarrow & C_\bullet(X_0 \times Y_0) & \xrightarrow{\mathrm{id} \oplus (-\mathrm{id})} & \begin{array}{c} C_\bullet(X \times Y_0) \\ \oplus C_\bullet(X_0 \times Y) \end{array} & \xrightarrow{+} & C_\bullet(X \times Y_0 \cup X_0 \times Y) & \to & 0 \\ & & \downarrow \mathcal{A} & & \downarrow \mathcal{A} \oplus \mathcal{A} & & \downarrow \mathcal{A} & & \\ 0 & \to & C_\bullet(X_0) \otimes C_\bullet(Y_0) & \xrightarrow{\mathrm{id} \oplus (-\mathrm{id})} & \begin{array}{c} C_\bullet(X) \otimes C_\bullet(Y_0) \\ \oplus C_\bullet(X_0) \otimes C_\bullet(Y) \end{array} & \xrightarrow{+} & \begin{array}{c} C_\bullet(X) \otimes C_\bullet(Y_0) \\ + C_\bullet(X_0) \otimes C_\bullet(Y) \end{array} & \longrightarrow & 0, \end{array}$$

[3]The 5-*Lemma* can be found in [Rot09, Proposition 2.72].

where the upper row is the *Mayer-Vietoris short exact sequence* as in [Hat02, Section 2.2] but for simplicial complexes and the bottom row is also a short exact sequence made by a similar construction. The left and the middle vertical map are both isomorphisms in homology by Eilenberg-Zilber. Furthermore, the diagram in homology commutes since on the rows are the same maps and the vertical maps are all $\mathcal{A}$. Again, by using the 5-*Lemma*, also the map on the right hand side is an isomorphism in homology. This is what we needed to show. □

We note that both maps $\mathcal{A}$ and $\mathcal{S}$ on quotients are natural in the sense that they commute with inclusions obtained by inclusions of pairs. This can be checked by using the naturality on the level of $C_k(X \times Y)$ and $(C_\bullet(X) \otimes C_\bullet(Y))_k$ of $\mathcal{A}$ and $\mathcal{S}$.

5.2 Mayer-Vietoris Blowup

In order to localize holes in a finite simplicial complex X, we consider a cover $\mathcal{U}$ and look at the disjoint union $X_0^{\mathcal{U}}$ of the pieces. Taking the homology of $X_0^{\mathcal{U}}$ yields local descriptions. We check which homology classes survive by connecting the pieces and computing persistent homology for this process. The *Mayer-Vietoris blowup* $X^{\mathcal{U}}$ is obtained by gluing the parts together and has the same homology as the original complex X but the parts where the subcomplexes intersect are blowed up as in Figure 5.4. In this section we mainly follow [ZC08].

At first we formalize the notion of a cover of a simplicial complex X and perform the construction of the blowup. For the rest of this chapter, let all simplicial complexes be equipped with total vertex orderings since we need this property to define products.

Definition 5.13 (Simplicial subcomplex). An *(abstract) simplicial subcomplex* of an (abstract) simplicial complex X is a subset of X, which is also a complex.

Definition 5.14 (Simplicial cover). Let X be an (abstract) simplicial complex. An *(abstract) simplicial cover* $\mathcal{U} = \{X_i\}_{i \in I}$ for any index set I is a set of subcomplexes of X such that their union $\bigcup_{i \in I} X_i$ is the whole complex X.

Definition 5.15 (Filtered Mayer-Vietoris blowup). Let X be a simplicial complex and $\mathcal{U} = \{X^i\}_{i \in [n-1]}$ be a simplicial cover, where $[n-1] = \{0, \dots, n-1\}$. For all $t \in [n-1]$ we define

$$X_t^{\mathcal{U}} := \bigcup_{\substack{J \subseteq [n-1] \\ 0 < |J| \leqslant t+1}} X^J \times \Delta^J \subseteq X \times \Delta^{n-1},$$

where $X^J := \bigcap_{j \in J} X^j$ and $\Delta^J \subseteq \Delta^{n-1}$ is the simplex with vertices induced by $j \in J$. The *Mayer-Vietoris blowup* of X and $\mathcal{U}$ is $X^{\mathcal{U}} = X_{n-1}^{\mathcal{U}}$. We call the family $\{X_t^{\mathcal{U}}\}_{t \in [n-1]}$ the *filtered Mayer-Vietoris blowup*.

We note that since $X^{\mathcal{U}}$ is contained in $X \times \Delta^{n-1}$, we have canonical projections $\pi_X : X^{\mathcal{U}} \to X$ and $\pi_\Delta : X^{\mathcal{U}} \to \Delta^{n-1}$. The projections yield chain maps given by

$$\begin{aligned} \pi_X : \quad & C_i(X \times \Delta^{n-1}) \longrightarrow C_i(X) \\ & ((x_0, \sigma_0), \dots, (x_i, \sigma_i)) \longmapsto \begin{cases} 0 & \text{, if } x_j = x_{j'} \text{ for some } j \neq j' \\ (x_0, \dots, x_i) & \text{, else} \end{cases} \end{aligned}$$

for all $i \in \mathbb{Z}_{\geqslant 0}$ and similarly for π_Δ.

Example 5.16. Let $X = \{a, b, c, d, e\}$ be a simplicial complex with a simplicial cover $\mathcal{U} = \{X^0, X^1\}$ containing the subcomplexes $X^0 = \{a, b, c, d\}$ and $X^1 = \{b, c, d, e\}$. Then $X_0^{\mathcal{U}}$ is of the form $X^0 \times \{0\} \cup X^1 \times \{1\}$ and the Mayer-Vietoris blowup is

$$X_1^{\mathcal{U}} = X_0^{\mathcal{U}} \cup \left\{ \begin{array}{l} \{(b,0),(b,1)\}, \{(c,0),(c,1)\}, \{(d,0),(d,1)\}, \{(b,0),(c,1)\}, \\ \{(c,0),(d,1)\}, \{(b,0),(b,1),(c,1)\}, \{(b,0),(c,0),(c,1)\}, \\ \{(c,0),(c,1),(d,1)\}, \{(c,0),(d,0),(d,1)\} \end{array} \right\}.$$

In Figure 5.4 we see a picture of the Mayer-Vietoris blowup $X_1^{\mathcal{U}}$ as described above.

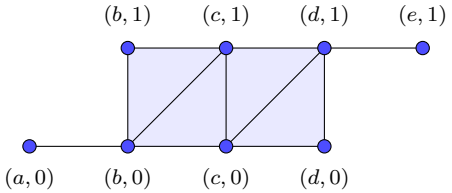


Figure 5.4: Example of a Mayer-Vietoris blowup.

For the simplicial complex $X = \{a\}$ we can define a cover $\mathcal{U} = \{X^0, X^1, X^2\}$ with $X^0 = X^1 = X^2 = \{a\}$. Then we obtain:

$$\begin{aligned} X_0^{\mathcal{U}} &= \{\{(a, e^0)\}, \{(a, e^1)\}, \{(a, e^2)\}\} \\ X_1^{\mathcal{U}} &= X_0^{\mathcal{U}} \cup \{\{(a, e^0), (a, e^1)\}, \{(a, e^0), (a, e^2)\}, \{(a, e^1), (a, e^2)\}\} \\ X_2^{\mathcal{U}} &= X_1^{\mathcal{U}} \cup \{\{(a, e^0), (a, e^1), (a, e^2)\}\} \end{aligned}$$

This filtered Mayer-Vietoris blowup is visualized in Figure 5.5.

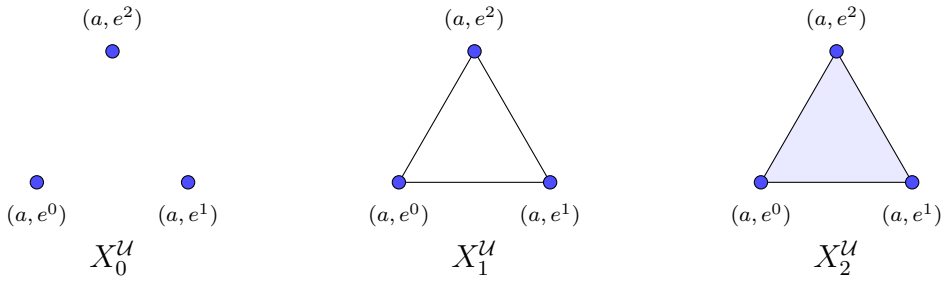


Figure 5.5: Filtered Mayer-Vietoris blowup for the cover consisting of three copies of one point.

Lemma 5.17 (Local description). *Let X be a simplicial complex with a simplicial cover $\mathcal{U} = \{X^i\}_{i \in [n-1]}$. For all $k \in \mathbb{Z}$ we have*

$$H_k(X_0^{\mathcal{U}}) \cong \bigoplus_{i \in [n-1]} H_k(X^i).$$

Proof. By the definition of the filtered Mayer-Vietoris blowup, we can easily see that

$$X_0^{\mathcal{U}} = \bigcup_{\substack{J\subseteq[n-1]\\|J|=1}} X^J \times \Delta^J = \bigcup_{j=0}^{n-1} X^j \times \{e^j\},$$

where $e^j, j \in [n-1]$ are the edges of Δ^{n-1}. This is the disjoint union of the pieces of the cover. □

Lemma 5.18 (Global description). *Let X be a simplicial complex and $\mathcal{U} = \{X^i\}_{i\in[n-1]}$ be a simplicial cover. Then for all $k \in \mathbb{Z}$ there exists an isomorphism*

$$H_k(X^{\mathcal{U}}) \cong H_k(X)$$

between the homology of $X^{\mathcal{U}}$ and X given by the chain map π_X.

Proof. We take the standard realization

$$|X^{\mathcal{U}}| = \bigcup_{\varnothing\neq J\subseteq[n-1]} |X^J \times \Delta^J| = \bigcup_{\varnothing\neq J\subseteq[n-1]} |X^J| \times |\Delta^J|.$$

In [ZC08, Section 4.1, Lemma 1] they do not give an explicit proof but refer to the case where they define a Mayer-Vietoris blowup for an open cover of the topological space $|X|$ and argue that it is homotopy equivalent to the space $|X|$ itself. The realization of a simplicial cover is in general not an open cover of $|X|$ but it should be possible to extend this proof to simplicial coverings.

Since we do not need a homotopy equivalence on the realizations, we follow a different approach. At first, we consider a simple case, where $\mathcal{U} = \{X^0, X^1\}$ is a simplicial cover consisting of two simplicial subcomplexes. The Mayer-Vietoris blowup is of the form

$$|X^{\mathcal{U}}| = |X^0 \times \{0\}| \cup |X^1 \times \{1\}| \cup |(X^0 \cap X^1) \times [0,1]| \subseteq |X| \times [0,1].$$

By using $\pi_\Delta : |X^{\mathcal{U}}| \to [0,1]$ we define $U := \pi_\Delta^{-1}([0, \frac{2}{3}))$ and $V := \pi_\Delta^{-1}((\frac{1}{3}, 1])$ which form an open cover of $|X^{\mathcal{U}}|$ as in Figure 5.6.

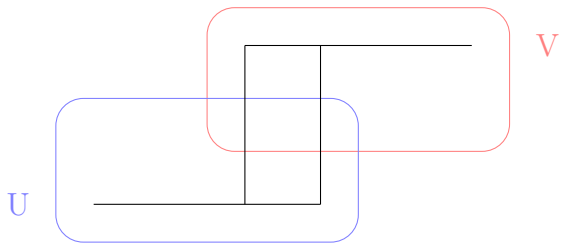


Figure 5.6: Open cover of $X^{\mathcal{U}}$ with $|\mathcal{U}| = 2$.

For the cover $\mathcal{U}$ of X we have a *Mayer-Vietoris long exact sequence* obtained by the short exact sequence

$$\begin{aligned} 0 \longrightarrow C_\bullet(X^0 \cap X^1) &\longrightarrow C_\bullet(X^0) \oplus C_\bullet(X^1) \longrightarrow C_\bullet(X) \longrightarrow 0 \\ a &\longmapsto (a, -a) \\ (a, b) &\longmapsto a + b. \end{aligned} \tag{5.5}$$

We transfer this long exact sequence to singular homology of the realizations and compare it with the *Mayer-Vietoris long exact sequences* of the cover $\{U, V\}$ of $|X^{\mathcal{U}}|$:

$$\begin{array}{ccccccccc}
\ldots & \longrightarrow & H_k(U \cap V) & \longrightarrow & H_k(U) \oplus H_k(V) & \longrightarrow & H_k(|X^{\mathcal{U}}|) & \longrightarrow & \ldots \\
 & & \downarrow{\scriptstyle (\pi_X)_*}\wr & & \downarrow{\scriptstyle (\pi_X)_* \oplus (\pi_X)_*}\wr & & \downarrow{\scriptstyle (\pi_X)_*}\wr & & \\
\ldots & \longrightarrow & H_k(|X^0| \cap |X^1|) & \longrightarrow & H_k(|X^0|) \oplus H_k(|X^1|) & \longrightarrow & H_k(|X|) & \longrightarrow & \ldots
\end{array}$$

The first two vertical maps of the diagram are isomorphisms since we find homotopy equivalences $U \cap V \simeq |X^0 \cap X^1| \times \{\frac{1}{2}\}$, $U \simeq |X^0| \times \{0\}$ and $V \simeq |X^1| \times \{1\}$ which preserve the x-coordinate by contracting linearly. Using the 5-*Lemma* yields the third isomorphism. This proves the simple case.

The remaining cases will be proven by induction. For $\mathcal{U} = \{X^0, \ldots, X^n\}$ we have

$$\pi_\Delta : |X^{\mathcal{U}}| \longrightarrow |\Delta^n|.$$

We consider the realization $|\Delta^n| = \operatorname{conv}\{0, e^1, \ldots e^n\} \subseteq \mathbb{R}^n$ and cover it by

$$U_\Delta := \left\{ (y_1, \ldots, y_n) \in |\Delta^n| \;\middle|\; \sum_i y_i < \tfrac{2}{3} \right\}$$
$$V_\Delta := \left\{ (y_1, \ldots, y_n) \in |\Delta^n| \;\middle|\; \sum_i y_i > \tfrac{1}{3} \right\}$$

as in Figure 5.7.

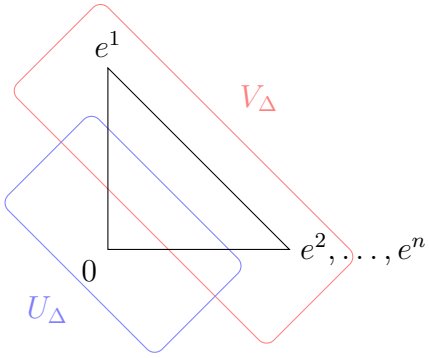


Figure 5.7: Open cover of $X^{\mathcal{U}}$ with $|\mathcal{U}| = n + 1$.

We obtain an open cover $\{U, V\}$ with $U := \pi_\Delta^{-1}(U_\Delta), V := \pi_\Delta^{-1}(V_\Delta)$ of $|X^{\mathcal{U}}|$. As before, we can find homotopy equivalent spaces

$$U \cap V \simeq S := |X^{\mathcal{U}}| \cap \left(|X| \times \left\{ (y_1, \ldots, y_n) \;\middle|\; \textstyle\sum_i y_i = \frac{1}{2} \right\} \right) = \pi_\Delta^{-1} \left(\left\{ y \;\middle|\; \textstyle\sum_i y_i = \frac{1}{2} \right\} \right)$$
$$U \simeq |X^0| \times \{0\} = \pi_\Delta^{-1}(\{0\})$$
$$V \simeq |(X^1 \cup \cdots \cup X^n)^{\{X^1, \ldots, X^n\}}| = \pi_\Delta^{-1}(\Delta^{\{1, \ldots, n\}})$$

by retracting linearly with the following homotopies:

$$\begin{aligned}
H_{U,V}: \quad U \cap V \times [0,1] &\longrightarrow U \cap V \\
((x,y),t) &\longmapsto (x, (1-t)\cdot y + t \cdot \tfrac{1}{2}\tfrac{y}{\sum_i y_i}) \\
H_U: \quad U \times [0,1] &\longrightarrow U \\
((x,y),t) &\longmapsto (x, (1-t)\cdot y + t \cdot 0) \\
H_V: \quad V \times [0,1] &\longrightarrow V \\
((x,y),t) &\longmapsto (x, (1-t)\cdot y + t \cdot \tfrac{y}{\sum_i y_i})
\end{aligned}$$

Furthermore, there is an isomorphism

$$\begin{aligned}
S &\xrightarrow{\sim} T := \pi_\Delta^{-1}(|\Delta^{\{1,\dots,n\}}|) \cap \pi_X^{-1}(|X^0|) \\
(x,y) &\longmapsto (x, 2y).
\end{aligned}$$

Since $\pi_\Delta^{-1}(\Delta^{\{1,\dots,n\}}) \cong |(X^1 \cup \dots \cup X^n)^{\{X^1,\dots,X^n\}}|$, we obtain that T is isomorphic to $|(\bigcup_{j=1}^n X^0 \cap X^j)^{\{X^0\cap X^1,\dots,X^0\cap X^1\}}|$. For better readability, we define $\widetilde{X}^1 := X^1 \cup \dots \cup X^n$. As in the simple case we want to find maps between the *Mayer-Vietoris long exact sequence* for the cover $\{U, V\}$ of $|X^{\mathcal{U}}|$ and $\{|X^0|, |\widetilde{X}^1|\}$ of $|X|$. We have the following isomorphisms

$$H_k(U \cap V) \cong H_k(S) \cong H_k(T) \xrightarrow[\sim]{(\pi_X)_*} H_k(|X^0 \cap \widetilde{X}^1|)$$

$$H_k(U) \cong H_k(|X^0| \times \{0\}) \xrightarrow[\sim]{(\pi_X)_*} H_k(|X^0|)$$

$$H_k(V) \cong H_k(|(\widetilde{X}^1)^{\{X^1,\dots,X^n\}}|) \xrightarrow[\sim]{(\pi_X)_*} H_k(|\widetilde{X}^1|)$$

where the first and the third hold by induction. Since all isomorphisms in these three rows preserve the x-coordinate, the whole composition in each row is the map $(\pi_X)_*$ and we obtain the following commutative diagram for the *Mayer-Vietoris long exact sequences*:

$$\begin{array}{ccccccccc}
\dots & \longrightarrow & H_k(U \cap V) & \longrightarrow & H_k(U) \oplus H_k(U) & \longrightarrow & H_k(|X^{\mathcal{U}}|) & \longrightarrow & \dots \\
 & & \downarrow{\scriptstyle (\pi_X)_*} & & \downarrow{\scriptstyle (\pi_X)_* \oplus (\pi_X)_*} & & \downarrow{\scriptstyle (\pi_X)_*} & & \\
\dots & \longrightarrow & H_k(|X^0| \cap |\widetilde{X}^1|) & \longrightarrow & H_k(|X^0|) \oplus H_k(|\widetilde{X}^1|) & \longrightarrow & H_k(|X|) & \longrightarrow & \dots
\end{array}$$

By the 5-*Lemma* we obtain that

$$(\pi_X)_* : H_k(|X^{\mathcal{U}}|) \xrightarrow{\sim} H_k(|X|)$$

is an isomorphism. The simplicial chain map π_X induces a map $(\pi_X)_*$ which coincides with the one from above in the sense that

$$\begin{array}{ccc} H_k(X^{\mathcal{U}}) & \xrightarrow{(\pi_X)_*} & H_k(X) \\ \downarrow\wr & & \downarrow\wr \\ H_k(|X^{\mathcal{U}}|) & \xrightarrow[\sim]{(\pi_X)_*} & H_k(|X|) \end{array}$$

commutes. This proves the lemma. □

The following lemma justifies the name of the Mayer-Vietoris blowup. The proof is an adaption from [ZC08, Section 4.1] where the authors shortly discuss the lemma in the singular instead of the simplicial case. We do not need this lemma for further arguments, therefore we give the proof only in the appendix in Section 6.3.

Lemma 5.19 (Justification of the name). *For a simplicial cover $\mathcal{U} = \{X^0, X^1\}$ of X consisting of two simplicial subcomplexes, we have an isomorphism from each homology module of the* Mayer-Vietoris long exact sequence *to the homology module of the long exact sequence for the pair $(X_1^{\mathcal{U}}, X_0^{\mathcal{U}})$ shifted by one:*

$$\begin{array}{ccccccccc} \ldots & \longrightarrow & H_i(X_0^{\mathcal{U}}) & \longrightarrow & H_i(X_1^{\mathcal{U}}) & \longrightarrow & H_i(X_1^{\mathcal{U}}, X_0^{\mathcal{U}}) & \longrightarrow & \ldots \\ & & \downarrow\wr & & \downarrow\wr & & \downarrow\wr & & \\ \ldots & \longrightarrow & H_i(X^0) \oplus H_i(X^1) & \longrightarrow & H_i(X) & \longrightarrow & H_{i-1}(X^{[1]}) & \longrightarrow & \ldots \end{array}$$

Moreover, the diagram commutes.

In Definition 5.15, we already introduced the filtration

$$\varnothing \hookrightarrow X_0^{\mathcal{U}} \hookrightarrow X_1^{\mathcal{U}} \hookrightarrow \ldots \hookrightarrow X_{n-1}^{\mathcal{U}} = X^{\mathcal{U}}.$$

Now we want to compute its persistent homology. With the tools we already discussed we would proceed as follows: We triangulate $X_t^{\mathcal{U}}$ as we did at the beginning of Section 5.1 and then compute the corresponding simplicial chain complex

$$C_\bullet(X^{\mathcal{U}})_t := C_\bullet(X_t^{\mathcal{U}}) = \sum_{\substack{J \subseteq [n-1] \\ 0 < |J| \leqslant t+1}} C_\bullet(X^J \times \Delta^J).$$

Using the persistence algorithm from the preceding chapter, we are able to compute a basis for the homology of $X^{\mathcal{U}}$. The intervals $[0, \infty)$ in the barcode are the ones that describe the homology classes of X which come from the local parts. It would be possible to proceed in this way, but computationally this procedure is very expensive.

Definition 5.20. Let X be a simplicial complex and $\mathcal{U} = \{X^i\}_{i \in [n-1]}$ a simplicial cover. We call the image of

$$\iota_* : H_k(X_0^{\mathcal{U}}) \longrightarrow H_k(X_{n-1}^{\mathcal{U}})$$

the k-th *$\mathcal{U}$-localized homology* of X.

Instead of computing persistent homology of $C_\bullet(X^{\mathcal{U}})_t$ we will consider a smaller chain complex $C^{\mathcal{U}}_\bullet(X)_t$, which provides the same homology. We follow [ZC08, Section 4.3].

Definition 5.21 (Filtered blowup chain complex). Let X be a simplicial complex and $\mathcal{U} = \{X^i\}_{i\in[n-1]}$ a simplicial cover as in Definition 5.14. For $t \in [n-1]$ we define the complex $C^{\mathcal{U}}_\bullet(X)_t \subseteq C_\bullet(X) \otimes C_\bullet(\Delta^{n-1})$ by

$$C^{\mathcal{U}}_k(X)_t := \sum_{\substack{J\subseteq[n-1]\\ 0<|J|\leq t+1}} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k$$

for all $k \in \mathbb{Z}$ with the typical boundary maps $\partial(a \otimes b) = \partial a \otimes b + (-1)^{\deg(a)} a \otimes \partial b$ for the tensor product $C_\bullet(X) \otimes C_\bullet(\Delta^{n-1})$. We call $C^{\mathcal{U}}_\bullet(X) := C^{\mathcal{U}}_\bullet(X)_{n-1}$ the *blowup chain complex* of X and $\mathcal{U}$ and the family $\{C^{\mathcal{U}}_\bullet(X)_t\}_{t\in[n-1]}$ the *filtered blowup chain complex*.

To prove that the chain complex of the filtered Mayer-Vietoris blowup and the filtered blowup chain complex yield the same persistent homology, we make use of the maps from the Eilenberg-Zilber theorem

$$C_\bullet(X^J \times \Delta^J) \underset{\mathcal{S}}{\overset{\mathcal{A}}{\rightleftarrows}} C_\bullet(X^J) \otimes C_\bullet(\Delta^J)$$

which were introduced in Theorem 5.4.

Definition 5.22. Let $\mathcal{U}$ be a cover of the simplicial complex X. For all $k \in \mathbb{Z}$ we denote by

$$C_k(X^{\mathcal{U}}) \underset{\mathcal{S}}{\overset{\mathcal{A}}{\rightleftarrows}} C^{\mathcal{U}}_k(X)$$

the maps defined by applying the maps from the Eilenberg-Zilber theorem on each summand $C_k(X^J \times \Delta^J)$ and $(C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k$ respectively.

Lemma 5.23. *Let X be a simplicial complex with a cover $\mathcal{U}$ consisting of n subcomplexes. For the maps $\mathcal{A}$ and $\mathcal{S}$ from the preceding definition, the following properties hold:*

(a) *$\mathcal{A}$ and $\mathcal{S}$ are well-defined.*

(b) *$\mathcal{A}$ and $\mathcal{S}$ are chain maps.*

(c) *We have $\mathcal{A}(C_k(X^{\mathcal{U}})_t) \subseteq C^{\mathcal{U}}_k(X)_t$ and $\mathcal{S}(C^{\mathcal{U}}_k(X)_t) \subseteq C_k(X^{\mathcal{U}})_t$ for all $k \in \mathbb{Z}$ and $t \in [n-1]$.*

Assuming that the lemma holds, we can consider the maps $\mathcal{A}_t := \mathcal{A}|_{C_\bullet(X^{\mathcal{U}})_t}$ and $\mathcal{S}_t := \mathcal{S}|_{C^{\mathcal{U}}_\bullet(X)_t}$ between the filtered blowup chain complex and the chain complex of the filtered Mayer-Vietoris blowup:

$$C_\bullet(X^{\mathcal{U}})_t \underset{\mathcal{S}_t}{\overset{\mathcal{A}_t}{\rightleftarrows}} C^{\mathcal{U}}_\bullet(X)_t$$

We recall

$$C_\bullet(X^{\mathcal{U}})_t = \sum_{\substack{J\subseteq[n-1]\\ 0<|J|\leqslant t+1}} C_\bullet(X^J\times\Delta^J) \subseteq C_\bullet(X^{\mathcal{U}}) = C_\bullet(X^{\mathcal{U}})_{n-1}$$
$$C^{\mathcal{U}}_\bullet(X)_t = \sum_{\substack{J\subseteq[n-1]\\ 0<|J|\leqslant t+1}} C_\bullet(X^J)\otimes C_\bullet(\Delta^J) \subseteq C^{\mathcal{U}}_\bullet(X) = C^{\mathcal{U}}_\bullet(X)_{n-1}$$

from Definition 5.15 and Definition 5.21.

Proof of Lemma *5.23.* At first we are going to prove (a). Let $k\in\mathbb{Z}$ be an integer. On each summand of $C^{\mathcal{U}}_k(X)$ and $C_k(X^{\mathcal{U}})$ we have the maps

$$C_k(X^J\times\Delta^J) \underset{\mathcal{S}}{\overset{\mathcal{A}}{\rightleftarrows}} (C_\bullet(X^J)\otimes C_\bullet(\Delta^J))_k$$

from Theorem 5.4. To obtain well-defined maps on the sum, we have to check that each map coincides on the intersection of two different summands. As mentioned in [ZC08] we make use of the intersection-formula for C_k and the naturality of $\mathcal{A}$. To do this for $\mathcal{S}$ too, we need its naturality and another intersection formula (5.2) for the tensor products $(C_\bullet(X^J)\otimes C_\bullet(\Delta^J))_k$.

Let $I,J\subseteq[n-1]$ be subsets with $|I|,|J|>0$. By using the intersection-formula for C_k we obtain

$$\begin{aligned} C_k(X^I\times\Delta^I)\cap C_k(X^J\times\Delta^J) &= C_k(X^I\cap X^J\times\Delta^I\cap\Delta^J)\\ &= C_k(X^{I\cup J}\times\Delta^{I\cap J}).\end{aligned}$$

In addition, we have

$$(C_\bullet(X^I)\otimes C_\bullet(\Delta^I))_k\cap(C_\bullet(X^J)\otimes C_\bullet(\Delta^J))_k = (C_\bullet(X^{I\cup J})\otimes C_\bullet(\Delta^{I\cap J}))_k.$$

The naturality of $\mathcal{A}$ and $\mathcal{S}$ yield the following commutative diagram

$$\begin{array}{ccccc}
(C_\bullet(X^I)\otimes C_\bullet(\Delta^I))_k & \xrightarrow{\mathcal{S}_I} & C_k(X^I\times\Delta^I) & \xrightarrow{\mathcal{A}_I} & (C_\bullet(X^I)\otimes C_\bullet(\Delta^I))_k\\
\uparrow{\scriptstyle\widetilde{\iota}_I} & & \uparrow{\scriptstyle\iota_I} & & \uparrow{\scriptstyle\widetilde{\iota}_I}\\
(C_\bullet(X^{I\cup J})\otimes C_\bullet(\Delta^{I\cap J}))_k & \xrightarrow{\mathcal{S}_{I,J}} & C_k(X^{I\cup J}\times\Delta^{I\cap J}) & \xrightarrow{\mathcal{A}_{I,J}} & (C_\bullet(X^{I\cup J})\otimes C_\bullet(\Delta^{I\cap J}))_k\\
\downarrow{\scriptstyle\widetilde{\iota}_J} & & \downarrow{\scriptstyle\iota_J} & & \downarrow{\scriptstyle\widetilde{\iota}_J}\\
(C_\bullet(X^J)\otimes C_\bullet(\Delta^J))_k & \xrightarrow{\mathcal{S}_J} & C_k(X^J\times\Delta^J) & \xrightarrow{\mathcal{A}_J} & (C_\bullet(X^J)\otimes C_\bullet(\Delta^J))_k
\end{array}$$

where the vertical maps are induced by inclusions. If we take for example an element σ in the intersection $C_k(X^{I\cup J}\times\Delta^{I\cap J})$ then it is mapped by $\mathcal{A}_I$ to

$$\mathcal{A}_I(\iota_I(\sigma)) = \widetilde{\iota}_I(\mathcal{A}_{I,J}(\sigma))$$

and by $\mathcal{A}_J$ to

$$\mathcal{A}_J(\iota_J(\sigma)) = \widetilde{\iota}_J(\mathcal{A}_{I,J}(\sigma)).$$

Both elements in the image coincide, since they come from the same element in the intersection. We can do this for all elements in the intersection for $\mathcal{A}$ and $\mathcal{S}$. This proves part (a). □(a)

$\mathcal{A}$ and $\mathcal{S}$ are indeed chain maps since they are on each chain group $C_k(X^J \times \Delta^J)$ or $(C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k$. We have

$$\partial\,\mathcal{A}(\sigma) = \partial\,\mathcal{A}_J(\sigma) = \mathcal{A}_J\,\partial(\sigma) = \mathcal{A}\,\partial(\sigma)$$
$$\partial\mathcal{S}(\sigma) = \partial\mathcal{S}_J(\sigma) = \mathcal{S}_J\partial(\sigma) = \mathcal{S}\partial(\sigma)$$

for the J with $\sigma \in C_\bullet(X^J \times \Delta^J)$ and $\sigma \in C_\bullet(X^J) \otimes C_\bullet(\Delta^J)$, respectively. □(b)

Also (c) holds since $C_k(X^{\mathcal{U}})_t$ and $C_k^{\mathcal{U}}(X)_t$ for $t \in [n-1]$ and $k \in \mathbb{Z}$ are sums over the same indices and $\mathcal{A}$ and $\mathcal{S}$ map summands with the same indices to each other. We have the inclusions $\mathcal{A}(C_k(X^{\mathcal{U}})_t) \subseteq C_k^{\mathcal{U}}(X)_t$ and $\mathcal{S}(C_k^{\mathcal{U}}(X)_t) \subseteq C_k(X^{\mathcal{U}})_t$. □(c)

This proves the lemma. □

In the following, we will show that the maps

$$C_\bullet(X^{\mathcal{U}})_t \underset{\mathcal{S}_t}{\overset{\mathcal{A}_t}{\rightleftarrows}} C_\bullet^{\mathcal{U}}(X)_t$$

as constructed above induce an isomorphism in homology. In this case the barcodes for the two chain complexes coincide.

Theorem 5.24. *Let X be a simplicial complex and $\mathcal{U} = \{X^i\}_{i\in[n-1]}$ be a simplicial cover of X. The chain map*

$$\mathcal{S} : C_\bullet^{\mathcal{U}}(X) \longrightarrow C_\bullet(X^{\mathcal{U}})$$

and all restrictions $\mathcal{S}_t$ induce isomorphisms in homology

$$(\mathcal{S}_t)_* : H_k(C_\bullet^{\mathcal{U}}(X)_t) \xrightarrow{\sim} H_k(C_\bullet(X^{\mathcal{U}})_t)$$

for all $k \in \mathbb{Z}$ and $t \in [n-1]$. Furthermore, the diagram

$$\begin{array}{ccc} H_k(C_\bullet^{\mathcal{U}}(X)_t) & \xrightarrow[\sim]{(\mathcal{S}_t)_*} & H_k(C_\bullet(X^{\mathcal{U}})_t) \\ \downarrow & & \downarrow \\ H_k(C_\bullet^{\mathcal{U}}(X)_{t'}) & \xrightarrow[\sim]{(\mathcal{S}_{t'})_*} & H_k(C_\bullet(X^{\mathcal{U}})_{t'}) \end{array} \tag{5.6}$$

commutes for all $k \in \mathbb{Z}$ and $t \leqslant t' \in [n-1]$.

The theorem also holds for the maps $\mathcal{A}_t$ instead of $\mathcal{S}_t$ as stated in [ZC08, Theorem 5]. If we assume that the theorem holds, we obtain

Corollary 5.25. *Consider the coefficients for homology to be a field. For the conditions as in* Theorem *5.24 the barcodes of $\{C_\bullet(X^{\mathcal{U}})_t\}_{t\in[n-1]}$ and $\{C_\bullet^{\mathcal{U}}(X)_t\}_{t\in[n-1]}$ coincide.*

Proof. Both filtrations are directed spaces in the sense of Section 3.2. The maps $(\mathcal{S}_t)_*$ for $t \in [n-1]$ form an isomorphism of directed spaces because of (5.6). □

Proof of THEOREM *5.24.* We already constructed the chain map $\mathcal{S}$ and it remains to check that it fulfills the desired properties. Diagram (5.6) indeed commutes since for all $t \leqslant t' \in [n-1]$

$$\begin{array}{ccc} C_\bullet^{\mathcal{U}}(X)_t & \xrightarrow{\mathcal{S}_t} & C_\bullet(X^{\mathcal{U}})_t \\ \big\downarrow & & \big\downarrow \\ C_\bullet^{\mathcal{U}}(X)_{t'} & \xrightarrow{\mathcal{S}_{t'}} & C_\bullet(X^{\mathcal{U}})_{t'} \end{array}$$

is a commutative diagram in the category of chain complexes of modules and H_k is a functor by [Wei94, Section 1.2] for all $k \in \mathbb{Z}$. Therefore, it suffices to show that the induced maps $(\mathcal{S}_t)_*$ on homology are isomorphisms for all $t \in [n-1]$. We will prove this by induction over t.

For $t = 0$ we have

$$\begin{aligned} C_k(X^{\mathcal{U}})_0 &= \sum_{\substack{J \subseteq [n-1] \\ |J|=1}} C_k(X^J \times \Delta^J) = \sum_{j=0}^{n-1} C_k(X^j \times \{e^j\}) \\ &\cong \bigoplus_{j=0}^{n-1} C_k(X^j) \cong \sum_{j=0}^{n-1} (C_\bullet(X^j) \otimes C_\bullet(\{e^j\}))_k, \end{aligned}$$

where the e^j for $j \in [n-1]$ are distinct points. We note that $C_q(\{e^j\})$ is the coefficient ring of the module for $q = 0$ and otherwise 0. Hence, $C_k(X^{\mathcal{U}})_0$ is isomorphic to $C_k^{\mathcal{U}}(X)_0$ by the canonical isomorphism $\mathcal{S}_0$.

Now we perform the induction step and assume that the maps $\mathcal{S}_0, \ldots, \mathcal{S}_{t-1}$ induce isomorphisms in homology. We consider the following short exact sequences of chain complexes

$$\begin{array}{ccccccccc} 0 & \longrightarrow & C_\bullet^{\mathcal{U}}(X)_{t-1} & \longrightarrow & C_\bullet^{\mathcal{U}}(X)_t & \longrightarrow & C_\bullet^{\mathcal{U}}(X)_t / C_\bullet^{\mathcal{U}}(X)_{t-1} & \longrightarrow & 0 \\ & & \big\downarrow{\scriptstyle \mathcal{S}_{t-1}} & & \big\downarrow{\scriptstyle \mathcal{S}_t} & & \big\downarrow{\scriptstyle \widehat{\mathcal{S}}_t} & & \\ 0 & \longrightarrow & C_\bullet(X^{\mathcal{U}})_{t-1} & \longrightarrow & C_\bullet(X^{\mathcal{U}})_t & \longrightarrow & C_\bullet(X^{\mathcal{U}})_t / C_\bullet(X^{\mathcal{U}})_{t-1} & \longrightarrow & 0 \end{array}$$

with maps between them, where $\widehat{\mathcal{S}}_t$ is the map induced by $\mathcal{S}$ on the quotient[4]. $\widehat{\mathcal{S}}_t$ is a well-defined chain map since $\mathcal{S}_t$ and $\mathcal{S}_{t-1}$ are well-defined chain maps. We obtain long exact sequences in homology with maps between them by the *Snake Lemma*:

$$\begin{array}{ccccccccc} \ldots & \longrightarrow & H_i\left(C_\bullet^{\mathcal{U}}(X)_{t-1}\right) & \longrightarrow & H_i\left(C_\bullet^{\mathcal{U}}(X)_t\right) & \longrightarrow & H_i\left(C_\bullet^{\mathcal{U}}(X)_t / C_\bullet^{\mathcal{U}}(X)_{t-1}\right) & \longrightarrow & \ldots \\ & & \big\downarrow{\scriptstyle (\mathcal{S}_{t-1})_*} & & \big\downarrow{\scriptstyle (\mathcal{S}_t)_*} & & \big\downarrow{\scriptstyle (\widehat{\mathcal{S}}_t)_*} & & \\ \ldots & \longrightarrow & H_i\left(C_\bullet(X^{\mathcal{U}})_{t-1}\right) & \longrightarrow & H_i\left(C_\bullet(X^{\mathcal{U}})_t\right) & \longrightarrow & H_i\left(C_\bullet(X^{\mathcal{U}})_t / C_\bullet(X^{\mathcal{U}})_{t-1}\right) & \longrightarrow & \ldots \end{array}$$

[4] We have $C_\bullet^{\mathcal{U}}(X)_t / C_\bullet^{\mathcal{U}}(X)_{t-1} = \ldots \xrightarrow{\partial} C_i^{\mathcal{U}}(X)_t / C_i^{\mathcal{U}}(X)_{t-1} \xrightarrow{\partial} C_{i-1}^{\mathcal{U}}(X)_t / C_{i-1}^{\mathcal{U}}(X)_{t-1} \xrightarrow{\partial} \ldots$ and an analog sequence for $C_\bullet(X^{\mathcal{U}})_t / C_\bullet(X^{\mathcal{U}})_{t-1}$.

The diagram commutes since the maps on the rows are equal and all vertical maps are $\mathcal{S}$. We want to prove that $(\mathcal{S}_t)_*$ is an isomorphism by using the 5-*Lemma*. The map $(\mathcal{S}_{t-1})_*$ is an isomorphism for all $i \in \mathbb{Z}$ by induction. To show that $(\widehat{\mathcal{S}}_t)_*$ is also an isomorphism we need to do some work. We will find chain complexes which are isomorphic to the domain and codomain of $\widehat{\mathcal{S}}_t$ respectively:

$$\begin{array}{ccc} C_k^{\mathcal{U}}(X)_t/C_k^{\mathcal{U}}(X)_{t-1} & \xrightarrow{\widehat{\mathcal{S}}_t} & C_k(X^{\mathcal{U}})_t/C_k(X^{\mathcal{U}})_{t-1} \\ \wr \downarrow [\sigma\otimes\eta] & [\sigma\otimes\eta] \mapsto [\mathcal{S}(\sigma\times\eta)] & [\sigma\times\eta] \uparrow \wr \\ \bigoplus_{\substack{J\subseteq[n-1]\\|J|=t+1}} (C_\bullet(X^J)\otimes C_\bullet(\Delta^J,\partial\Delta^J))_k & \xrightarrow{f} & \bigoplus_{\substack{J\subseteq[n-1]\\|J|=t+1}} C_k(X^J\times\Delta^J, X^J\times\partial\Delta^J) \end{array} \tag{5.7}$$

If we assume that we already constructed the isomorphism and take f as $\mathcal{S}$ from the Eilenberg-Zilber theorem for the quotient, then the diagram commutes. By using Theorem 5.12 with $X = X^J$, $Y = \Delta^J$, $X_0 = \varnothing$ and $Y_0 = \partial\Delta^J$ for each J individually, we conclude that f induces an isomorphism in homology. The vertical maps are isomorphisms and also yield an isomorphism in homology. Therefore, $\widehat{\mathcal{S}}_t$ has to be an isomorphism in homology, too. In the remaining part of the proof we will construct the vertical isomorphisms of diagram (5.7).

We start with the vertical map on the right side:

$$\begin{aligned} C_k(X^{\mathcal{U}})_t/C_k(X^{\mathcal{U}})_{t-1} \quad &\cong \quad \bigoplus_{\substack{J\subseteq[n-1]\\|J|=t+1}} C_k(X^J\times\Delta^J, X^J\times\partial\Delta^J) \\ [\sigma\times\eta] \quad &\longmapsto\!\shortmid \quad [\sigma\times\eta] \end{aligned}$$

In the following we denote by $I(t)$ the set of all $J \subseteq [n-1]$ with $0 < |J| \leqslant t$ for the sake of better readability. We divide the construction of the isomorphism into two steps: Let $\{J_1, \ldots, J_m\}$ be the set $\{J \subseteq [n-1] \mid |J| = t+1\}$. In the first step we show that the sum in the bottom row of

$$\begin{aligned} C_k(X^{\mathcal{U}})_t/C_k(X^{\mathcal{U}})_{t-1} &= \sum_{J\in I(t+1)} C_k(X^J\times\Delta^J)/\sum_{J\in I(t)} C_k(X^J\times\Delta^J) \\ &= \sum_{i=1}^{m} \left((C_k(X^{J_i}\times\Delta^{J_i}) + \sum_{J\in I(t)} C_k(X^J\times\Delta^J))/\sum_{J\in I(t)} C_k(X^J\times\Delta^J) \right) \end{aligned} \tag{5.8}$$

is a direct sum. This can be shown by proving that

$$C_k(X^{J_i}\times\Delta^{J_i}) \cap \sum_{j\in\{1,\ldots,m\}-\{i\}} C_k(X^{J_j}\times\Delta^{J_j}) \overset{!}{\subseteq} \sum_{J\in I(t)} C_k(X^J\times\Delta^J)$$

for all $i \in \{1, \ldots, m\}$. It suffices to prove the inclusion for $i = 1$ since we can reorder

$J_1, \dots, J_m$. We obtain

$$\begin{aligned} C_k(X^{J_1} \times \Delta^{J_1}) \cap \sum_{j=2}^{m} C_k(X^{J_j} \times \Delta^{J_j}) &= C_k(X^{J_1} \times \Delta^{J_1}) \cap C_k(\bigcup_{j=2}^{m} X^{J_j} \times \Delta^{J_j}) \\ &= C_k(X^{J_1} \times \Delta^{J_1} \cap \bigcup_{j=2}^{m} X^{J_j} \times \Delta^{J_j}) \\ &= C_k(\bigcup_{j=2}^{m} X^{J_1 \cup J_j} \times \Delta^{J_1 \cap J_j}) \\ &= \sum_{j=2}^{m} C_k(X^{J_1 \cup J_j} \times \Delta^{J_1 \cap J_j}) =: (*). \end{aligned}$$

There are inclusions $X^{J_1 \cup J_j} \subseteq X^{J_1 \cap J_j}$ for all $j \in \{2, \dots, m\}$. Therefore, we have

$$C_k(X^{J_1 \cup J_j} \times \Delta^{J_1 \cap J_j}) \subseteq C_k(X^{J_1 \cap J_j} \times \Delta^{J_1 \cap J_j}).$$

Since $|J_1 \cap J_j| \leqslant t$ for all $j \in \{2, \dots, m\}$, we obtain

$$(*) \subseteq \sum_{j=2}^{m} C_k(X^{J_1 \cap J_j} \times \Delta^{J_1 \cap J_j}) \subseteq \sum_{J \in I(t)} C_k(X^J \times \Delta^J).$$

In the second step we will find an isomorphism for each summand

$$Q_i := (C_k(X^{J_i} \times \Delta^{J_i}) + \sum_{|J| \leqslant t} C_k(X^J \times \Delta^J)) \Big/ \sum_{|J| \leqslant t} C_k(X^J \times \Delta^J)$$

with $i \in \{1, \dots, m\}$ from (5.8):

$$\begin{aligned} Q_i \quad &\cong \quad C_k(X^{J_i} \times \Delta^{J_i}) \Big/ C_k(X^{J_i} \times \partial\Delta^{J_i}) \\ [\eta \times \sigma] \quad &\longleftarrow\!\shortmid \quad [\eta \times \sigma] \end{aligned}$$

Let i be some element in $\{1, \dots, m\}$. We use $(A + B)/B \cong A/(A \cap B)$ for the modules $A = C_k(X^{J_i} \times \Delta^{J_i})$ and $B = \sum_{J \in I(t)} C_k(X^J \times \Delta^J)$. Then it remains to show that

$$C_k(X^{J_i} \times \Delta^{J_i}) \cap \sum_{J \in I(t)} C_k(X^J \times \Delta^J) \stackrel{!}{=} C_k(X^{J_i} \times \partial\Delta^{J_i}).$$

The inclusion "$\supseteq$" holds since

$$C_k(X^{J_i} \times \partial\Delta^{J_i}) = \sum_{\emptyset \neq J \subsetneq J_i} C_k(X^{J_i} \times \Delta^J) = \sum_{\emptyset \neq J \subsetneq J_i} C_k(X^J \times \Delta^J)$$

and $|J_i| = t + 1$. We prove the other inclusion "$\subseteq$" by looking at

$$\begin{aligned} C_k(X^{J_i} \times \Delta^{J_i}) \cap \sum_{J \in I(t)} C_k(X^J \times \Delta^J) &= C_k(X^{J_i} \times \Delta^{J_i}) \cap C_k(\bigcup_{J \in I(t)} X^J \times \Delta^J) \\ &= C_k(X^{J_i} \times \Delta^{J_i} \cap \bigcup_{J \in I(t)} X^J \times \Delta^J) \\ &= C_k(\bigcup_{\substack{J \in I(t) \\ J_i \cap J \neq \emptyset}} X^{J_i \cup J} \times \Delta^{J_i \cap J}). \end{aligned}$$

For all $J \in I(t)$ with $J_i \cap J \neq \varnothing$ we define $\widetilde{J} := J_i \cap J$. It satisfies $\widetilde{J} \in I(t)$, $\widetilde{J} \subseteq J_i$ and the inclusion $X^{J_i \cup J} \times \Delta^{J_i \cap J} \subseteq X^{J_i \cup \widetilde{J}} \times \Delta^{J_i \cap \widetilde{J}}$ holds. Therefore, we obtain

$$C_k(\bigcup_{\substack{J \in I(t) \\ J_i \cap J \neq \varnothing}} X^{J_i \cup J} \times \Delta^{J_i \cap J}) \subseteq C_k(\bigcup_{\substack{\widetilde{J} \in I(t) \\ \widetilde{J} \subseteq J_i}} X^{J_i \cup \widetilde{J}} \times \Delta^{J_i \cap \widetilde{J}}) = C_k(\bigcup_{\substack{J \in I(t) \\ J \subseteq J_i}} X^{J_i \cup J} \times \Delta^{J_i \cap J}).$$

Since $J_i \cup J = J_i$ and $J_i \cap J = J$ for all $J \subseteq J_i$, we have

$$X^{J_i \cup J} \times \Delta^{J_i \cap J} = X^{J_i} \times \Delta^J.$$

The set of all $J \subseteq J_i$ with $0 < |J| \leqslant t$ is the set of all $\varnothing \neq J \subsetneq J_i$ since $|J_i| = t + 1$. Hence, we obtain

$$\begin{aligned} C_k(\bigcup_{\substack{J \in I(t) \\ J \subseteq J_i}} X^{J_i} \times \Delta^J) &= C_k(X^{J_i} \times \bigcup_{\varnothing \neq J \subsetneq J_i} \Delta^J) \\ &= C_k(X^{J_i} \times \partial\Delta^{J_i}) \end{aligned}$$

by using that $\partial\Delta^{J_i} = \bigcup_{|J|=|J_i|-1} \Delta^J = \bigcup_{\varnothing \neq J \subsetneq J_i} \Delta^J$. This concludes the proof of the second step.

To show that the isomorphism is even an isomorphism of simplicial chain complexes we need that the map commutes with the boundary operator ∂. But the map and ∂ are defined on the quotient by its counterparts on the representatives where the map is the identity, therefore this holds.

Now we consider the domain of $\widehat{\mathcal{S}}_t$ and aim to obtain the following isomorphism:

$$\begin{aligned} C_\bullet^{\mathcal{U}}(X)_t / C_\bullet^{\mathcal{U}}(X)_{t-1} \quad &\cong \quad \bigoplus_{\substack{J \subseteq [n-1] \\ |J| = t+1}} C_\bullet(X^J) \otimes C_\bullet(\Delta^J, \partial\Delta^J) \\ [\sigma \times \eta] \quad &\longmapsto \quad [\sigma \times \eta] \end{aligned}$$

We use similar arguments as in the proof for the codomain but write them down differently. Comparing the bases of chain complexes as we will do it here was done before by the rules $C_\bullet(X) \cap C_\bullet(Y) = C_\bullet(X \cap Y)$ and $C_\bullet(X) + C_\bullet(Y) = C_\bullet(X \cup Y)$. Now we are working with tensor products and we cannot use these rules anymore. We again proceed in two steps. In the first step we want to show that the quotient

$$\sum_{i=1}^{m} \left(((C_\bullet(X^{J_i}) \otimes C_\bullet(\Delta^{J_i}))_k + \sum_{J \in I(t)} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k) / \sum_{J \in I(t)} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k \right) \tag{5.9}$$

which is equal to $C_k^{\mathcal{U}}(X)_t / C_k^{\mathcal{U}}(X)_{t-1}$ is a direct sum. As before, to prove this it remains to show that

$$(C_\bullet(X^{J_1}) \otimes C_\bullet(\Delta^{J_1}))_k \cap \sum_{i=2}^{m} (C_\bullet(X^{J_i}) \otimes C_\bullet(\Delta^{J_i}))_k \overset{!}{\subseteq} \sum_{J \in I(t)} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k.$$

The modules $A := (C_\bullet(X^{J_1}) \otimes C_\bullet(\Delta^{J_1}))_k$ and $B := \sum_{i=2}^m (C_\bullet(X^{J_i}) \otimes C_\bullet(\Delta^{J_i}))_k$ are contained in $(C_\bullet(X) \otimes C_\bullet(\Delta^{n-1}))_k$, which is a free module with basis

$$\mathcal{B} = \left\{\sigma \otimes \eta \;\middle|\; \exists p, q \in \mathbb{Z}_{\geqslant 0} : p + q = k, \sigma \in X \text{ } p\text{-simplex}, \eta \in \Delta^{n-1} \text{ } q\text{-simplex}\right\}$$

where the orientations of the simplices are induced by the total orderings on X and Δ^{n-1}. Moreover, we have bases

$$\begin{aligned}\mathcal{B}_A &= \left\{\sigma \otimes \eta \in \mathcal{B} \;\middle|\; \sigma \in X^{J_1}, \eta \in \Delta^{J_1}\right\}\\ \mathcal{B}_B &= \left\{\sigma \otimes \eta \in \mathcal{B} \;\middle|\; \exists i \in \{2, \dots, m\} : \sigma \in X^{J_i}, \eta \in \Delta^{J_i}\right\}\end{aligned}$$

for A and B, which are both contained in $\mathcal{B}$. Hence, the intersection of A and B has the basis

$$\begin{aligned}\mathcal{B}_{A\cap B} &= \left\{\sigma \otimes \eta \in \mathcal{B} \;\middle|\; \exists i \in \{2, \dots, m\} : \sigma \in X^{J_i} \cap X^{J_1}, \eta \in \Delta^{J_i} \cap \Delta^{J_1}\right\}\\ &= \mathcal{B}_A \cap \mathcal{B}_B\,.\end{aligned}$$

We have the inclusion $X^{J_i} \cap X^{J_1} = X^{J_i \cup J_1} \subseteq X^{J_i \cap J_1}$ for all $i \in \{2, \dots, m\}$. Since $\Delta^{J_i} \cap \Delta^{J_1} = \Delta^{J_i \cap J_1}$ and $|J_i \cap J_1| \leqslant t$ holds, the basis $\mathcal{B}_{A\cap B}$ is contained in

$$\sum_{J \in I(t)} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k.$$

Hence, also $A \cap B$ is contained in $\sum_{J\in I(t)} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k$.

In the second step we want to prove that there are isomorphisms

$$\begin{aligned}&((C_\bullet(X^{J_i}) \otimes C_\bullet(\Delta^{J_i}))_k + \sum_{J\in I(t)} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k)\Big/ \sum_{J\in I(t)} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k\\ &\cong \left(C_\bullet(X^{J_i}) \otimes \left(C_\bullet(\Delta^{J_i})\Big/ C_\bullet(\partial\Delta^{J_i})\right)\right)_k\end{aligned}$$

for each summand of (5.9). If we show that for each pair (p, q) with $p + q = k$ there is an isomorphism

$$\begin{aligned}&(C_p(X^{J_i}) \otimes C_q(\Delta^{J_i}) + \textstyle\sum_{J\in I(t)} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k)\Big/ \sum_{J\in I(t)} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k\\ &\cong (C_p(X^{J_i}) \otimes C_q(\Delta^{J_i}))\Big/ (C_p(X^{J_i}) \otimes C_q(\partial\Delta^{J_i}))\\ &\cong C_p(X^{J_i}) \otimes \left(C_q(\Delta^{J_i})\Big/ C_q(\partial\Delta^{J_i})\right),\end{aligned}$$

then they sum up to an isomorphism like above. It suffices to show that

$$\underbrace{C_p(X^{J_i}) \otimes C_q(\Delta^{J_i})}_{=:A'} \cap \underbrace{\sum_{J\in I(t)} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k}_{=:B'} \overset{!}{=} \underbrace{C_p(X^{J_i}) \otimes C_q(\partial\Delta^{J_i})}_{=:C'}.$$

We have the inclusion $C' \subseteq A'$ since $C_q(\partial\Delta^{J_i}) \subseteq C_q(\Delta^{J_i})$. Furthermore, we know that $C_q(\partial\Delta^{J_i}) = \sum_{\emptyset\neq J\subsetneq J_i} C_q(\Delta^J)$. Hence, we obtain

$$\begin{aligned}C' = C_p(X^{J_i}) \otimes C_q(\partial\Delta^{J_i}) &= \sum_{\emptyset\neq J\subsetneq J_i} C_p(X^{J_i}) \otimes C_q(\Delta^J)\\ &\subseteq \sum_{\emptyset\neq J\subsetneq J_i} C_p(X^J) \otimes C_q(\Delta^J) \subseteq B'\end{aligned}$$

and the inclusion "$\supseteq$" is proven. To show that also the inclusion "$\subseteq$" holds, we need to compare bases as in the first step. We have the bases

$$\begin{aligned}
\mathcal{B}_{A'} &= \left\{\sigma \otimes \eta \in \mathcal{B} \;\middle|\; \sigma \in X^{J_i} \text{ } p\text{-simplex and } \eta \in \Delta^{J_i} \text{ } q\text{-simplex}\right\} \\
\mathcal{B}_{B'} &= \left\{\sigma \otimes \eta \in \mathcal{B} \;\middle|\; \exists J \in I(t) : \sigma \in X^{J} \text{ and } \eta \in \Delta^{J}\right\} \\
\mathcal{B}_{A' \cap B'} &= \left\{\sigma \otimes \eta \in \mathcal{B} \;\middle|\; \exists J \in I(t) : \sigma \in X^{J} \cap X^{J_i} \text{ } p\text{-simplex and } \eta \in \Delta^{J} \cap \Delta^{J_i} \text{ } q\text{-simplex}\right\}
\end{aligned}$$

and we want to show that $\mathcal{B}_{A' \cap B'}$ is contained in C'. We use that for all $J \in I(t)$ with $J \cap J_i \neq \varnothing$ we have $X^J \cap X^{J_i} = X^{J \cup J_i} \subseteq X^{J_i}$ and $\Delta^J \cap \Delta^{J_i} = \Delta^{J \cap J_i}$. Moreover, we can rewrite the set $\{J \cap J_i \neq \varnothing \mid J \in I(t)\}$ as $\{\widetilde{J} \mid \varnothing \neq \widetilde{J} \subsetneq J_i\}$ since $|J_i| = t + 1$. By using $\bigcup_{\varnothing \neq \widetilde{J} \subsetneq J_i} \Delta^{\widetilde{J}} = \partial \Delta^{J_i}$ this yields

$$\begin{aligned}
\mathcal{B}_{A' \cap B'} &\subseteq \{\sigma \otimes \eta \in \mathcal{B} \mid \exists \varnothing \neq \widetilde{J} \subsetneq J_i : \sigma \in X^{J_i} \text{ } p\text{-simplex and } \eta \in \Delta^{\widetilde{J}} \text{ } q\text{-simplex}\} \\
&= \{\sigma \otimes \eta \in \mathcal{B} \mid \sigma \in X^{J_i} \text{ } p\text{-simplex and } \eta \in \partial \Delta^{J_i} \text{ } q\text{-simplex } \}.
\end{aligned}$$

But this is a basis for C'. We obtain $\mathcal{B}_{A' \cap B'} \subseteq C'$ and therefore $A' \cap B' \subseteq C'$ as desired. □

5.3 The Localization Algorithm

To execute the localization algorithm we need a finite simplicial complex X together with a simplicial cover $\mathcal{U} = \{X^i\}_{i \in [n-1]}$. As already mentioned in this chapter, we should keep in mind that the construction of the Mayer-Vietoris blowup requires a total vertex ordering of X. We can compute persistent homology of the corresponding filtered Mayer-Vietoris blowup $\{X^{\mathcal{U}}_t\}_{t \in [n-1]}$ by Algorithm 1 from Chapter 4. This yields a description of the localized homology of X, represented by the intervals from 0 to ∞ in the barcode. By considering all classes that persist until the end, we obtain a full description of the homology of $X^{\mathcal{U}} \simeq X$.

As we have seen in the preceding section, instead of looking at the chains $C_\bullet(X^{\mathcal{U}})_t$ of the filtered Mayer-Vietoris blowup we can consider the chains $C^{\mathcal{U}}_\bullet(X)_t$, which yield the same homology. To use the persistence algorithm we already implemented, we just need to specify a basis for $C^{\mathcal{U}}_\bullet(X)$ which can be restricted to a basis for $C^{\mathcal{U}}_\bullet(X)_t$ for all $t \in [n-1]$ and understand how the boundary map for this basis looks like.

Lemma 5.26 (Basis for $C^{\mathcal{U}}_k(X)$). *The set*

$$\mathcal{B}_k := \{\sigma \otimes \Delta^J \mid \varnothing \neq J \subseteq [n-1], \sigma \in X^J, \dim \sigma + \dim \Delta^J = k\}$$

where the orientation of the simplices is induced by the total orderings on X and Δ^{n-1} is a basis for the chain module $C^{\mathcal{U}}_k(X)$ for all $k \in \mathbb{Z}_{\geqslant 0}$.

Proof. Let $k \in \mathbb{Z}_{\geqslant 0}$ be some non-negative integer We already know that the filtered blowup chain complex

$$C^{\mathcal{U}}_k(X) \subseteq (C_\bullet(X) \otimes C_\bullet(\Delta^{n-1}))_k$$

is included in the tensor product of two chain complexes. The basis for this tensor product is

$$\widetilde{\mathcal{B}}_k = \{\sigma \otimes \Delta^I \mid \sigma \in X, \dim \sigma + \dim \Delta^I = k\}$$

with orientations induced by the total orderings of X and Δ^{n-1}. The set

$$\widetilde{\mathcal{B}}_k^J = \{\sigma \otimes \Delta^I \in \widetilde{\mathcal{B}}_k \mid \sigma \in X^J, \Delta^I \subseteq \Delta^J\}$$

is a basis for the module $(C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k$. Hence, the basis for the chain module $C_k^{\mathcal{U}}(X) = \sum_{\substack{J\subseteq[n-1]\\ 0<|J|}} (C_\bullet(X^J) \otimes C_\bullet(\Delta^J))_k$ is

$$\mathcal{B}_k = \bigcup_{\varnothing\neq J\subseteq[n-1]} \widetilde{\mathcal{B}}_k^J = \bigcup_{\varnothing\neq J\subseteq[n-1]} \{\sigma \otimes \Delta^I \in \widetilde{\mathcal{B}}_k \mid \sigma \in X^J, I \subseteq J\}.$$

If $\sigma \otimes \Delta^I$ is in the part of the union for index J, then it is also in every part for index $J' \supseteq J$. By removing duplicates, we obtain

$$\begin{aligned}\mathcal{B}_k &= \bigcup_{\varnothing\neq J\subseteq[n-1]} \{\sigma \otimes \Delta^J \in \widetilde{\mathcal{B}}_k \mid \sigma \in X^J\} \\ &= \{\sigma \otimes \Delta^J \mid \sigma \in X^J, \dim\sigma + \dim\Delta^J = k, \varnothing \neq J \subseteq [n-1]\}\end{aligned}$$

with orientations induced by the total vertex orderings as stated in the lemma. □

Remark 5.27. Analogously, for all $k \in \mathbb{Z}_{\geqslant 0}$ and $t \in [n-1]$ we obtain a basis

$$\mathcal{B}_k^t := \{\sigma \otimes \Delta^J \in \mathcal{B}_k \mid |J| \leqslant t+1\}$$

of $C_k^{\mathcal{U}}(X)_t$.

The boundary map on $C_k^{\mathcal{U}}(X)$ is the standard boundary map on tensor products from [Hat02, Section 3.B]:

Lemma 5.28. *Let $\sigma \otimes \Delta^J \in \mathcal{B}_k$ be a basis element of $C_k^{\mathcal{U}}(X)$ with $k \in \mathbb{Z}_{\geqslant 0}$. Then the boundary map is given by*

$$\begin{aligned}\partial(\sigma \otimes \Delta^J) &= \partial\sigma \otimes \Delta^J + (-1)^{\dim\sigma}\sigma \otimes \partial\Delta^J \\ &= \sum_{i=0}^{\dim\sigma} (-1)^i \widehat{\sigma}_i \otimes \Delta^J + (-1)^{\dim\sigma} \sum_{j=0}^{\dim\Delta^J} (-1)^j \sigma \otimes \widehat{\Delta}_j^J,\end{aligned}$$

where $\widehat{(\cdot)}_j$ denotes that the j-th vertex is deleted from the sequence.

Proof. We use the typical definition of the boundary map for tensor products in the first equation. Then we evaluate the map on each entry. □

Now we discuss how to implement the construction of the filtered blowup chain complex. As in Chapter 4 we use coefficients in $\mathbb{F}_2$ such that we do not need to care about the orientation of the simplices. We consider the filtered blowup chain complex of the simplicial complex X and its cover $\mathcal{U}$. Its basis elements $\sigma \otimes \Delta^J$ with $\sigma \in X$ and $\varnothing \neq J \subseteq [n-1]$ are stored as tuples (σ, J). For each of those tuples we create a cell by computing its boundary as in Lemma 5.28 and dimension $\dim((\sigma, J)) := \dim(\sigma) + \dim(\Delta^J) = \dim(\sigma) + |J| - 1$ and add it to a list K. To each cell (σ, J) in K we store its order $|J| - 1$ in the sequence $\{C_\bullet^{\mathcal{U}}(X)_t\}_{t\in[n-1]}$. We note

that by this implementation the cell (σ, J) is in $C_k^{\mathcal{U}}(X)_t$ if and only if $\dim((\sigma, J)) = k$ and $\operatorname{ord}((\sigma, J)) \leqslant t$.

Now we order the elements in the list K by their order and if they have the same order, we order them by their dimension. By doing this, we obtain a well-defined filtration of complexes, where in each step one cell is added. Then we can use the persistent homology algorithm from Chapter 4 to get the barcode.

We are interested in those cells whose corresponding basis element persists in homology until the end of the sequence. Those are the cells that do not have a partner. Since the second step of the persistence algorithm changes only the cells which have a partner with a higher index, it is irrelevant for the localization algorithm. Hence, we can even use Algorithm 3.

The implementation to construct the filtered blowup chain complex can be found in *blowup.py*. It uses the class tuple from *tuple.py* to solve the technical problem that two lists whose entries coincide are not equal by default in Python3. The code can be found in the appendix in Section 6.4. It is also available at [Gün19], where the reader in addition can find a simple examples for the computation of localized homology in the files *Example_Localization1.py* and *Example_Localization2.py*.

To interpret the results, we want to transfer the homology classes of $H_k(C_\bullet^{\mathcal{U}}(X))$ with $k \in \mathbb{Z}$ obtained by the persistence algorithm to $H_k(X)$. Let $[\sigma] \in H_k(C_\bullet^{\mathcal{U}}(X)_t)$ with $t \in [n-1]$ be some homology class obtained from the algorithm which is mapped by the induced map

$$H_k(C_\bullet^{\mathcal{U}}(X)_t) \xrightarrow{\iota *} H_k(C_\bullet^{\mathcal{U}}(X)_{n-1}) = H_k(C_\bullet^{\mathcal{U}}(X))$$

of the inclusion $\iota : C_\bullet^{\mathcal{U}}(X)_t \to C^{\mathcal{U}}(X)_{n-1}$ to $[\sigma] \neq 0 \in H_k(C_\bullet^{\mathcal{U}}(X))$. From Theorem 5.24 we know the map

$$\begin{aligned} \mathcal{S}_* : H_k(C_\bullet^{\mathcal{U}}(X)) &\xrightarrow{\sim} H_k(C_\bullet(X^{\mathcal{U}})) \\ [\sigma] &\longmapsto [\mathcal{S}(\sigma)]. \end{aligned}$$

It preserves the persistence structure in the sense of (5.6). From Lemma 5.18 we know that the chain map $\pi_X : C_k(X^{\mathcal{U}}) \to C_k(X)$ yields an isomorphism in homology. This gives us the corresponding basis element in the homology of $H_k(X)$:

$$\begin{aligned} (\pi_X)_* : H_k(C_\bullet(X^{\mathcal{U}})) &\xrightarrow{\sim} H_k(C_\bullet(X)) \\ [\mathcal{S}(\sigma)] &\longmapsto [\pi_X(\mathcal{S}(\sigma))] \end{aligned}$$

We are interested in how $\pi_X(\mathcal{S}(\sigma))$ looks like on the chain-level. By Lemma 5.26 the module $C_k^{\mathcal{U}}(X)$ has the basis

$$\mathcal{B}_k := \left\{ \eta \otimes \Delta^J \;\middle|\; \varnothing \neq J \subseteq [n-1], \eta \in X^J, \dim \eta + \dim \Delta^J = k \right\}.$$

It suffices to describe $\pi_X \circ \mathcal{S}$ for elements of the basis $\mathcal{B}_k$. Let $\eta \otimes \Delta^J \in \mathcal{B}_k$ be a basis element with $\eta = (\eta_0, \ldots, \eta_p)$, $\Delta^J = (e^{j_0}, \ldots, e^{j_q})$ and $p + q = k$. In particular, η is a p-simplex and Δ^J is a q-simplex. Let $I_{p,q}$ be the set of all indices $t \subseteq \{0, \ldots, p\} \times \{0, \ldots, q\}$

which are ordered in both coordinates and satisfy $|t| = p+q+1$. Then $\mathcal{S}$ maps $\eta \otimes \Delta^J$ to

$$(\iota_\eta \times \iota_{\Delta^J})(\sum_{t \in I_{p,q}} \mathsf{sign}(\pi_t) s_t)$$

by its construction in LEMMA 5.10. If q is at least 1, then we consider some element $t = ((r_0, r_0'), \ldots, (r_k, r_k'))$ in $I_{p,q}$. We have

$$(\iota_\eta \times \iota_{\Delta^J}(s_t)) = ((\eta_{r_0}, e^{r_0'}), \ldots, (\eta_{r_k}, e^{r_k'})).$$

At least two indices r_i, r_{i+1} in $r_0, \ldots, r_k$ coincide. Hence, also η_{r_i} and $\eta_{r_{i+1}}$ coincide and π_X maps $\iota_\eta \times_{\Delta^J} (s_t)$ to 0. Since t was chosen arbitrarily, π_X also maps $\mathcal{S}(\eta \otimes \Delta^J)$ to 0. If $q = 0$, then $\sum_{t \in I_{p,q}} \mathsf{sign}(\pi_t) s_t = s_{((0,0),\ldots,(k,0))}$. Therefore,

$$\mathcal{S}(\eta \otimes \Delta^J) = ((\eta_0, e^{j_0}), \ldots, (\eta_k, e^{j_0}))$$

and $\pi_X(\mathcal{S}(\eta \times \Delta^J)) = (\eta_0, \ldots, \eta_k) = \eta$.

We conclude that we can interpret our results from the algorithm by writing each chain representing an interval $[t, \infty)$ for some t in $\mathbb{Z}_{\geqslant 0}$ as linear combination of basis elements $\eta \otimes \Delta^J$ and project them to their first entry η if $|J| = 0$ and map them to 0 for $|J| > 0$.

We will prove in the following lemma that those cells obtained from the algorithm whose corresponding basis element persists form order 0 up to the end yield homology classes of X which are in one component $C_k(X^j)$ with $k \in \mathbb{Z}, j \in [n-1]$. Therefore, this procedure in fact provides an improved basis of the homology classes of X as described at the beginning of this chapter.

LEMMA 5.29. *Let X be a simplicial complex and $\mathcal{U} = \{X^i\}_{i \in [n-1]}$ be a cover consisting of n subcomplexes. We consider the filtered blowup chain complex $\{C_\bullet^{\mathcal{U}}(X)_t\}_{t \in [n-1]}$. The basis elements obtained by Algorithm 1 or Algorithm 3 representing a class in the homology-sequence that persist from degree 0 until the end are chains in one component $C_\bullet(X^j) \otimes C_\bullet(\{e^j\})$ with $j \in [n-1]$ of $C_\bullet^{\mathcal{U}}(X)_0$.*

Proof. This can be shown by induction over the index of the elements in the cell list K used in Algorithm 1 and analogously for Algorithm 3.

The cell with index 0 in the list K is of the form $\{p\} \otimes \{e^j\} \in K$ with $j \in [n-1]$ and $p \in X$ a 0-simplex. Therefore, step 1 terminates directly. In Algorithm 3 basisel($\{p\} \otimes \{e^j\}$) can not be modified again. By PROPERTIES 4.6 (3) and (5), in Algorithm 1 the basis element of $\{p\} \otimes \{e^j\}$ can only be changed again if the cell has a partner with a higher index. In this case the corresponding basis element in the homology-sequence would not persist from degree 0 until the end. Therefore, $\{p\} \otimes \{e^j\}$ is in $C_\bullet(X^j) \otimes C_\bullet(\{e^j\})$.

Let $\sigma \otimes \{e^j\} \in C_\bullet^{\mathcal{U}}(X)_0$ be the cell in K which has the index $i > 0$. We assume that all cells with an index which is lower than i are chains in just one component. During step 1 of the iteration $\sigma \otimes \{e^j\}$ of the for-loop the basis element of partner$(\tau) \otimes \{e^{j'}\}$ can be added to the basis element of $\sigma \otimes \{e^j\}$. The basis element of partner$(\tau) \otimes \{e^{j'}\}$ is of the form $\eta \otimes \{e^{j'}\}$ for $\eta \in C_\bullet(X^j)$ by induction. By the definition of partner$(\tau) \otimes \{e^{j'}\}$, its basis element and the basis element of $\sigma \otimes \{e^j\}$ have a coinciding element $\tau \otimes \{e^{j'}\}$ in their boundary. Therefore, we obtain $j = j'$. The new basis element obtained by adding the basis element of partner$(\tau) \otimes \{e^j\}$ to the basis element of $\sigma \otimes \{e^j\}$ still

has only $\{e^j\}$ as its second entry. If the corresponding basis element of $\sigma \otimes \{e^j\}$ after step 1 persists until the end in the homology-sequence, then it can not be modified by the algorithm once again by the same reason as in the base case. Hence, it is in the component $C_\bullet(X^j) \otimes C_\bullet(\{e^j\})$. □

6 Appendix

6.1 Experiment – Naive Approach

These are the results of the experiment discussed in Section 4.6. We use a MacBook Air 2016 with an Intel Core i5, 1.8 GHz processor. The evenly distributed points are implemented in $\mathbb{R}^2$ by using the classes and functions of

points_to_complex.py

which can be found in the next section or at [Gün19]. We construct complexes for these points by Vietoris-Rips and Čech. Then we use functions from

homology.py

to compute their homology and draw their barcodes. We test up to an radius of 1.1 and all classes that still persist at this point are labeled with a red line at the end of the interval. The cases for up to 3 points are trivial and we exclude them from our observations. For 6 and more points interesting behavior can be observed. All files mentioned in the remaining part of this section can be found at [Gün19].

At first we consider the Vietoris-Rips complexes. The file

Example_points_on_circle_VR_numeric.py

is used for the construction of the following barcodes. For 9 points the barcodes are portrayed in Figure 6.1.

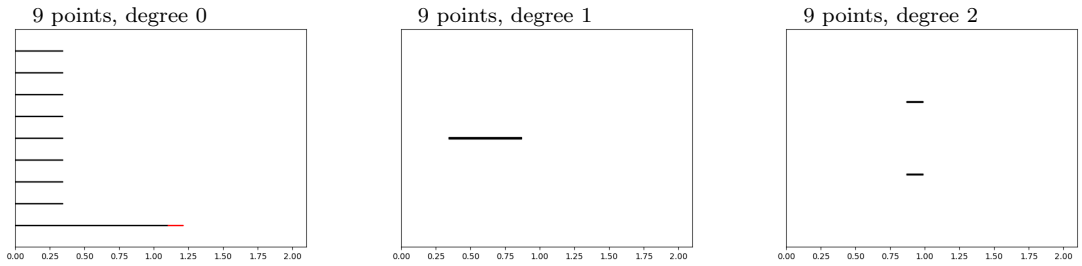


Figure 6.1: Barcodes of the Vietoris-Rips complex for 9 points in degree 0, 1 and 2.

For up to 12 points we always see the homology of distinct points, the 1-sphere and one point in degree 0 and 1 as in the figure for 9 points. The other results are displayed in Figure 6.2.

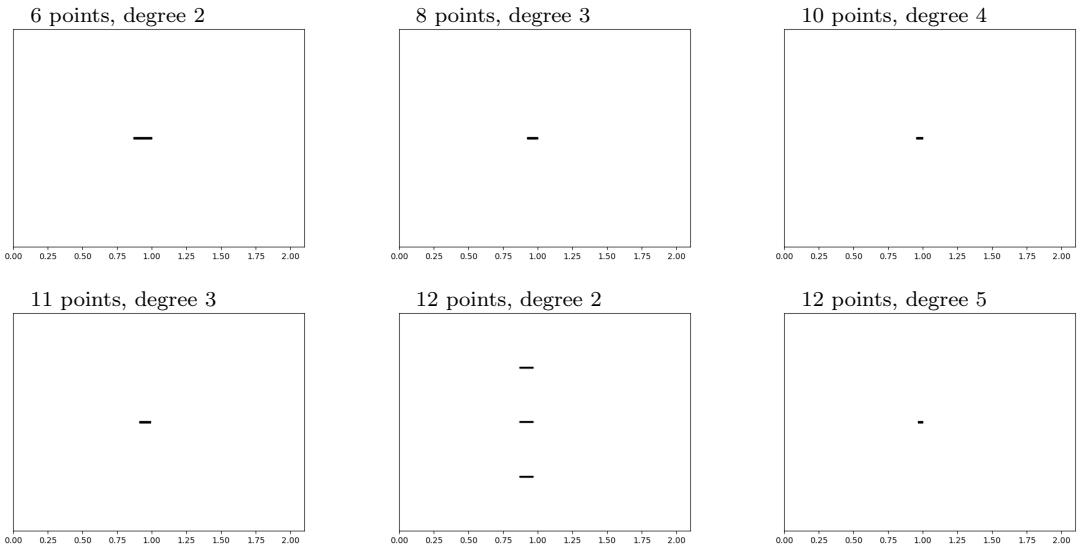


Figure 6.2: More barcodes obtained for Vietoris-Rips complexes.

Now we consider the barcodes obtained by using the Čech complex. They can be reconstructed by using the file

$$Example\_points\_on\_circle\_CECH\_numeric.py.$$

For each cell we start with the radius of the Vietoris-Rips complex and increase it incrementally by the value

$$precision = 0.001$$

until the radius satisfies the condition for the Čech complex. For up to 5 points the behavior of the barcodes is not really interesting since they are again just composed by the intervals representing the points and the 1-sphere. If we consider barcodes for the Čech complexes of at least 6 points we see intervals which exceed the 1, for example the three small intervals in degree 4 in Figure 6.3. Moreover, there are many other small intervals in the barcodes which indicate numerical errors. The biggest numerical errors can be found in degree 1 as seen in Figure 6.4.

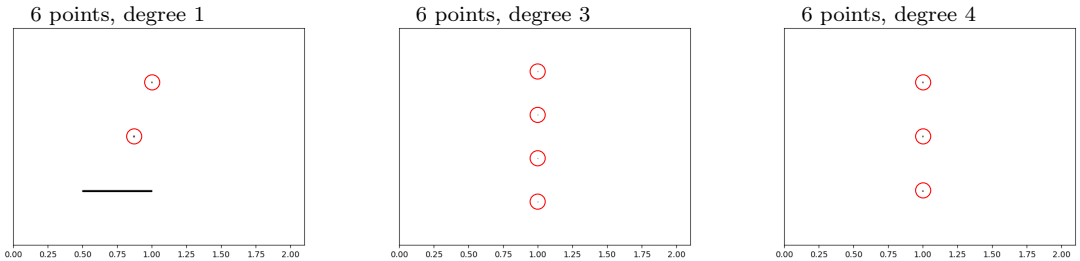


Figure 6.3: Barcodes of Čech complex for 6 points in degree 1, 3 and 4.

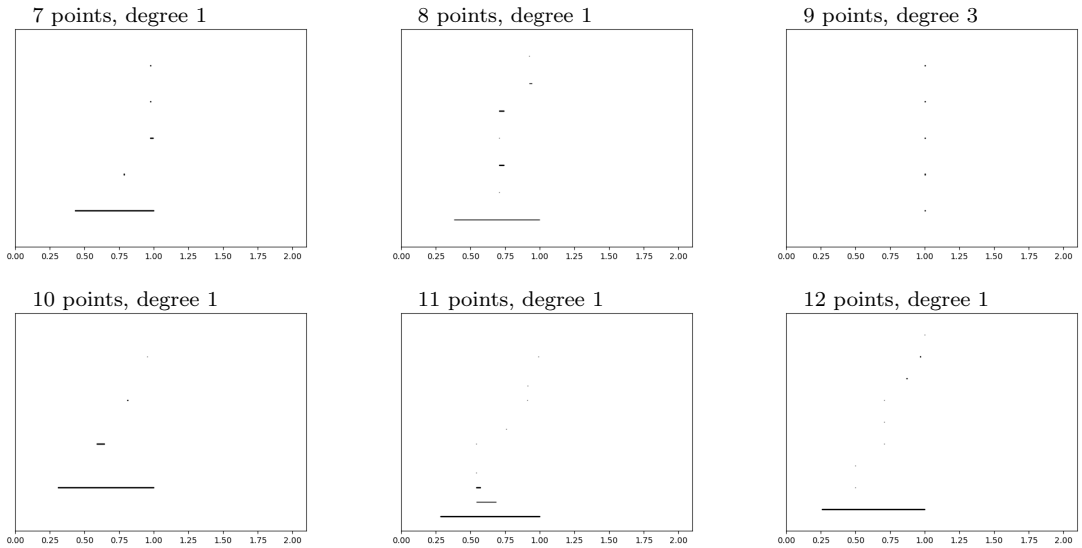


Figure 6.4: More barcodes of Čech complexes with errors.

6.2 Experiment – More Precise Approach

If we implement the simplicial cells immediately and omit the representation of points in $\mathbb{R}^2$ as described at the end of Section 4.6, we obtain the following results. The Čech complex yields just barcodes consisting of intervals representing the distinct points, a circle and one point. There are no barcodes in higher degrees. This can be tested by the reader by executing the file

Example_points_on_circle_CECH_improved.py.

The barcodes for the Vietoris-Rips complexes are more interesting. They can be computed by using the functions from the file

Example_points_on_circle_CECH_improved.py.

We used

Experiment_VR.py

to start the algorithm and store the barcodes in a .txt file automatically. The results can be found in Table 6.1 for up to 16 points. We did not include the barcodes for the zeroth or first degree since there we always see the homology of the points, of the 1-sphere and of one point. The column *time 1* denotes the time to create the cells and *time 2* denotes the time to compute the persistent homology. The number in front of each interval describes how often this interval appears in the barcode.

# points	# cells	interesting barcodes	time 1	time 2
6	63	deg 2: $1 \cdot [0.8660, 1.0)$	0.00s	0.00s
7	127	–	0.02s	0.00s
8	255	deg 3: $1 \cdot [0.9238, 1.0)$	0.12s	0.01s
9	511	deg 2: $2 \cdot [0.8660, 0.9848)$	0.41s	0.02s
10	1023	deg 4: $1 \cdot [0.9510, 1.0)$	1.85s	0.04s
11	2047	deg 3: $1 \cdot [0.9096, 0.9898)$	7.78s	0.09s
12	4095	deg 2: $3 \cdot [0.8660, 0.9659)$	35.63s	0.34s
		deg 5: $1 \cdot [0.9659, 1.0)$		
13	8191	deg 3: $1 \cdot [0.9350, 0.9927)$	153.49s	0.80s
14	16383	deg 3: $1 \cdot [0.9010, 0.9749)$	654.06s	2.97s
		deg 6: $1 \cdot [0.9749, 1.0)$		
15	32767	deg 2: $4 \cdot [0.8660, 0.9511)$	3068.06s	7.91s
		deg 4: $2 \cdot [0.9511, 0.9945)$		
16	65535	deg 3: $1 \cdot [0.9239, 0.9808)$	12559.12s	26.86s
		deg 7: $1 \cdot [0.9808, 1.0)$		

Table 6.1: The results of the improved approach to compute barcodes for the Vietoris-Rips complexes.

In the table we see equally many interesting barcodes in odd degrees as in even degrees. The main part of the computation time is used for the construction of the simplicial complex. An improvement there could enable us to compute further barcodes quickly and obtain new insights. Two possible approaches would be to use multiprocessing or a better sorting algorithm to speed up the assignment of boundaries.

6.3 Justification of the Name of the Mayer-Vietoris Blowup

In the following we give the proof of Lemma 5.19 from Section 5.2:

Lemma (Justification of the name). *For a simplicial cover $\mathcal{U} = \{X^0, X^1\}$ of X consisting of two simplicial subcomplexes, we have an isomorphism from each homology module of the* Mayer-Vietoris long exact sequence *to the homology module of the long exact sequence for the pair $(X_1^{\mathcal{U}}, X_0^{\mathcal{U}})$ shifted by one:*

$$\begin{array}{ccccccccc}
\dots & \longrightarrow & H_i(X_0^{\mathcal{U}}) & \longrightarrow & H_i(X_1^{\mathcal{U}}) & \longrightarrow & H_i(X_1^{\mathcal{U}}, X_0^{\mathcal{U}}) & \longrightarrow & \dots \\
 & & \downarrow\wr & & \downarrow\wr & & \downarrow\wr & & \\
\dots & \longrightarrow & H_i(X^0) \oplus H_i(X^1) & \longrightarrow & H_i(X) & \longrightarrow & H_{i-1}(X^{[1]}) & \longrightarrow & \dots
\end{array}$$

Moreover, the diagram commutes.

Proof. For better readabilty, we denote $U = X^0$ and $V = X^1$. It holds $X = U \cup V$, $X_0^{\mathcal{U}} = U \times \{0\} \cup V \times \{1\}$ and $X_1^{\mathcal{U}} = X_0^{\mathcal{U}} \cup (U \cap V) \times [0, 1]$. We have the short exact

sequences of simplicial chain complexes

$$0 \longrightarrow C_\bullet(X_0^{\mathcal{U}}) \longrightarrow C_\bullet(X_1^{\mathcal{U}}) \longrightarrow {}^{C_\bullet(X_1^{\mathcal{U}})}\!/_{C_\bullet(X_0^{\mathcal{U}})} \longrightarrow 0$$

$$0 \longrightarrow C_\bullet(U \cap V) \longrightarrow C_\bullet(C) \oplus C_\bullet(V) \longrightarrow C_\bullet(U \cup V) \longrightarrow 0$$

where the first one is the short exact sequence for the pair $(X_1^{\mathcal{U}}, X_0^{\mathcal{U}})$ given by the canonical inclusion and projection and the second one is the *Mayer-Vietoris short exact sequence*. The induced long exact sequence of the pair $(X_1^{\mathcal{U}}, X_0^{\mathcal{U}})$ is given by the maps

$$\begin{array}{ccccccccc} \ldots \longrightarrow & H_i(X_0^{\mathcal{U}}) & \longrightarrow & H_i(X_1^{\mathcal{U}}) & \longrightarrow & H_i(X_1^{\mathcal{U}}, X_0^{\mathcal{U}}) & \longrightarrow & H_{i-1}(X_0^{\mathcal{U}}) & \longrightarrow \ldots \\ & [\sigma] & \longmapsto & [\sigma] & & [\sigma] & \longmapsto & [\partial\sigma]. & \\ & & & [\sigma] & \longmapsto & [\sigma] & & & \end{array}$$

The long exact sequence for *Mayer-Vietoris* is

$$\begin{array}{ccccccccc} \ldots \longrightarrow & H_i(U \cap V) & \longrightarrow & H_i(U) \oplus H_i(V) & \longrightarrow & H_i(X) & \xrightarrow{\delta} & H_{i-1}(U \cap V) & \longrightarrow \ldots \\ & [\sigma] & \longmapsto & ([\sigma], [-\sigma]) & & & & & \\ & & & ([\tau], [\eta]) & \longmapsto & [\tau + \eta], & & & \end{array}$$

where $\delta : H_i(X) \to H_{i-1}(U \cap V)$ maps $[\sigma]$ with $\sigma = \gamma + \gamma'$, $\gamma \in C_i(U), \gamma' \in C_i(V)$ to $[\partial\gamma]$. This can be checked by going through the proof of the *Snake Lemma* as in [HS97, Chapter III, Lemma 5.1]. We have an isomorphism

$$\begin{array}{ccccc} H_i(X_0^{\mathcal{U}}) & \xrightarrow{\sim} & H_i(U \times \{0\}) \oplus H_i(V \times \{1\}) & \xrightarrow{\sim} & H_i(U) \oplus H_i(V) \\ [\sigma] = [\mathrm{pr}_0(\sigma) + \mathrm{pr}_1(\sigma)] & \longmapsto & ([\mathrm{pr}_0(\sigma)], [\mathrm{pr}_1(\sigma)]) & \longmapsto & ([\pi_X(\mathrm{pr}_0(\sigma))], [\pi_X(\mathrm{pr}_1(\sigma))]), \end{array}$$

where the chain $\mathrm{pr}_j(\sigma)$ is the part of σ which lies in $X_0^{\mathcal{U}} \cap (X \times \{j\})$. The map π_X is the chain map defined at the beginning of Section 5.1. Furthermore, we have the isomorphism

$$\begin{array}{ccc} H_i(X_1^{\mathcal{U}}) & \xrightarrow{\sim} & H_i(X) \\ [\sigma] & \longmapsto & [\pi_X(\sigma)] \end{array}$$

by Lemma 5.18. For the setting

$$\begin{array}{ccccccccccc} \ldots \longrightarrow & H_i(X_0^{\mathcal{U}}) & \longrightarrow & H_i(X_1^{\mathcal{U}}) & \longrightarrow & H_i(X_1^{\mathcal{U}}, X_1^{\mathcal{U}}) & \longrightarrow & H_{i-1}(X_0^{\mathcal{U}}) & \longrightarrow \ldots \\ & \downarrow & \text{(A)} & \downarrow & \text{(B)} & \vdots\, f_i & \text{(C)} & \downarrow & \\ \ldots \longrightarrow & \begin{array}{c} H_i(U) \\ \oplus H_i(V) \end{array} & \longrightarrow & H_i(X) & \longrightarrow & H_{i-1}(U \cap V) & \longrightarrow & \begin{array}{c} H_{i-1}(U) \\ \oplus H_{i-1}(V) \end{array} & \longrightarrow \ldots \end{array}$$

we want to construct a map f_i such that the whole diagram commutes.

In the following we will use *Excision* as in [Hat02, Theorem 2.20] to construct the map. By [Hat02, Theorem 2.27], we have an isomorphism between the relative homology of simplicial complexes and their realizations:

$$H_i(X_1^{\mathcal{U}}, X_0^{\mathcal{U}}) \xrightarrow{\sim} H_i(|X_1^{\mathcal{U}}|, |X_0^{\mathcal{U}}|)$$

Let $W = U \cap V$ be the intersection of U and V. We want to use *Excision* to obtain

$$H_i(|X_1^{\mathcal{U}}|, |X_0^{\mathcal{U}}|) \xleftarrow[\sim]{(\subseteq)_*} H_i(|W| \times [0,1], |W| \times \{0,1\}) \tag{6.1}$$

but we cannot use it directly since the open sets $|X_0^{\mathcal{U}}|^o$ and $(|W| \times [0,1])^o$ do not cover $|X_1^{\mathcal{U}}|$ in general. We consider the homotopy

$$\begin{aligned} H : [0,1] \times [0,1] &\longrightarrow [0,1] \\ (s,t) &\longmapsto s + \mathsf{sign}(s - \tfrac{1}{2}) \cdot t \cdot \min\{s, |s - \tfrac{1}{2}|, 1 - s\}. \end{aligned}$$

It yields isomorphisms on homology

$$\begin{aligned} H_i(|X_1^{\mathcal{U}}|, |X_1^{\mathcal{U}}| \cap X \times ([0, \tfrac{1}{4}] \cup [\tfrac{3}{4}, 1])) &\xrightarrow[\sim]{(\mathsf{id} \times H(\cdot,1))_*} H_i(|X_1^{\mathcal{U}}|, |X_0^{\mathcal{U}}|) \\ H_i(|W| \times [0,1], |W| \times ([0, \tfrac{1}{4}] \cup [\tfrac{3}{4}, 1])) &\xrightarrow[\sim]{(\mathsf{id} \times H(\cdot,1))_*} H_i(|W| \times [0,1], |W| \times \{0,1\}) \end{aligned}$$

which can be proven by the 5-*Lemma* of the long exact sequences of pairs with maps $(\mathsf{id} \times H(\cdot, 1))_*$ between them. Now we can use *Excision* and obtain the commutative diagram

$$\begin{array}{ccc} H_i(|W| \times [0,1], |W| \times \{0,1\}) & \xrightarrow{\qquad (\subseteq)_* \qquad} & H_i(|X_1^{\mathcal{U}}|, |X_0^{\mathcal{U}}|) \\ {\scriptstyle (\mathsf{id} \times H(\cdot,1))_*}\uparrow\wr & & {\scriptstyle (\mathsf{id} \times H(\cdot,1))_*}\uparrow\wr \\ H_i(|W| \times [0,1], |W| \times ([0, \tfrac{1}{4}] \cup [\tfrac{3}{4}, 1])) & \xrightarrow[\sim]{(\subseteq)_*} & H_i(|X_1^{\mathcal{U}}|, |X_1^{\mathcal{U}}| \cap X \times ([0, \tfrac{1}{4}] \cup [\tfrac{3}{4}, 1])). \end{array}$$

Hence, even (6.1) is an isomorphism as desired.

Now we consider the long exact sequence of the pair $(|W| \times [0,1], |W| \times \{0,1\})$. Since $h_i : H_i(|W| \times \{0,1\}) = H_i(|W| \times \{0\}) \oplus H_i(|W| \times \{1\}) \to H_i(|W| \times [0,1])$ is the inclusion and $|W| \times [0,1]$ is homotopy equivalent to $|W| \times \{0\}$, the map is surjective and the next map in the long exact sequence is the zero-map. We obtain

$$\begin{aligned} \ldots \xrightarrow{0} H_i(|W| \times [0,1], |W| \times \{0,1\}) &\xhookrightarrow{g_i} H_{i-1}(|W| \times \{0,1\}) \xrightarrow{h_{i-1}} \ldots \\ [\sigma] &\longmapsto [\partial\sigma], \end{aligned}$$

where $g_i, i \in \mathbb{Z}$ denote the *connecting homomorphisms* of the long exact sequence in homology. By the exactness of the sequence, g_i is injective and we have

$$H_i(|W| \times [0,1], |W| \times \{0,1\}) \xrightarrow[\sim]{g_i} \mathrm{im}(g_i) = \ker(h_{i-1}).$$

The kernel of h_{i-1} is of the form $\ker(h_{i-1}) = \{([a],[-a]) \in H_{i-1}(|W| \times \{0,1\})\}$. We define isomorphisms

$$\ker(h_{i-1}) \xrightarrow[\sim]{(\mathrm{pr}_0)_*} H_{i-1}(|W| \times \{0\}) \xrightarrow{(\pi_X)_*} H_{i-1}(|W|) \xrightarrow{\sim} H_{i-1}(W)$$

where pr_0 is the projection of $C_\bullet(|W| \times \{0\}) \oplus C_\bullet(|W| \times \{1\})$ to $C_\bullet(|W| \times \{0\})$. By combining all these maps we obtain the required map

$$\begin{aligned} f_i : H_i(X_1^{\mathcal{U}}, X_0^{\mathcal{U}}) \xleftarrow[\sim]{(\subseteq)_*} H_i(W \times [0,1], W \times \{0,1\}) &\xrightarrow{\sim} H_{i-1}(W) \\ [\sigma] &\longmapsto [\pi_X(\mathrm{pr}_0(\partial\sigma))]. \end{aligned} \tag{6.2}$$

We want to check whether the diagram with this map commutes. The square (A) is commutative, since $[\pi_X(\mathrm{pr}_0(\sigma)) + \pi_X(\mathrm{pr}_1(\sigma))] = [\pi_X(\mathrm{pr}_0(\sigma) + \mathrm{pr}_1(\sigma))] = [\pi_X(\sigma)]$ for $\sigma \in H_i(X_0^{\mathcal{U}})$.

To prove the commutativity of (B), let $[\sigma] = [\alpha + \beta + \tau] \in H_i(X_1^{\mathcal{U}})$ with $\alpha \in C_i(U \times \{0\}), \beta \in C_i(V \times \{0\})$ and $\tau \in C_i(W \times [0,1])$ be an arbitrary homology class. In $H_i(X_1^{\mathcal{U}}, X_0^{\mathcal{U}})$ we have $[\sigma] = [\tau]$ and by the right part of (B), we obtain

$$\begin{aligned} H_i(X_1^{\mathcal{U}}) \longrightarrow H_i(X_1^{\mathcal{U}}, X_0^{\mathcal{U}}) &\longrightarrow H_{i-1}(W) \\ [\sigma] &\longmapsto [\pi_X(\mathrm{pr}_0(\partial\tau))]. \end{aligned}$$

Furthermore, we know that $\pi_X(\alpha) \in C_i(U)$ and $\pi_X(\beta + \tau) \in C_i(V)$. Therefore, the maps on the left part of the square yield

$$\begin{aligned} H_i(X_1^{\mathcal{U}}) \longrightarrow H_i(X) &\longrightarrow H_{i-1}(W) \\ [\sigma] &\longmapsto [\partial\pi_X(\alpha)]. \end{aligned}$$

We have to show that $[\partial\pi_X(\alpha)]$ and $[\pi_X(\mathrm{pr}_0(\partial\tau))]$ coincide. Since $\mathrm{pr}_0(\partial\beta) = 0$, it holds $\mathrm{pr}_0(\partial\alpha) + \mathrm{pr}_0(\partial\tau) = \mathrm{pr}_0(\partial(\alpha + \beta + \tau))$. This is zero since $\partial(\alpha + \beta + \tau) = 0$. We obtain $[\pi_X(\mathrm{pr}_0(\partial\tau))] = [-\pi_X(\mathrm{pr}_0(\partial\alpha))] = [-\pi_X(\partial\alpha)]$. The square commutes up to a sign since π_X is a chain map. By multiplying those vertical maps which map to a homology with even degree we make the diagram commutative[1].

It remains to prove, that the square (C) commutes. Consider $[\sigma] \in H_i(X_1^{\mathcal{U}}, X_0^{\mathcal{U}})$. By using the maps from the right and left part of the square, we obtain

$$\begin{aligned} [\sigma] \longmapsto\ & [\partial\sigma] \longmapsto [(\pi_X(\mathrm{pr}_0(\partial\sigma)), \pi_X(\mathrm{pr}_1(\partial\sigma)))] \\ [\sigma] \longmapsto\ & [\pi_X(\mathrm{pr}_0(\partial\sigma))] \longmapsto [(\pi_X(\mathrm{pr}_0(\partial\sigma)), -\pi_X(\mathrm{pr}_0(\partial\sigma)))]. \end{aligned}$$

To decide whether the maps coincide, we need to check if

$$[\pi_X(\mathrm{pr}_1(\partial\sigma))] = [-\pi_X(\mathrm{pr}_0(\partial\sigma))].$$

[1] Alternatively, one could multiply the bottom map of the square (B) with (-1). This also preserves the property of being a long exact sequence and we obtain a commutative diagram.

We know that $[\sigma]$ is in the kernel of ${C_i(X_1^{\mathcal{U}})}/{C_i(X_0^{\mathcal{U}})} \to {C_{i-1}(X_1^{\mathcal{U}})}/{C_{i-1}(X_0^{\mathcal{U}})}$. It holds $(\mathrm{pr}_0 + \mathrm{pr}_1)(\partial\sigma) = \partial\sigma$ since $\partial\sigma \in C_{i-1}(X_0^{\mathcal{U}})$. We conclude

$$\pi_X(\mathrm{pr}_1(\partial\sigma)) + \pi_X(\mathrm{pr}_0(\partial\sigma)) = \pi_X(\partial\sigma) = \partial\pi_X(\sigma).$$

Hence, the difference of $\pi_X(\mathrm{pr}_1(\partial\sigma))$ and $-\pi_X(\mathrm{pr}_0(\partial\sigma))$ is in the image of ∂. □

6.4 Source Code

The entire source code including some examples is uploaded to github at [Gün19] to make it easily available to the reader. Nevertheless, the most important files can also be found here.

simpcells.py

```
1
2
3  class cell:
4      """Simplicial cells.
5
6      Args:
7          name: Name of the cell.
8          boundary(list): The boundary cells.
9
10     Attributes:
11         name: Name of the cell.
12         boundary(list): List of boundary cells.
13         partner: Partner cell.
14         basisel: Corresponding basis element.
15         order: Order of the cell.
16         dimension: Dimension of the cell.
17         index: Index of the cell in the filtration.
18     """
19
20     def __init__(self, name, boundary):
21         self.name = name
22         self.boundary = boundary
23         self.partner = None
24         self.basisel = [self, ]
25         self.order = None
26         self.dimension = None
27         self.index = None
28
29     def __repr__(self):
30         """Return the string of a cell."""
31         # return 'name: '+str(self.name) + ', boundary: '+str(self.boundary)
32         return str(self.name)
33
34
35 def boundary(list):
36     """Computes the boundary of a chain.
37
38     Args:
39         list(list): A list of cells representing a chain in F2.
40
41     Returns:
42         list: A list of cells representing the boundary in F2.
43     """
44     boundary_list = []
45     for k in list:
46         for j in k.boundary:
47             if j in boundary_list:
48                 boundary_list.remove(j)
49             else:
50                 boundary_list.append(j)
51     return boundary_list
52
53
54 def add_chains(A, B):
55     """Addition of two chains in F2.
56
57     Args:
58         A(list): List of cells.
59         B(list): list of cells.
60
61     Returns:
62         list: The addition of A and B.
63     """
64     C = A.copy()
65     for b in B:
66         if b in C:
```

```
            C.remove(b)
        else:
            C.append(b)
    return C
```

homology.py

```
import simpcells as sc


def change_basis(K):
    """Algorithm 1.

    Args:
        K(list): List of cells representing the filtration.
    """
    i = 0  # number of iteration
    for k in K:
        # STEP 1
        while True:
            boundary_of_basisel = sc.boundary(k.basisel)  # compute boundary
            if len(boundary_of_basisel) == 0:
                break
            else:
                tau = youngest(boundary_of_basisel)
                if tau.partner is None:
                    assign_partner(tau, k)
                    break
                else:
                    k.basisel = sc.add_chains(k.basisel, tau.partner.basisel)
        # STEP 2
        partner = k.partner  # For better readability
        if partner is not None:
            eliminate = sc.add_chains(partner.basisel, sc.boundary(k.basisel))
            while len(eliminate) != 0:
                tau = youngest(eliminate)
                partner.basisel = sc.add_chains(partner.basisel, tau.basisel)
                eliminate = sc.add_chains(eliminate, tau.basisel)
        # print number to track the progress for long lists:
        if i % 100 == 0:
            print(i)
        i += 1


def youngest(list_of_cells):
    """Find the youngest cell in a list of cells.

    Notes:
        Every cells needs to have an index.

    Args:
        list_of_cells(list): List of cells to search for youngest.

    Returns:
        cell: Youngest cell in list_of_cells.
    """
    min = None
    min_index = None
    for k in list_of_cells:
        if min is None:
            min = k
            min_index = k.index
        else:
            if min_index < k.index:
                min = k
                min_index = k.index
    return min


def assign_partner(cell1, cell2):
    """Algorithm 2.

    Args:
        cell1(cell): Cell to assign the other cell as partner.
        cell2(cell): Cell to assign the other cell as partner.
    """
    cell1.partner = cell2
    cell2.partner = cell1


def change_basis_without2(K):
    """Algorithm 3.

    Args:
        K(list): List of cells representing the filtration.
    """
    i = 0  # number of iteration
    for k in K:
        while True:
            boundary_of_basisel = sc.boundary(k.basisel)  # compute boundary
            if len(boundary_of_basisel) == 0:
                break
            else:
```

```
                tau = youngest(boundary_of_basisel)
                if tau.partner is None:
                    assign_partner(tau, k)
                    break
                else:
                    k.basisel = sc.add_chains(k.basisel, tau.partner.basisel)
        # print number to track the progress for long lists:
        if i % 100 == 0:
            print(i)
        i += 1


def adjust_basisel(K):
    """Algorithm 4.

    Args:
        K(list): List of cells representing the filtration.
    """
    i = 0  # number of iteration
    for k in K:
        partner = k.partner  # For better readability
        if partner is not None and partner.index > k.index:
            k.basisel = sc.boundary(partner.basisel)
        # print number to track the progress for long lists:
        if i % 100 == 0:
            print(i)
        i += 1


def compute_homology(K, step2=False):
    """Compute homology with Algorithm 3 (and Algorithm 4).

    Args:
        K(list): List of cells representing the filtration.
        step2(boolean, optional): Decide whether to execute step 2 or not.
            Defaults to False.
    """
    add_indices(K)  # Add indices to the cells in the list
    print(' -- change basis without step 2 -- ')
    change_basis_without2(K)
    if step2:
        print(' -- adjust basis elements -- ')
        adjust_basisel(K)


def add_indices(K):
    """Adds indices to the cells with respect to the list.

    Args:
        K(list): List of all cells.
    """
    i = 0
    for k in K:
        k.index = i
        i += 1


def get_barcodes(K, max_value=None):
    """Get the barcodes of cells by partner assignment.

    Notes:
        The function compute_homology() should habe beeen used before.
        The order of each simplicial cell has to be defined.

    Args:
        K(list): List of cells.
        max_value(optional): Value that should be used as right entry of the
            interval. Defaults to None.

    Returns:
        list: List of intervals.
        list: List of corresponding generators.
    """
    max_dimension = 0
    for k in K:
        if k.dimension > max_dimension:
            max_dimension = k.dimension

    list = []
    list2 = []
    for i in range(max_dimension+1):
        list.append([i, []])
        list2.append([i, []])

    for k in K:
        if k.partner is None:
            list[k.dimension][1].append([k.order, max_value])
            list2[k.dimension][1].append(k)
        elif k.partner.order > k.order:
            list[k.dimension][1].append([k.order, k.partner.order])
            list2[k.dimension][1].append(k)

    return list, list2


```

```
def draw_barcode(list, dimension, K, max_value=None):
    """Draw a barcode.

    Notes:
        The function uses the module matplotlib.

    Args:
        list(list): List of intervals obtained from get_barcodes() that should
            be drawn.
        dimension(int): Dimension of the homology that should be drawn.
        K: List where the barcode comes from. We need this to compute the
            maximal value of the intervals to decide where to end the diagram.
        max_value(optional): Value where to cut the diagram. If this argument
            is given, K can be set to an arbitrary value like None.
    """

    if max_value is None:
        max_value = 0
        for k in K:
            if k.order > max_value:
                max_value = k.order

    if dimension > len(list)-1:
        print('no homology in this dimension')
        return

    import matplotlib.pyplot as plt

    item = list[dimension]
    print(item)
    index = 0
    fig, ax = plt.subplots()
    for interval in item[1]:
        if interval[1] is None:
            ax.arrow(interval[0], index, max_value-interval[0], 0,
                     color='black', width=0.02, head_width=0, head_length=0)
            ax.arrow(max_value, index, 0.1*max_value, 0,
                     color='r', width=0.02, head_width=0, head_length=0)
        else:
            ax.arrow(interval[0], index, interval[1]-interval[0], 0,
                     color='black', width=0.02, head_width=0, head_length=0)
        index = index + 1
    ax.set_yticks([])
    ax.set_ylim(-1, index)
    ax.set_xlim(0, max_value+1)
    plt.show()
```

tuple.py

```


class tuple:
    """A general tuple.

    Notes:
        If their entries coincide, they should be equal. If we just compare
        lists itself in python, they do not coincide in general if their
        entries coincide.

    Args:
        list_of_points(list): A list of points representing a tuple.

    Attributes:
        tuple(list): The tuple
    """

    def __init__(self, list):
        self.tuple = list

    def without(self, number):
        """Returns a new tuple without item with index 'number'."""
        t = self.tuple[:number]+self.tuple[number+1:]
        return tuple(t)

    def __eq__(self, other):
        """Decide whether two points coincide by =."""
        if len(self.tuple) != len(other.tuple):
            return False
        token = True
        for i in range(len(self.tuple)):
            if self.tuple[i] != other.tuple[i]:
                token = False
        return token

    def __repr__(self):
        """Return the string of a tuple."""
        return str(self.tuple)

    def __getitem__(self, ii):
        """Get a list item."""
        return self.tuple[ii]


```

```
def all_indices(k, m):
    """Yield all ordered lists of length k with pairwise different entries
        in {0,...,m-1} .

    Notes:
        Yield is used in this function. It behaves like return but it returns a
        generator over which we can iterate.

    Args:
        k(int): Number of entries in the sublists we want to generate.
        m(int): Length of the list that we want to choose from.
    """

    # yield the first sublist:
    liste = []
    for i in range(k):
        liste.append(i)
    yield liste

    # function to modify the sublist:
    def moveentry(j, list):
        # if no element in the sublist can be changed return the empty list
        if j == -1:
            return []
        # increase the j-th element and minimize all following entries
        if list[j]+1 < m and list[j]+1 not in list:
            new_list = list[:j]
            for i in range(len(list)-len(new_list)):
                new_list.append(list[j]+1+i)
            return new_list
        # if the j-th entry can not be increased, try the (j-1)-th
        else:
            return moveentry(j-1, list)

    # yield a sublist and change change it by the function from above:
    while True:
        liste = moveentry(len(liste)-1, liste)
        if liste == []:  # quit if the list can not be changed anymore
            break
        else:
            yield liste
```

points_to_complex.py

```
import simpcells as sc
import tuple as tup
import numpy as np  # for square and square root


class point:
    """Points in the real dimensional space.

    Notes:
        We want to save points as lists and do it as object for the same reason
        as for tuple.

    Args:
        list(list): List of values representing a point in a finite dimensional
            real vecotr space.

    Attributes:
        coordinates(list): Coordinates of the point in form of a list.
        dimension(int): Dimension des Punktes.
    """

    def __init__(self, list):
        self.coordinates = list
        self.dimension = len(list)

    def distance(self, other_point):
        """Distance to another point."""
        x = 0
        for i in range(self.dimension):
            x = x + np.square(self.coordinates[i] - other_point.coordinates[i])
        return np.sqrt(x)

    def add(self, other_point):
        """Return the sum of the point and the other point."""
        list = []
        for i in range(self.dimension):
            list.append(self.coordinates[i] + other_point.coordinates[i])
        p = point(list)
        return p

    def scalarmult(self, scalar):
        """Return the multiplication with a scalar."""
        list = []
        for i in range(self.dimension):
            list.append(scalar * self.coordinates[i])
        p = point(list)
        return p

    def absolute_value(self):
```

```
        """Compute the absolute value of the point."""
        x = 0
        for i in range(self.dimension):
            x = x + np.square(self.coordinates[i])
        return np.sqrt(x)

    def __eq__(self, other_point):
        """Decide whether two points coincide by =."""
        if self.dimension != other_point.dimension:
            return False
        token = True
        for i in range(self.dimension):
            if self.coordinates[i] != other_point.coordinates[i]:
                token = False
        return token

    def __repr__(self):
        """Return the string of a point."""
        return str(self.coordinates)

    def __getitem__(self, ii):
        """Get a coordinate."""
        return self.coordinates[ii]


def points_to_cells_VR(list_of_points, max_radius):
    """Compute the Vietoris Rips complex for a list of points.

    Args:
        list_of_points(list): List of all points.
        max_radius: prescribed radius for the construction.

    Returns:
        list: List of cells already ordered in the right way.
    """

    list_of_cells = []  # list of all cells
    list_of_cells_by_len = [[]]  # store by dimension of cells
    index_list = [[]]  # queue of lists of indices
    number_of_points = len(list_of_points)

    printindex = 0

    while len(index_list) != 0:  # while there are lists in the queue
        last_index = index_list.pop(0)  # take the last element
        max_last_index = max(last_index) if len(last_index) != 0 else -1
        for i in range(max_last_index+1, number_of_points):
            indices = last_index + [i]  # create new list of indices
            # create tuple of points for this list of indices:
            list_some_points = []
            for i in indices:
                list_some_points.append(list_of_points[i])
            tuple_some_points = tup.tuple(list_some_points)
            # compute boundary:
            boundary_some_points = []
            for i in indices:  # loop over all elements in the boundary
                new_boundary = []
                for j in indices:
                    if i != j:
                        new_boundary.append(list_of_points[j])
                new_boundary_tuple = tup.tuple(new_boundary)
                # search for cells in right dimension:
                for c in list_of_cells_by_len[len(new_boundary_tuple.tuple)-1]:
                    if c.name == new_boundary_tuple:
                        boundary_some_points.append(c)  # append boundary
            # create a cell using the tuple as name and the computed boundary:
            new_cell = sc.cell(tuple_some_points, boundary_some_points)
            # add dimension:
            new_cell.dimension = len(indices)-1
            # add order to the cell:
            new_cell_distance = 0
            for k1 in range(len(indices)):
                for k2 in range(k1+1, len(indices)):
                    dist = tuple_some_points.tuple[k1].distance(
                        tuple_some_points.tuple[k2])
                    if dist > new_cell_distance:
                        new_cell_distance = dist
            new_cell.order = new_cell_distance/2
            # add cell to the list if it does not exceed the maximal radius
            if new_cell_distance/2 <= max_radius:
                list_of_cells.append(new_cell)  # add to cell list
                index_list.append(indices)  # add list of indices to queue
                if len(new_cell.name.tuple)-1 >= len(list_of_cells_by_len):
                    list_of_cells_by_len.append([])
                list_of_cells_by_len[len(new_cell.name.tuple)-1].append(new_cell)
        if printindex % 100 == 0:
            print(printindex)
        printindex += 1
    # At this point the cells are ordered by dimension. We order them at first
    # by their order and then by their dimension.
    list_of_cells = sort_by_order(list_of_cells)

    return list_of_cells


```

```
145 def sort_by_order(list_of_cells):
146     """Sort a list of cells at first by order and then by their dimension.
147 
148     Notes:
149         The algorithm just sorts the elements that are not already ordered.
150         Therefore, if the list is ordered before by dimension, then after this
151         procedure it is ordered at first by the order of the cells and then by
152         their dimension.
153 
154     Args:
155         list_of_cells(list): A list of cells which we want to order.
156 
157     Returns:
158         list: A new list containing all cells of list_of_cells but ordered by
159         their order.
160     """
161     new_list_of_cells = []
162     for c in list_of_cells:
163         i = len(new_list_of_cells)  # we begin searchin on the right side
164         while True:
165             # if we arrive at 0, we add the cell there:
166             if i == 0:
167                 new_list_of_cells.insert(i, c)
168                 break
169             # if the order of the next one is still higher, we go one step
170             elif new_list_of_cells[i-1].order > c.order:
171                 i = i-1
172             # insert cell if the next one has at most the same order
173             else:
174                 new_list_of_cells.insert(i, c)
175                 break
176 
177     return new_list_of_cells
178 
179 
180 def points_to_cells_Cech(list_of_points, max_radius, precision=0.01):
181     """Compute the Cech complex for a list of points.
182 
183     Notes:
184         This algorithm can just be used for points in two dimensional space
185         since the function verification_for_cech() is just implemented for
186         two dimensions.
187 
188     Args:
189         list_of_points(list): List of all points.
190         max_radius: prescribed radius for the construction.
191         precision(float, optional): The value by which we enlarge the radius in
192             each step.
193 
194     Returns:
195         list: List of cells already ordered in the right way.
196     """
197 
198     '''
199     ACHTUNG: wir muessen draw_barcode mit einer tolerance benutzen, damit nicht
200     jeder kleine strich angezeigt wird, der von den vielleicht nicht ganz
201     genauen punkten stammt.
202     '''
203     # compute cells by Vietoris-Rips:
204     cell_list = points_to_cells_VR(list_of_points, max_radius)
205 
206     # enlarge the radius for each cell until it satisfies the property for Cech
207     radius = 0
208     for c in cell_list:
209         radius = c.order  # start with radius from VR
210         while True:
211             if verification_for_cech(c.name.tuple, radius):  # check
212                 c.order = radius  # set order
213                 break
214             else:
215                 radius = radius + precision
216 
217     # order cells:
218     cell_list = sort_by_order(cell_list)
219 
220     return cell_list
221 
222 
223 def verification_for_cech(candidate_points, radius):
224     """Check whether all balls with presribed radius at the points intersect.
225 
226     Args:
227         candidate_points(list): List of the points.
228         radius(float): Radius of the balls.
229 
230     Returns:
231         boolean: True if all balls at the points intersect.
232     """
233     # create list of all pairwise intersection points:
234     intersection_points = []
235     h = len(candidate_points)
236     for i in range(h):
237         for j in range(i+1, h):
238             c1 = candidate_points[i]
239             c2 = candidate_points[j]
```

```
            for k in intersection(c1, c2, radius):
                intersection_points.append([i, j, k])

    # if there are less then two balls then they all intersect:
    if h < 2:
        return True

    # check whether one intersection point is in all balls:
    veri = False
    for inter in intersection_points:
        # Note that inter[0] and inter[1] are indices of the points of the
        # center of the balls and inter[2] is one of their intersection points.
        veri = True
        for k in range(h):
            if k != inter[0] and k != inter[1]:
                point_in_all_balls = True
                for p in candidate_points:
                    if p.distance(inter[2]) > radius:
                        point_in_all_balls = False
                if not point_in_all_balls:
                    veri = False
                    break
        if veri:
            break
    return veri


def intersection(p1, p2, radius):
    """Compute the intersection of two balls.

    Args:
        p1,p2(point): Centers of the balls.
        radius(float): Radius of the balls.

    Returns:
        boolean: True if all balls at the points intersect.
    """
    # no intersection if their radius is smaller than 1/2 of their distance:
    distance = p1.distance(p2)
    if radius < distance/2.0:
        return []

    # compute intersection points:
    vector_p1_to_p2 = p2.add(p1.scalarmult(-1))
    center = p1.add(vector_p1_to_p2.scalarmult(0.5))
    ortho = orthonormal(vector_p1_to_p2)
    h = np.sqrt(np.square(radius) - np.square(distance/2.0))
    if h == 0:
        intersectionlist = [center]
    else:
        intersectionlist = []
        intersectionlist.append(center.add(ortho.scalarmult(h)))
        intersectionlist.append(center.add(ortho.scalarmult(-h)))

    return intersectionlist


def orthonormal(p):
    """Compute orthonormal vector.

    Notes:
        This function works only for two dimensions.

    Args:
        p(point): A vector of dimension 2.

    Returns:
        point: A vector which is orthonormal to p.
    """
    orthogonal = point([p.coordinates[1], -1 * p.coordinates[0]])
    orthonormal = orthogonal.scalarmult(1/orthogonal.absolute_value())

    return orthonormal
```

blowup.py

```
import simpcells as sc
import tuple as tup


def construct_mv_blowup(cell_list, cover):
    """Construct the Mayer-Vietoris blowup for a given cover.

    Args:
        cell_list(list): A list of cells describing the simplicial complex.
        cover(list): A list of cell lists, describing the cover by
            subcomplexes.

    Returns:
        list: A new list of cells describing the complex of the Mayer-Vietoris
            blowup.
    """
    n = len(cover)  # number of subcomplexes in the cover
```

```
    new_cell_list = []

    # compute generators for the blowup and safe them as pairs:
    for i in range(n):  # consider i-th intersections
        for l in tup.all_indices(i+1, n):  # J from the end of Chapter 5
            l_cover = []
            for j in l:
                l_cover.append(cover[j])
            inter = intersection(l_cover)  # compute intersection
            for c in inter:  # sigma from the end of Chapter 5
                # create cell for the pair (sigma, J):
                new_cell = sc.cell(tup.tuple([c, tup.tuple(l)]), [])
                new_cell_list.append(new_cell)  # append cell to new_cell_list

    # sort by #J and then by dim(sigma):
    new_cell_list = sort_for_loc(new_cell_list)

    # compute and set boundary
    for c in new_cell_list:
        boundary_list = []
        sigma = c.name.tuple[0]
        # the part where we take the boundary of sigma:
        J = c.name.tuple[1]
        for b in sigma.boundary:
            # search for boundary in new_cell_list:
            for i in new_cell_list:
                if i.name.tuple[0] == b and i.name.tuple[1] == J:
                    boundary_list.append(i)
                    break
        # the part where we take the boundary of J
        if len(J.tuple) > 1:  # otherwise the boundary forms no cell
            for j in range(len(J.tuple)):
                new_J = J.without(j)
                for i in new_cell_list:
                    if i.name.tuple[0] == sigma and i.name.tuple[1] == new_J:
                        boundary_list.append(i)
                        break
        # store the boundary list in the cell
        c.boundary = boundary_list

    # set dimension
    for c in new_cell_list:
        d = c.name.tuple[0].dimension + len(c.name.tuple[1].tuple)-1
        c.dimension = d

    # set order
    for c in new_cell_list:
        o = len(c.name.tuple[1].tuple)-1
        c.order = o

    return new_cell_list


def intersection(list_of_lists):
    """Compute the intersection of several lists.

    Args:
        list_of_lists(list): A list, which contains all lists, that we want to
            intersect.

    Returns:
        list: The intersection of all lists in list_of_lists.
    """
    new_list = []
    # take elements in the first list:
    for i in list_of_lists[0]:
        token = True
        # check if they are in all other lists:
        for l in list_of_lists[1:]:
            if i not in l:
                token = False
                break
        # if they are in all other lists, add them to the intersection:
        if token:
            new_list.append(i)
    return new_list


def sort_for_loc(list_of_cells):
    """Sort the list for the Mayer-Viertoris blowup."""
    new_list_of_cells = sort_by_dimension_of_sigma(list_of_cells)
    new_list_of_cells = sort_by_len_of_J(new_list_of_cells)
    return new_list_of_cells


def sort_by_dimension_of_sigma(list_of_cells):
    """Sort the list for the Mayer-Viertoris blowup by dimension of sigma."""
    # insertion sort
    new_list_of_cells = []
    for c in list_of_cells:
        i = len(new_list_of_cells)
        token = True
        while token:
            if i == 0:
                new_list_of_cells.insert(i, c)
```

```
113                 token = False
114             elif (new_list_of_cells[i-1].name.tuple[0].dimension
115                   > c.name.tuple[0].dimension):
116                 i = i-1
117             else:
118                 new_list_of_cells.insert(i, c)
119                 token = False
120
121     return new_list_of_cells
122
123
124 def sort_by_len_of_J(list_of_cells):
125     """Sort the list for the Mayer-Viertoris blowup by length of J."""
126     # insertion sort
127     new_list_of_cells = []
128     for c in list_of_cells:
129         i = len(new_list_of_cells)
130         token = True
131         while token:
132             if i == 0:
133                 new_list_of_cells.insert(i, c)
134                 token = False
135             elif (len(new_list_of_cells[i-1].name.tuple[1].tuple)
136                   > len(c.name.tuple[1].tuple)):
137                 i = i-1
138             else:
139                 new_list_of_cells.insert(i, c)
140                 token = False
141
142     return new_list_of_cells
```

Bibliography

[AA17] MichałAdamaszek and Henry Adams. The Vietoris-Rips complexes of a circle. *Pacific J. Math.*, 290(1):1–40, 2017.

[Bre97] Glen E. Bredon. *Topology and geometry*, volume 139 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1997. Corrected third printing of the 1993 original.

[Car09] Gunnar Carlsson. Topology and data. *Bull. Amer. Math. Soc. (N.S.)*, 46(2):255–308, 2009.

[Cro05] Martin D. Crossley. *Essential topology*. Springer Undergraduate Mathematics Series. Springer-Verlag London, Ltd., London, 2005.

[CZCG05] Gunnar Carlsson, Afra Zomorodian, Anne Collins, and Leonidas J. Guibas. Persistence barcodes for shapes. *International Journal of Shape Modeling*, 11(02):149–187, 2005.

[EH10] Herbert Edelsbrunner and John L. Harer. *Computational topology – An introduction*. American Mathematical Society, Providence, RI, 2010.

[ELZ00] Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. Topological persistence and simplification. In *41st Annual Symposium on Foundations of Computer Science (Redondo Beach, CA, 2000)*, pages 454–463. IEEE Comput. Soc. Press, Los Alamitos, CA, 2000.

[Ghr08] Robert Ghrist. Barcodes: the persistent topology of data. *Bull. Amer. Math. Soc. (N.S.)*, 45(1):61–75, 2008.

[GND+19] Rickard Brüel Gabrielsson, Bradley J. Nelson, Anjan Dwaraknath, Primoz Skraba, Leonidas J. Guibas, and Gunnar E. Carlsson. A topology layer for machine learning. *CoRR*, abs/1905.12200, 2019.

[Gün19] Mario Günzel. Persistent-homology-localizing. `https://github.com/mariognzl/Persistent-Homology-Localizing/`, 21-Oct-2019.

[GZ67] P. Gabriel and M. Zisman. *Calculus of fractions and homotopy theory*. Ergebnisse der Mathematik und ihrer Grenzgebiete, Band 35. Springer-Verlag New York, Inc., New York, 1967.

[Hat02] Allen Hatcher. *Algebraic topology*. Cambridge University Press, Cambridge, 2002.

[HS97] P. J. Hilton and U. Stammbach. *A course in homological algebra*, volume 4 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, second edition, 1997.

[LMDV15] Ngoc-Khuyen Le, Philippe Martins, Laurent Decreusefond, and Anais Vergne. Construction of the generalized czech complex. In *2015 IEEE 81st Vehicular Technology Conference (VTC Spring)*, pages 1–5. IEEE, 2015.

[QTT+18] Talha Qaiser, Yee-Wah Tsang, Daiki Taniyama, Naoya Sakamoto, Kazuaki Nakane, David B. A. Epstein, and Nasir M. Rajpoot. Fast and accurate tumor segmentation of histology images using persistent homology and deep convolutional features. *CoRR*, abs/1805.03699, 2018.

[Rot09] Joseph J. Rotman. *An introduction to homological algebra*. Universitext. Springer, New York, second edition, 2009.

[SZ88] Ralph Stöcker and Heiner Zieschang. *Algebraische Topologie – Eine Einführung*. Mathematische Leitfäden. B. G. Teubner, Stuttgart, 1988.

[Wan12] Kairui Glen Wang. The basic theory of persistent homology, 2012. REU 2012.

[Wei94] Charles A. Weibel. *An introduction to homological algebra*, volume 38 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge, 1994.

[XW14] Kelin Xia and Guo-Wei Wei. Persistent homology analysis of protein structure, flexibility, and folding. *International journal for numerical methods in biomedical engineering*, 30(8):814–844, 2014.

[ZC05] Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete Comput. Geom.*, 33(2):249–274, 2005.

[ZC08] Afra Zomorodian and Gunnar Carlsson. Localized homology. *Comput. Geom.*, 41(3):126–148, 2008.

Declaration of Authorship

I hereby declare that this thesis is based on my own unaided work, unless stated otherwise. All references and verbatim extracts have been marked as such and I assure that no other sources have been used. Moreover, this thesis has not been part of any other exam in this or any other form.

Essen, _____ _____

Signature