

Constructive Mayer-Vietoris Algorithm: Computing the Homology of Unions of Simplicial Complexes

Dobrina Boltcheva, Sara Merino Aceitunos, Jean-Claude Léon, Franck Hétroy

▶ **To cite this version:**

Dobrina Boltcheva, Sara Merino Aceitunos, Jean-Claude Léon, Franck Hétroy. Constructive Mayer-Vietoris Algorithm: Computing the Homology of Unions of Simplicial Complexes. [Research Report] RR-7471, INRIA. 2010. inria-00542717

HAL Id: inria-00542717

https://inria.hal.science/inria-00542717

Submitted on 3 Dec 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Constructive Mayer-Vietoris Algorithm: Computing the Homology of Unions of Simplicial Complexes

Sara Merino — Dobrina Boltcheva — Jean-Claude Léon — Franck Hétroy

N° 7471

December 2010

Thème NUM

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--7471--FR+ENG



Constructive Mayer-Vietoris Algorithm: Computing the Homology of Unions of Simplicial Complexes

Sara Merino*, Dobrina Boltcheva†, Jean-Claude Léon‡, Franck Hétroy§

Thème NUM — Systèmes numériques
Projet EVASION

Rapport de recherche n° 7471 — December 2010 — 133 pages

Abstract: In this research report, we present an efficient method for computing the homology of a large simplicial complex from the homologies of its sub-complexes.

The method uses a *constructive* version of the *Mayer-Vietoris exact sequence* which is an algebraic tool relating the homology of a topological space to the homologies of its sub-spaces and their intersection. The method starts by decomposing the input simplicial complex into smaller sub-complexes, for which the homology is computable with the Smith Normal Form reduction algorithm. Then, the method uses the Mayer-Vietoris sequence on the decomposition graph and computes the homology of the input complex by recursive unions of the homological attributes of the sub-complexes.

The proposed method outputs all homological attributes (Betti numbers, torsion coefficients and generators) and may be applied to any kind of finite simplicial complexes (manifold/non-manifold, orientable or not, embeddable or not, with heterogeneous dimensionality, etc.)

Key-words: constructive homology, Mayer-Vietoris exact sequence, simplicial complex, algorithm, generators

Special thanks are due to Francis Sergeraert, researcher at the Fourier Institute of Grenoble. *http://www-fourier.ujf-grenoble.fr/ sergerar/*

* Evasion-INRIA, LJK
† Evasion-INRIA, LJK
‡ G-SCOP, Evasion-INRIA
§ Evasion-INRIA, LJK

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier (France)
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Algorithme constructif de Mayer-Vietoris : Calcul de l'homologie de l'union de complexes simpliciaux

Résumé : Dans ce rapport, nous présentons une méthode efficace pour le calcul de l'homologie d'un complexe simplicial, à partir des homologies de ses sous-complexes.

La méthode est basée sur une version *constructive* de la *séquence exacte de Mayer-Vietoris* qui est un outil algébrique permettant de trouver l'homologie de l'union de deux espaces topologiques, à partir de l'homologie de leur somme directe et de leur intersection. La méthode commence par décomposer le complexe simplicial 3D en sous-complexes pour lesquels l'homologie est plus facile à calculer par la *réduction de Smith*. Ensuite, l'algorithme parcourt le graphe de la décomposition et utilise la séquence de Mayer-Vietoris pour calculer l'homologie du complexe initial par unions récursives des attributs homologiques de ses sous-complexes.

La méthode fournit tous les attributs homologiques (nombres de Betti, coefficients de torsion et générateurs) et peut être appliquée à tout type de complexes simpliciaux finis (variété/non-variété, orientable/non-orientable, plongeable ou non, muilt-dimensionalité, etc).

Mots-clés : homologie constructive, séquence exacte de Mayer-Vietoris, complexe simplicial, algorithme, générateurs

Contents

Introduction **7**
0.1 General context 7
0.2 Why homology 8
0.3 Computing homology: related work 9
0.4 Contribution 10
0.5 Report structure 12

I Topological background **13**

1 Simplicial complexes **17**
1.1 Definition 17
1.2 Simplicial maps 18
1.3 Abstract simplicial complex 19
1.3.1 Combinatorial structure 19
1.3.2 Relation with simplicial complexes 20
1.4 Embedding of simplicial complexes 20

2 Simplicial homology **23**
2.1 Chain complexes 23
2.1.1 Chain-complex morphisms 25
2.2 Homology 26
2.2.1 Betti numbers and generators 27

3 Classical methods for homology computation **29**
3.1 Long exact sequence 29
3.1.1 Short exact sequence 29
3.1.2 Definition 30
3.1.3 Technique to compute the homology 31
3.1.4 Limitations: the extension problem 33
3.2 Smith Reduction algorithm 33
3.2.1 Presentation 33

3.2.2 General description 34
3.2.3 Limitations 36

Conclusion **37**

II Constructive Homology **39**

4 The constructiveness problem **43**

5 Reduction **45**
5.1 Intuitive introduction 45
5.2 Formal definition 46
5.3 Special case: Homological Smith Reduction 48
5.3.1 Base classification 49
5.3.2 Chain-complex reduction 50
5.3.3 The Homological Smith Reduction algorithm 52
5.3.4 Interpretation of the elements in the reduction 57
5.4 Composition of reductions 59
5.5 Equivalence of reductions 60

6 Cone construction **61**
6.1 The concept of a cone 61
6.1.1 Topological cone 61
6.1.2 Mapping cone and cone of a morphism 62
6.2 Cone of a morphism 63
6.3 Cone and reductions 66
6.4 Cone equivalences 68

7 Effective short exact sequence **69**
7.1 Cone, effective short exact sequences and Lemma 82 70

Conclusion **73**

III Constructive Mayer-Vietoris Algorithm **75**

8 Theoretical algorithm **77**
8.1 One step algorithm 78
8.1.1 Computations simplification 82
8.2 Iterative algorithm 82
8.2.1 Reminder of the formulas in the algorithm 84
8.3 Proof of the correctness of the algorithm 88

9 Observations on the complexity **89**
9.1 Drawbacks 90

10 Optimizations **93**

11 Conclusion and future works **95**

IV Annexe **97**

A List of symbols **99**

B How to compute the Smith Normal Form **101**
B.1 Reminder about chain complex 101
B.2 How to construct the chain complex and the border operator. 102
B.2.1 How to give an orientation to the faces of a simplicial complex 102
B.2.2 How to define a module: give a basis 103
B.2.3 Boundary matrices of border operators 103
B.3 How to find the homological information 104
B.3.1 Smith normal form 104
B.4 Example 109

C Proposed data structure for the implementation **113**
C.0.1 Proposed functions implementation 115

D Handwritten example of One step algorithm **123**

Bibliography **133**

Index **133**

Introduction

0.1 General context

The global project

The present work is part of a project called IDEAL in which three laboratories cooperate: the team EVASION of INRIA (French National Institute for Research in Computer Science and Control) Rhône-Alpes (Montbonnot), the laboratory G-SCOP (Laboratory of Grenoble for Sciences of Conception, Optimisation and Production) (Grenoble) and DISI (Dipartimento di Informatica e Scienze dell'Informazione) (Genova, Italy).

The motivation

The general goal of the IDEAL project is to find out characteristics to classify geometric models and criteria to determine particular features of the shape of these models. The goal is to study, in particular, non-manifold spaces such as idealized industrial CAD models (see figure 1), since they are still ill-understood even if they are frequently used in practice.

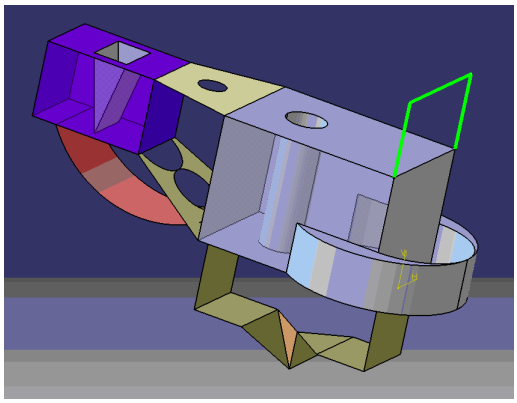


Figure 1: An example of non-manifold simplicial complex. *Author: Jean-Claude Léon*

The goal

The main goal of the work presented in this report was to propose a method for computation the homology of a non-manifold simplicial complex which has been previously decomposed into a set of MC-components (manifold-connected components) with the algorithm developed in DISI, [DFPH09] (see figure 2).

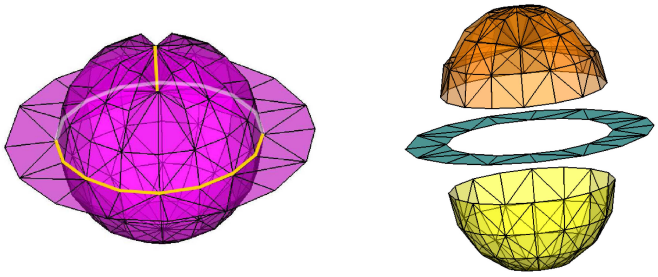


Figure 2: An example of a MC-decomposition. From *A set of tools for representing, decomposing and visualizing non-manifold cellular complexes* Leila De Floriani, Daniele Panozzo, Annie Hui. DISI, University of Genova, Italy

0.2 Why homology

In recent years, the problem of computing the topological features of a space has drawn much attention because it has found real-world applications in many computational disciplines. Unlike geometrical features which are invariant under rigid transformations, such as the curvature of a surface, topological features are invariant under continuous deformations and provide more qualitative and global information about the underlying space, such as the number of its connected components and holes, for example. In particular, for problems involving the shape characterization of objects, topological features offer a more meaningful description than geometric measures and tend to be more robust.

Simplicial homology characterizes a simplicial complex of dimension n by its *homology groups* and their *descriptors*. The notion of *homological descriptor* is defined in any dimension $k \in [0, n]$ and is related to the non-trivial *k-cycles* in the complex. Up to dimension 3, the homological descriptors have comprehensive geometrical meanings. The homological descriptors in dimension 0 are related to the connected components of the complex, in dimension 1 they are related to the non-contractible 1-cycles surrounding tunnels and holes, and, in dimension 2, they describe the shells (2-cycles) surrounding voids or cavities. For higher dimensions, the homological descriptors have analogous interpretations.

Roughly speaking, computing the homology group of dimension k consists in finding all non-trivial k-cycles, i.e.which are not the boundaries of any $(k+1)$-simplexes. Then, the *k^{th}-Betti number* is defined as the number of classes of such non-contractible k-cycles. The representative k-cycles of these classes are called *generators* and localize the cycles within the simplicial complex. The *torsion coefficients* have a very particular interpretation and appear only for simplicial complexes which cannot be embeddedable in $\mathbb{R}^3$, such as some industrial CAD models, for example. The torsion coefficients are related to the *weak-boundaries* which are k-cycles that are not boundaries by themselves but become boundaries when they are considered t times, with $t > 1$ being the associated torsion coefficient.

0.3 Computing homology: related work

Simplicial homology exploits the combinatorial structure of the simplicial complexes and reformulates the homological problem into an algebraic one. It studies the incidence relations between simplices whose dimensions differ by one and encodes these relations into integer *incidence matrices*. The classical approach for computing *all* homological descriptors (including Betti numbers, torsion coefficients and generators) from the incidence matrices is based on the famous *Smith Normal Form (SNF)* algorithm [Mun99, Hat02]. The algorithm uses a Gaussian-like elimination in order to reduce each matrix into a special diagonal form which is very convenient to find out the Betti-numbers and the torsion coefficients. In addition, if the algorithm keeps track of all elementary operations during the reduction , it is also possible to express the *generators* of the homology groups within the input simplicial complex, [Ago76].

Although this algorithm is theoretically defined in any dimension and for any kind of simplicial complexes it faces some strong practical impediments. The problems are mainly linked to the size of the incidence matrices and the high complexity of the reduction algorithm. The best available reduction algorithms have super-cubical complexity [Sto96, DHSW03] and they are suitable only for relatively small simplicial complexes. However, real-ward applications usually deal with very large simplicial complexes and computing their homology via the classical SNF algorithm can easily lead to computations which cannot be handled on modern equipment. An other well-known problem is the possible appearance of huge integers during the reduction while computing over the ring of integers $\mathbb{Z}$, [HM91].

As a consequence, various optimizations have been developed over the years which try to gain in efficiency by minimizing or entirely avoiding the algebraization, i.e. the construction of the incidence matrices used by the SNF algorithm. Therefore, most of these alternative methods restrict the type of the input models and the homological attributes which are actually computed.

There are roughly 3 main approaches.

First, many work has been done on the optimization of the Smith reduction algorithm itself. There are *stochastic optimizations* [Gie96] which are efficient on sparse integer matrices but do not provide the homological generators. There are also *deterministic methods* [KA79] which usually gain in efficiency by performing the computations modulo an integer chosen by a determined criteria [Sto96]. However, the information on the torsion coefficients is lost with this strategy and the algorithms are still not enough efficient to be applied on large simplicial complexes. A particular study has been carried out for the optimisation of the Smith reduction in the particular context of homology, [DHSW03], however, the proposed algorithm does not compute the generators and its practical validity is not yet available.

An alternative to these solutions is the *reduction approach* which iteratively reduces the input complex into a smaller one with the same homology, and computes the homology when no more reductions are possible [KMS98, MPZ08, PIK+09]. This approach has been mainly investigated in the context of cubical complexes and many algorithms have been designed for the homology computation from this special kind of models.

Finally, there is also one more *algorithmic approach* for the homology computation which entirely avoids the algebraization and yields incremental algorithms. These methods are usually designed for simplicial complexes with dimensional restrictions in the most cases. For example, there is a fast algorithm for computing the Betti numbers of simplicial 3-complexes, [DE93]. Many methods are restricted to 2-dimensional spaces. In [EW05, CdVL05] is shown how to compute basis for the non-trivial 1-cycles in a compact 2-manifold which are optimal according to some geometrical measures. Similar approach has been used in [GW01] to remove topological noise of 2-dimensional triangulated surfaces.

In *persistent homology* [EH10] [ELZ00, CSEH07, CSEHM09, ZC05], one considers the global space as a set of nested spaces (filtration) and studies which homological attributes appear, disappear and are maintained through the nesting. In [AGH+09] a persistence-sensitive simplification of functions on surfaces in linear time has been proposed. The method provided in [DLSCS08] distinguishes between the two types of possible 1-cycles on 2-manifold surfaces that localize handles or tunnels. In [ZC08] a method is explained to find out good shape generators over a cover of the space. A Mayer-Vietoris formula for persistent homology with an application to shape recognition in the presence of occlusions is also introduced in [FLM09].

As a conclusion, none of the existing methods is based on a *decomposition approach* which first splits the intput complex into smaller pieces, for which the homology is timely computable with the SNF algorithm, and then computes the homology of the initial complex using the homology of the pieces. An obvious advantage of such an approach would be that the homology of the pieces can be computed in parallel and may even be pre-computed and stored if the application allows this pre-processing.

0.4 Contribution

In this research report, we propose a new algorithmic approach allowing to compute the homology of a large simplicial complex from the homologies of its sub-complexes.

The method is based on a *constructive* version of the *Mayer-Vietoris exact sequence* which is an algebraic tool relating the homology of a topological space to the homologies of its sub-spaces and their intersection, [SR06].

The method starts by decomposing the input simplicial complex into smaller sub-complexes, for which the homology is computable with the Smith Normal Form algorithm. Then, the method uses the Mayer-Vietoris sequence and the decomposition graph and computes the homology of the input complex by recursively merging the homological attributes of the sub-complexes.

The general scheme of our method is illustrated in figure 3 and is the following:

- Decompose the input non-manifold simplicial complexe into MC-components (with the algorithm introduced in [DFPH09]),

- Compute the homological attributes of each MC-component using the Smith Normal Form algorithm,
- Compute the homology of the input complex by merging iteratively the homological attributes of the MC-components using the constructive Mayer-Vietoris sequence,
- Study the evolution of homology during the process of merging of the homological attributes of the subcomplexes.

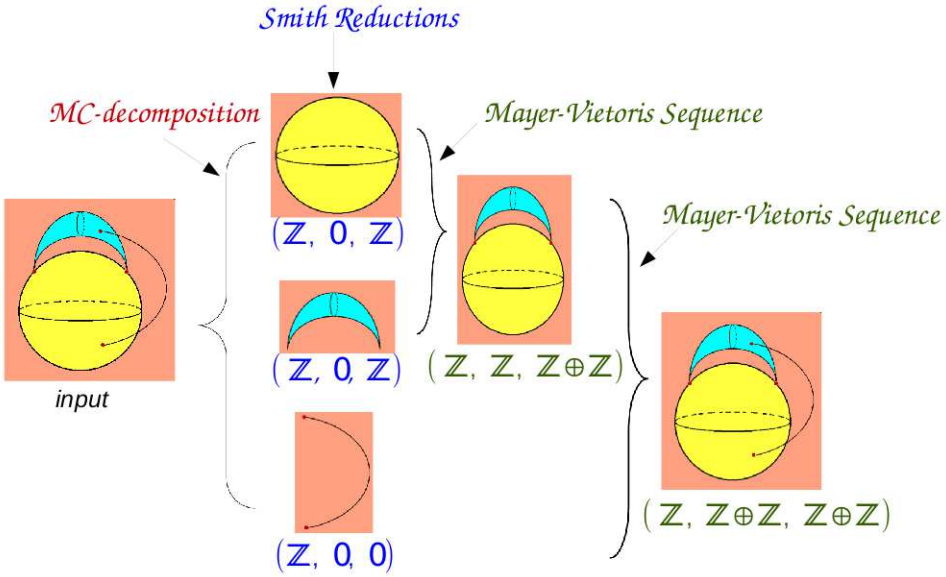


Figure 3: A general overview of the proposed method.

The proposed method outputs all homological attributes (including Betti numbers, torsion coefficients and generators) and may be applied to any kind of finite simplicial complexes (manifold/non-manifold, orientable or not, embeddable or not, with heterogeneous dimensionality, etc.)

In this report, we also give:

- The proof of the correctness of the algorithm (section 8.3).
- An example of data structure and functions implementations (annex C).
- An intuitive introduction to the concept of *reduction* in the sense of [SR06] (section 5.1).
- A detailed description of the algebraic concepts used in our algorithm, namely: the reduction 5 and the *cone of a morphism* 6.2;

- We also show how to build a *reduction* from the Smith Normal Form of the incidence matrices and prove that it is a reduction (section 5.3.4).
- A handwritten example of the One-step Constructive Mayer-Vietoris algorithm (annex D).

The effective implementation of the algorithms has been carried out by Dobrina Boltcheva (post-doctoral researcher at EVASION-INRIA) in collaboration with David Canino (PhD student at DISI).

0.5 Report structure

The report is divided in three parts and some annexes. The first part provides a background on the classical concepts of Algebraic Topology. It is the theoretical foundation of the problem we try to solve (the geometrical model that we consider, the simplicial complexes; the properties that we want to study, Simplicial Homology; the classical methods to compute homology and their limitations).

The second part is a presentation on the concepts of Constructive Homology that will be used in the Constructive Mayer-Vietoris algorithm. Finally, in the third part, the algorithm is explained.

Following the conclusion there are some annexes. These annexes contain related information that, if put inside the report itself, would have broken the rhythm of lecture.

Remark 0.5.1. *We have not included the proofs that are in the indicated references. The only proofs of the document are the ones done by ourselves and that we have not found in the references.*

Part I

Topological background

introduction

In this part, the basic topological concepts are presented and the following questions are answered:

- *Which are the geometrical models considered?*
  Simplicial complexes (chapter 1).
- *How simplicial complexes can be related between them?*
  Simplicial maps (section 1.2).
- *How geometrical models based on simplicial complexes are interpreted?*
  Embeddings (section 1.4).
- *How an algebraic structure is obtained from a simplicial complex?*
  Oriented simplicial complexes, chain complexes and border operator (chapter 2).
- *Which are the properties studied of a simplicial complex ?*
  Simplicial homology (chapter 2).
- *Which are the classical methods to find out the simplicial homology?*
  Exact long sequence of homology, spectral sequences and Smith reduction (chapter 3).
- *Which are the limitations of the previous methods?*
  The extension problem and the computational problem (chapter 3).

The concepts in this chapter are heavily borrowed (and most of the time literally extracted) from [Mun99].

The figure 4 presents a schema of the concepts detailed in this part.

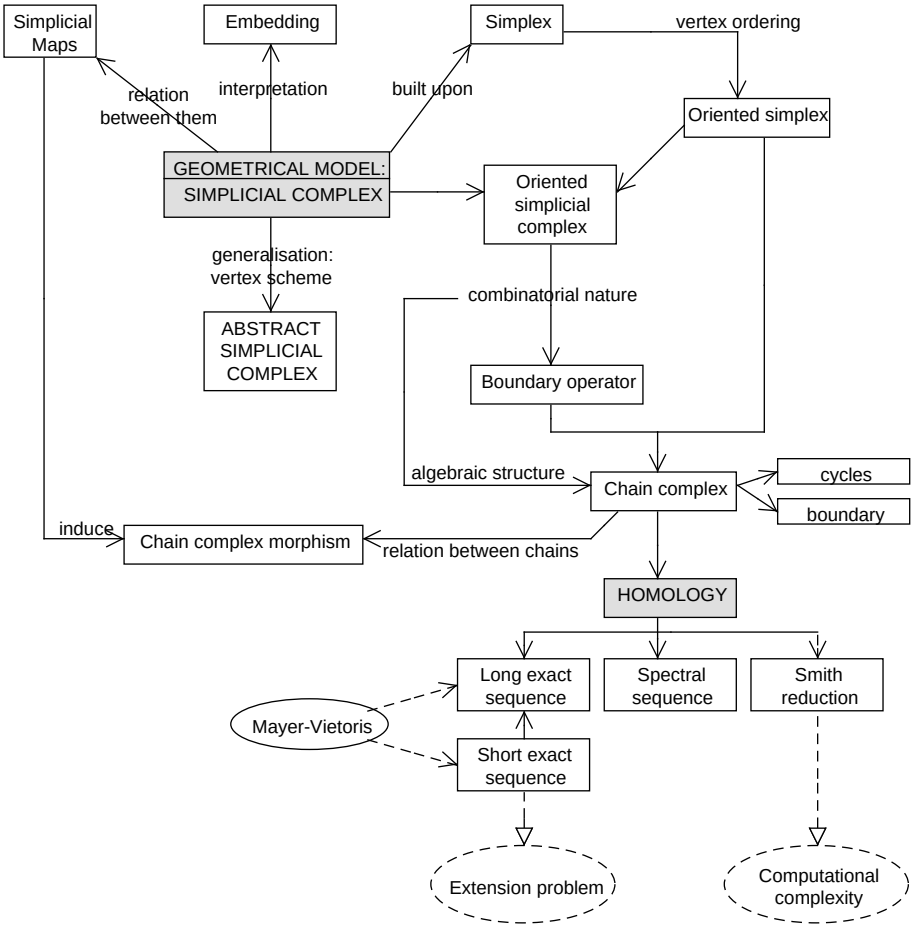


Figure 4: Diagram of concepts in the problem conception

Chapter 1

Simplicial complexes

1.1 Definition

The geometric models considered in the present work are simplicial complexes, those are built upon the concept of a simplex:

Definition 1. Simplex.

Let $\{v_0, v_1, \ldots, v_n\}$ be a geometrically independent set of points in $\mathbb{R}^N$. We define the ***n-simplex*** *σ spanned by $v_0, v_1, \ldots, v_n$ to be the set of all points x of $\mathbb{R}^N$ such that $x = \sum_{i=0}^{n} t_i a_i$, where $\sum_{i=0}^{n} t_i = 1$ and $t_i \geq 0$ for all i.*

unitary element: simplex

Definition 2. Vertices, face, proper face, boundary.

- ***Vertices*** *of σ: points $v_0, v_1, \ldots, v_n$ that span σ.*
- *n is the* ***dimension****.*
- ***Face (or cells) of a simplex****: Any simplex spanned by the subset of $\{v_0, v_1, \ldots, v_n\}$*
  - ***Proper faces****: the faces of σ different from σ itself.*
  - ***Boundary (Bd*** *σ: union of the proper faces of σ).*

Now one can define:

Definition 3. Simplicial complex.

A ***simplicial complex*** *(see figure 1.1 at page 18) X in $\mathbb{R}^N$ is a collection of simplices in $\mathbb{R}^N$ such that:*

1. *Every face of a simplex of X is in X.*
2. *The intersection of any two simplices of X is a face of each one of them.*

The geometrical models: simplicial complexes

an alternative definition is given by the next lemma:

Lemma 1.1.1 (Equivalent definition of simplicial complex)**.** *A collection X of simplices is a simplicial complex if and only if the following hold:*

1. *Every face of a simplex of X is in X.*
2. *Every pair of distinct simplices of X have disjoint interiors.*

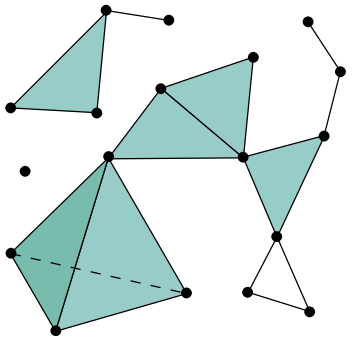


Figure 1.1: Example of a simplicial complex *Source: Wikipedia entry 'Simplicial complex'*

Computational interest

Simplicial complexes are a natural choice to represent geometrical objects computationally because they can be easily implemented in a computer by an enumeration of the faces at each dimension. An example of a file type that can describe simplicial complexes is the OFF file[1]. In addition, simplicial complexes are provided with a combinatorial structure, as explained in the next chapter.

Related structures

A simplicial complex can be associated with the following objects:

- **Subcomplex**: subcollection, Y, of a simplicial complex X that contains all faces of its elements. Y is a simplicial complex in itself.

topology of the simplicial complexes

- **Polytope of X, $|X|$**: is the subset of $\mathbb{R}^N$ that is the union of the simplices of X. Giving each simplex its natural topology as a subspace of $\mathbb{R}^N$, $|X|$ is provided with a topological structure by declaring a subset A of $|X|$ to be closed in $|X|$ if and only if $A \cap \sigma$ is closed in σ, for each σ in X.

The polytope concept is a topological space that has the inner structure of a simplicial complex. It is through this inner structure that an algebraic structure is derived from the topological space.

1.2 Simplicial maps

Lemma 1.2.1 (Simplicial map)**.** *Let X and Y be simplicial complexes, and let $f : X^{(0)} \to Y^{(0)}$ be a map. Suppose that whenever the vertices $v_0, \ldots, v_n$ of X span a simplex of X,*

[1] See http://people.sc.fsu.edu/ burkardt/data/off/off.html for a format description of OFF files

the points $f(v_0), \ldots, f(v_n)$ are vertices of a simplex of Y. Then f can be extended to a continuous map $g : |X| \to |Y|$ such that

$$x = \sum_{i=0}^{n} t_i v_i \Rightarrow g(x) = \sum_{i=0}^{n} t_i f(v_i)$$

We call g the (linear) ***simplicial map*** *induced by the vertex map f.*

The composition of simplicial maps is a simplicial complex.
This concept is important because:

- These maps induce also a relation between the algebraic structures that we are going to associate at each simplicial complex.
- From these relations we can construct new objects (like the cone of a morphism 6) and obtain new information (like in 3.1).

1.3 Abstract simplicial complex

Simplicial complexes can be generalised by studying its combinatorial nature. A new object is then defined, the abstract simplicial complex.

1.3.1 Combinatorial structure

This part is nearly completely extracted from [Mun99]. Firstly, an abstract structure is defined and, afterwards, its relation with simplicial complexes is studied.

Definition 4. Abstract simplicial complex.
An ***abstract simplicial complex*** *is a collection Σ of finite non-empty sets, such that if A is an element of Σ, so is every nonempty subset of A.*

The element A of Σ is called a simplex of Σ; its dimension is one less that the number of its elements. Each nonempty subset of A is called a ***face*** *of A. The dimension of Σ is the largest dimension of one of its simplices, or is infinite if there is no such largest dimension. The* ***vertex set*** *V of Σ is the union of the one-point elements of Σ; we shall make no distinction between the vertex $v \in V$ and the 0-simplex $\{v\} \in \Sigma$. A subcollection of Σ that is itself a complex is called a* ***subcomplex*** *of Σ.*

An abstract simplicial complex can be defined from a simplicial complex by giving its vertex schema:

Definition 5. Vertex scheme.
If X is a simplicial complex, let V be the vertex set of X. Let R be the collection of all subsets $\{a_0, \ldots, a_n\}$ of V such that the vertices $a_0, \ldots, a_n$ span a simplex of X. The collection R is called the ***vertex scheme*** *of X.*

Representation

The collection R is a particular example of an abstract simplicial complex.

1.3.2 Relation with simplicial complexes

- Every abstract simplicial complex Σ is isomorphic to the vertex scheme of some simplicial complex X.
- Two simplicial complexes are linearly isomorphic if and only if their vertex schemes are isomorphic as abstract simplicial complexes.

Definition 6. Geometric realization.
If the abstract simplicial complex Σ is isomorphic with the vertex scheme of the simplicial complex X, we call X a geometric realization of Σ. It is uniquely determined up to a linear isomorphism. (See figure 1.2 at page 20).

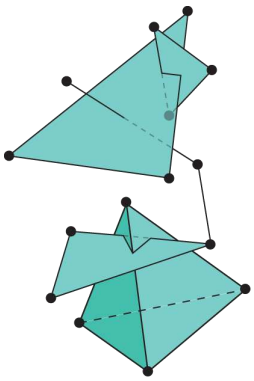


Figure 1.2: A geometrical representation of an abstract simplicial complex that is not a valid simplicial complex *Source: Wikipedia entry abstract simplicial complex*

1.4 Embedding of simplicial complexes

In this project, to find out shape features of geometrical models, they are modeled using simplicial complexes. In this section it is explain how some information is "lost" when the simplicial complexes are considered solely.

When one thinks on a geometrical object, one considers it 'in the space', i.e, not only the object but also the 'space surrounding it'. However, in the study of simplicial complexes (or any topological space), only the topological space is considered; more precisely, its intrinsic properties, which are not related to the ambient space (normally one would think in $\mathbb{R}^3$). To consider an object 'inside' another space is called an **embedding**.

Definition 7. Embedding.
(From Wikipedia entry 'embedding') In general topology, an embedding is a homeomorphism onto its image. More explicitly, a map $f : X \longrightarrow Y$ between topological spaces X and Y is an embedding if f yields a homeomorphism between X and $f(X)$ (where $f(X)$ carries the

subspace topology inherited from Y). Intuitively then, the embedding $f : X \longrightarrow Y$ lets us treat X as a subspace of Y.

However, the theoretical difficulties find in the study of embedded topological spaces are considerably high. Therefore, in this report, only the abstract case is considered; i.e, simplicial complexes are studied without consider them embedded in a space.

As a consequence, geometrical objects, that considered embedded in $\mathbb{R}^3$ are intuitively different, can become indistinguishable in a topological sense.

Example 1. Embedding in $\mathbb{R}^2$

Let X be the topological space formed by two disjoint circles $X = S^1 \sqcup S^1$ and the next embeddings of X in $\mathbb{R}^2$ (figure 1):

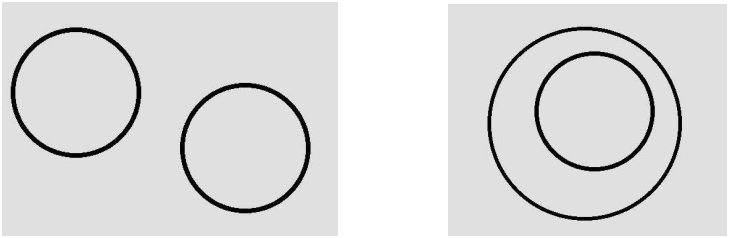


Figure 1.3: Two different embeddings of $S^1 \sqcup S^1$ in $\mathbb{R}^2$

In figure 1.3(a), one would say that the two a circles are 'separated' and in the second embedding, that one is inside the other. However, if the space X is considered abstractly, the differences between the embeddings presented in 1.3(a) and 1.3(b) are not made.

Example 2. Another embedding

Now, as another example, consider the map on a sphere that consists in identifying (or collapsing) two different points. Intuitively, there are 'two possible ways' of collapsing the points. One way is to collapse the two points covering the segment *inside* the sphere that unites them. The other is to cover the outer space of the sphere.

The two figures obtained are visually different, but topologically identical because they are the result of the same construction. The two representations (embeddings) are different, but abstractly, they are the same topological space.

Chapter 2

Simplicial homology

2.1 Chain complexes

The polytope was defined (section) as the underlying topological space of a simplicial complex. Through the combinatorial and discrete nature of a simplicial complex, an algebraic structure can be provided to these topological spaces. This algebraic structure is called chain complex.

This definition is found in [SR06, page 2].

Definition 8. Chain complex.
It is a sequence $(C_k, d_k)_{k\in\mathbb{Z}}$ *where:*

1. $\forall k \in \mathbb{Z}$, C_k *is a* V*-module called* ***chain group*** *of degree* k.
2. $\forall k \in \mathbb{Z}$, d_k *is a module morphism* $d_k : C_k \longrightarrow C_{k-1}$, *called* ***differential map***.
3. $\forall k \in \mathbb{Z}$, $d_k d_{k+1} = 0$
$$(C_*, d) : \ldots \xleftarrow{d_{k-2}} C_{k-2} \xleftarrow{d_{k-1}} C_{k-1} \xleftarrow{d_k} C_k \xleftarrow{d_{k+1}} C_{k+1} \xleftarrow{d_{k+2}} C_{k+2} \xleftarrow{d_{k+3}} \ldots$$

Definition 9. subchain complex, definition.
Let (C_*, d) *be a chain complex. A* ***subchain complex*** D *of* C_* *is a chain complex whose* k*th chain group is a subgroup of* C_k*, and whose boundary operator in each dimension* k *is the restriction of* d_k

To be able to associate a chain complex with a simplicial complex, one must:

1. orient the simplicial complex (which will consist in an ordering of the vertices),
2. define a sequence of modules (which will be called module chains),
3. define a differential operator d so that $\forall k \in \mathbb{Z}$, $d_k d_{k+1} = 0$ (which will be called boundary operator).

Oriented simplicial complex

Definition 10. Oriented simplex.
Let σ be a simplex (either geometric or abstract). Define two orderings of its vertex set to be equivalent if they differ from one another by an even permutation. If $\dim(\sigma) > 0$, the orderings of the vertices of σ then fall into two equivalence classes. Each of these classes is called an ***orientation*** *of σ. (If σ is a 0-simplex, then there is only one class and hence only one orientation of σ). An* ***oriented simplex*** *is a simplex σ together with an orientation of σ. (See figure 2.1.)*

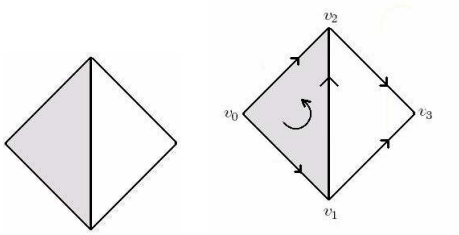


Figure 2.1: (a) simplicial complex; (b) oriented simplicial complex

Module of chains and boundary operator

Definition 11. Chain and module of k-chains.
Let X be a simplicial complex. A ***k-chain*** *on X is a function c from the set of oriented k-simplices of X to the integers, such that:*

1. *$c(\sigma) = -c(\sigma')$ if σ and σ' are opposite orientations for the same simplex.*

2. *$c(\sigma) = 0$ for all but finitely many oriented k-simplices σ.*

We add k-chains by adding these values; the resulting module is denoted $C_p(X)$ and is called the ***module of (oriented) p-chains*** *of X. If $k < 0$ or $k > dimX$, we let $C_p(X)$ denote the trivial group.*

By abuse of notation, we often use the symbol σ to denote not only a simplex, or an oriented simplex, but also to denote the elementary k-chain c corresponding to the oriented simplex σ. With this convention, if σ and σ' are opposite orientations of the same simplex, then we can write $\sigma' = -\sigma$, because this equation holds when σ and σ' are interpreted as elementary chains.

Lemma 2.1.1. *Considered as a group, $C_k(X)$ is free and commutative. A base for $C_k(X)$ can be obtained by orienting each p-simplex and using the corresponding elementary chains as a basis.*

Now lets define the differential morphism called boundary operator:
Definition 12. Boundary operator.
We now define a homomorphism

$$d_k = C_k(X) \longrightarrow C_{k-1}(X)$$

called the ***boundary operator****. If $\sigma = [v_0, \ldots, v_k]$ is an oriented simplex with $k > 0$, we define:*

$$d_k\sigma = d_k[v_0, \ldots, v_k] = \sum_{i=0}^{k}(-1)^i[v_0, \ldots, v_{i-1}, v_{i+1}, \ldots, v_k]$$

Note that, since $C_k(X)$ is the trivial group for $k < 0$, the operator d_k is the trivial homomorphism for $k \leq 0$.

Lemma 2.1.2. $d_{k-1} \circ d_k = 0$

Summarizing, we can represent all these structures in the next diagram:

$$(C_*, d) : 0 \xleftarrow{0} C_0 \xleftarrow{d_1} C_1 \xleftarrow{d_2} \ldots \xleftarrow{d_{n-1}} C_{n-1} \xleftarrow{d_n} C_n \xleftarrow{0} 0$$

Finally, note that, if n is the dimension of the simplicial complex, $C_k = 0$ for $k > n$.

2.1.1 Chain-complex morphisms

A chain-complex morphism is a map that relates two chain complexes and preserves their algebraic structure.
Definition 13. Chain complex morphism.
[SR06, page 10] Let $C_* = \{C_k, d_k\}$ and $D_* = \{D_k, d_k\}$ be two chain-complexes. A ***chain-complex morphism*** *$f : C_* \longrightarrow D_*$ is a collection of linear morphisms $f = \{f_k : C_k \longrightarrow D_k\}_k$ satisfying the differential condition: for every k, the relation $f_{k-1}\ d_k = d_k\ f_k$; ($df = fd$).*

A simplicial map between simplicial complexes induces a chain-complex morphism between their respective chain-complexes.

Definition 14. Chain complex morphism induced by a simplicial map.
Let $f : X \to Y$ be a simplicial map. If $v_0, \ldots, v_k$ are a simplex of X, then the points $f(v_0), \ldots, f(v_k)$ span a simplex of Y. We define a homomorphism $f_\# : C_k(X) \to C_k(Y)$ by defining it on oriented simplices as follows:

$$f_\#([v_0, \ldots, v_p]) = \begin{cases} [f(v_0), \ldots, f(v_p)] & \text{if } f(v_0), \ldots, f(v_p) \text{ are distinct;} \\ 0 & \text{otherwise.} \end{cases}$$

This map is clearly well-defined; exchanging two vertices in the expression $[v_0, \ldots, v_k]$ changes the sign of the right side of the equation. The family of homomorphism $\{f_\#\}$; one in each dimension, is called the ***chain-complex morphism induced by the simplicial map*** *f.*

Lemma 2.1.3. *The homomorphism $f_\#$ commutes with the border operator d, therefore, $f_\#$ is a chain-complex morphism.*

Example 3. Mayer-Vietoris chain-complex morphisms

Mayer-Vietoris relates the chain complexes $(A \cap B)_*$, $A_* \oplus B_*$, $(A \cup B)_*$. The chain complex $A_* \oplus B_*$ is called sum of chain complexes of A_* and B_* and its chains are $(\sigma, \widetilde{\sigma})$ with $\sigma \in A$ and $\widetilde{\sigma} \in B$. The chain $A_* \oplus B_*$ is originated from the disjoint union of the subcomplexes A and B, $A \sqcup B$.

For these chain complexes the following chain-complex morphisms are defined:

$$\begin{array}{rccc} i = i_A \oplus i_B : & (A \cap B)_* & \longrightarrow & A_* \oplus B_* \\ & \sigma & \longmapsto & (\sigma, \sigma) \end{array}$$

$$\begin{array}{rccc} j = j_A \ominus j_B : & A_* \oplus B_* & \longrightarrow & (A \cup B)_* \\ & (\sigma, \widetilde{\sigma}) & \longmapsto & \sigma - \widetilde{\sigma} \end{array}$$

Observe that i is the inclusion of a chain of the intersection $A \cap B$ on A and B. The chain-complex morphism j takes a chain from $A_* \oplus B_*$ and builds a chain of $(A \cup B)_*$ by changing the orientation of the subchain $\widetilde{\sigma} \in B$ and combining it with the subchain $\sigma \in A$.

Definition 15. Chain homotopy.

Let $f, g : X \to Y$ be simplicial maps. Suppose that for each k, one has a homomorphism

$$S : C_k(X) \to C_{k+1}(Y)$$

satisfying the equation

$$dS + Sd = g_\# - f_\#$$

then S is said to be a ***chain homotopy*** *between $f_\#$ and $g_\#$.*

Definition 16. Homotopy operator.

[SR06, page 11] Let $C_* = \{C_k, d_k\}$ and $D_* = \{D_k, d_k\}$ be two chain-complexes. A ***homotopy operator*** *$h : C_* \longrightarrow D_*$ is a collection of $h = \{h_k : C_k \longrightarrow B_{k+1}\}_k$ (noted $h : C_* \longrightarrow D_*$) of linear maps.*

2.2 Homology

Definition 17. Cycles, Boundaries, Homology.

The kernel of $d_k : C_k(X) \to C_{k-1}(X)$ is called the module of ***k-cycles*** *and denoted $Z_k(X)$. The image of $d_{k+1} : C_{k+1}(X) \to C_k(X)$ is called the module of* ***k-boundaries*** *and is denoted $B_k(X)$. Each boundary of a $k+1$ chain is a k-cycle. That is $B_k(X) \subset Z_k(X)$. We define*

$$H_k(X) = Z_k(X)/B_k(X)$$

and call it the kth ***homology group*** *of X.*

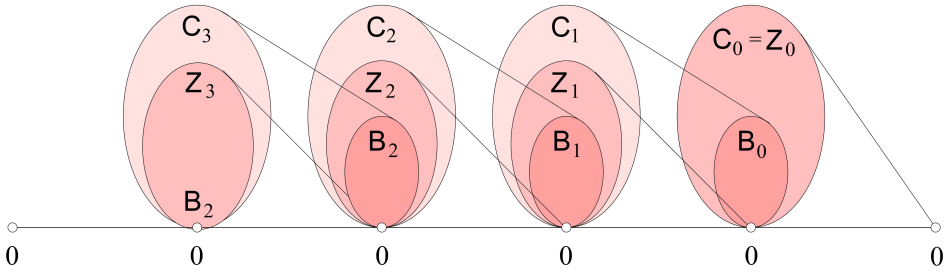


Figure 2.2: Representation of a chain complex of a three dimensional chain-complex
Author: Afra Zomorodian [Zom01]

2.2.1 Betti numbers and generators

Here, only finite simplicial complexes are considered, that means that the modules of their chain complexes are always finite. Those modules are also free by construction (canonical base formed by the cells at each dimension).

However, by performing the quotient to obtain the homology, one can obtain a group that is not free, i.e, H_k will be of the form:

$$H_k(X) = \overbrace{\mathbb{Z}[\gamma_1] \oplus \dots \mathbb{Z}[\gamma_s]}^{\text{free group}} \oplus \overbrace{\mathbb{Z}/\lambda_1\mathbb{Z}[\gamma_{s+1}] \oplus \dots \oplus \mathbb{Z}/\lambda_p\mathbb{Z}[\gamma_p]}^{\text{torsion group}}$$

where γ_i are a representative element of the class equivalence.

Definition 18. Betti numbers and torsion coefficients.
If the homology considered as a group:

$$H_k = \overbrace{\mathbb{Z} \oplus \dots \mathbb{Z}}^{s} \oplus \overbrace{\mathbb{Z}/\lambda_1\mathbb{Z} \oplus \dots \oplus \mathbb{Z}/\lambda_p\mathbb{Z}}^{p}$$

is called ***Betti numbers*** *the dimension of the free part of the homology (s in our notation) for each k.* $\lambda_1, \dots, \lambda_p$ *are the* ***torsion coefficients***

The information homology is interpreted as follows: the dimension of $H_0(X)$ indicates the number of connected components of the space X, the Betti number of $H_1(X)$ is the number of classes of non-contractible cycle, i.e, what intuitive corresponds to 'holes' on X. The Betti number of $H_2(X)$ tells the number of non-contractible surfaces; i.e, the number of 'voids' in X. $H_k(X)$ have analogous interpretation for $k > 2$.

The torsion coefficient has very particular interpretation, if a cycle σ is in a class of the torsion subgroup of H_k with torsion coefficient $\lambda > 1$, it means that σ is not a boundary, but $\lambda\sigma$ is a boundary. In the case, of closed manifold they have an immediate interpretation;

if a closed manifold has an homology group with a torsion subgroup, then the manifold is non-orientable (for example, the Klein bottle or the projective space).

Chapter 3

Classical methods for homology computation

3.1 Long exact sequence

This section is greatly inspired on [SR06, page 13-14]

3.1.1 Short exact sequence

Definition 19. Short exact sequence.
A short exact sequence is formed by three modules A, B, C and two maps $i : C \longrightarrow B$ and $j : B \longrightarrow A$ so that:

1. *i is injective.*
2. *j is surjective.*
3. *$im\ i = \ker j$*

It is represented by a diagram:

$$0 \longleftarrow A \xleftarrow{j} B \xleftarrow{i} C \longleftarrow 0$$

These properties imply the next relations:

$$C \cong \ker j$$

$$\boldsymbol{A \cong} \text{coker } i = B/\text{im } i \cong \boldsymbol{B/C} \quad (^1)$$

[1] if $f : C \to C'$ is a map between two modules C and C', then it is defined coker $f := C'/\text{im } f$

It is said that B is an **extension** of A and C. In general there are several possible extensions. It is called **extension problem** the problem of determining the right extension B when the modules A and C are known.

Example 4. Extension problem

Consider the following short exact sequence:

$$0 \longleftarrow \mathbb{Z}_2 \longleftarrow B \longleftarrow \mathbb{Z}_2 \longleftarrow 0$$

to which the morphism i, j are unknown. Then, there are two possible extensions for B; $\mathbb{Z}_2 \oplus \mathbb{Z}_2$ and $\mathbb{Z}_4$ which are not isomorphic.

3.1.2 Definition

Definition 20. (Long) exact sequence.

A sequence

$$... \xleftarrow{i_{k-1}} C_{k-1} \xleftarrow{i_k} C_k \xleftarrow{i_{k+1}} C_{k+1} \xleftarrow{i_{k+2}} ...$$

of modules and homomorphisms of modules is exact if im i_k = ker i_{k-1} $\forall k$.

From a short exact sequence of chain-complexes a long exact sequence can be built:

Theorem 3.1.1 (Long exact sequence from a short exact sequence)**.** *Given a short exact sequence of chain complexes:*

$$0 \longleftarrow A_* \xleftarrow{j} B_* \xleftarrow{i} C_* \longleftarrow 0$$

a canonical long exact sequence is defined:

$$\ldots \longleftarrow H_{k-1}(C_*) \xleftarrow{\partial} H_k(A_*) \xleftarrow{j} H_k(B_*) \xleftarrow{i} H_k(C_*) \xleftarrow{\partial} H_{k+1}(A_*) \longleftarrow \ldots$$

where ∂ is what is called ***connection morphism****.*

In the proof of this theorem, the connection morphism ∂ is built using a common technique in topology called *diagram chasing* (see [SR06, page 15])

Example 5. Mayer-Vietoris sequence

Consider a simplicial complex X with two subcomplexes A and B with $A \cap B \neq \emptyset$ and $A \cup B = X$. We can consider the simplicial maps: $i : A \cap B \longrightarrow A \sqcup B$ and $j : A \sqcup B \longrightarrow A \cup B$ (see precedent example).

We know that simplicial maps induce chain-complex morphisms. Notating them in the same manner, we obtain a short exact sequence:

$$0 \longleftarrow (A \cup B)_* \xleftarrow{j=j_A \ominus j_B} A_* \oplus B_* \xleftarrow{i=i_A \oplus i_B} (A \cap B)_* \longleftarrow 0$$

which, by the precedent proposition, induces a long exact sequence:

$$\ldots \longleftarrow H_{k-1}((A \cap B)_*) \xleftarrow{\partial} H_k((A \cup B)_*) \xleftarrow{j} H_k(A_* \oplus B_*) \xleftarrow{i} H_k((A \cap B)_*) \xleftarrow{\partial} H_{k+1}((A \cup B)_*) \longleftarrow \ldots$$

3.1.3 Technique to compute the homology

First, let us see how to build short exact sequences from a long exact sequence. Consider the long exact sequence:

$$... \xleftarrow{i_{k-1}} C_{k-1} \xleftarrow{i_k} C_k \xleftarrow{i_{k+1}} C_{k+1} \xleftarrow{i_{k+2}} ...$$

then the short sequence

$$0 \longleftarrow C_{k-1} \xleftarrow{i_k} C_k \xleftarrow{i_{k+1}} C_{k+1} \longleftarrow 0$$

is not exact because there is no reason for i_{k+1} to be injective and i_k to be surjective. These properties must be forced; one obtains a short exact sequence in the following way:

$$0 \longleftarrow \text{im } i_k \xleftarrow{i_k} C_k \xleftarrow{i_{k+1}} C_{k+1}/\ker i_{k+1} \longleftarrow 0$$

which is equivalent to:

$$0 \longleftarrow \ker i_{k-1} \xleftarrow{i_k} C_k \xleftarrow{i_{k+1}} \text{coker } i_{k+2} \longleftarrow 0$$

Therefore, if one knows the four modules surrounding a module (the two at left, C_{k+1}, C_{k+2}, and the two at right, C_{k-1}, C_{k-2}), the kernel of i_{k-1}, and the cokernel of i_{k+2}, then, the previous exact sequence can be built. Afterwards, to deduce C_k, there is still an extension problem that must be solved.

Now, the general technique for computing the homology of a chain complex B_* using a long exact sequence is as follows. Suppose that there are two chain complexes A_* and C_* with known homology and a short exact sequence:

$$0 \longleftarrow A_* \xleftarrow{j} B_* \xleftarrow{i} C_* \longleftarrow 0$$

Then, by the theorem 3.1.1, one knows that a canonical long exact sequence exists:

$$\dots \longleftarrow \underbrace{H_{k-1}(C_*)}_{known} \xleftarrow{\partial} \underbrace{H_k(A_*)}_{known} \xleftarrow{j} \underbrace{H_k(B_*)}_{unknown} \xleftarrow{i} \underbrace{H_k(C_*)}_{known} \xleftarrow{\partial} \underbrace{H_{k+1}(A_*)}_{known} \longleftarrow \dots$$

We have supposed that $H_k(C_*)$ and $H_k(A_*)$ are known for all k. Then short exact sequences can be constructed as explained previously. If one has enough information and can solve the extension problem of these sequences, then $H_k(B_*)$ can be determined.

Remark 3.1.2. *We have supposed that B_* is the chain complex in the center of the initial short exact sequence*

$$0 \longleftarrow A_* \xleftarrow{j} B_* \xleftarrow{i} C_* \longleftarrow 0$$

However, the method works in the same way if one wants to determine the homology of A_* or C_* provided that the homology of the other two chain-complexes are known.

Example 6. Determine the homology of the sphere using a long exact sequence

Let S^n be the n-sphere, $n > 1$ (S^n is homeomorphic to the n-simplex Δ^n). The goal is to compute its homology. Let S^n_+ the superior hemisphere of S^n and S^n_- its inferior hemisphere. Observe that $S^n = S^n_+ \cup S^n_-$ and $S^{n-1} = S^n_+ \cap S^n_-$. Then $\forall n > 0$ S^n_+ and S^n_- are homeomorphic to a disc, therefore they are contractile (i.e, they have the same homology of a point); $H_k(S^n_+) = H_k(S^n_-) = 0 \ \ \forall k > 0$ and $H_k(S^n_+) = H_k(S^n_-) = \mathbb{Z}$ for $k = 0$, and $H_0(S^n) = \mathbb{Z} \ \ \forall n > 0$ because S^n is connected for $n > 0$

Consider the following short exact sequence of Mayer-Vietoris:

$$0 \longleftarrow C(S^n)_* \xleftarrow{j} C(S^n_+)_* \oplus C(S^n_-)_* \xleftarrow{i} C(S^{n-1})_* \longleftarrow 0$$

Applying the theorem 3.1.1, one obtains the long exact sequence:

$$\ldots \underbrace{H_{k-1}(C(S^n_+)_* \oplus C(S^n_-)_*)}_{known} \xleftarrow{i} \underbrace{H_{k-1}(C(S^{n-1})_*)}_{unknown} \xleftarrow{\partial} \underbrace{H_k(C(S^n)_*)}_{unknown} \xleftarrow{j} \underbrace{H_k(C(S^n_+)_* \oplus C(S^n_-)_*)}_{known} \xleftarrow{i} \ldots$$

If the known homologies are substituted:

$$0 \longleftarrow H_0(S^n) = \mathbb{Z} \xleftarrow{j} H_0(C(S^n_+)_* \oplus C(S^n_-)_*) = \mathbb{Z}^2 \xleftarrow{i} H_0(C(S^{n-1})) = \mathbb{Z} \xleftarrow{\partial} H_1(S^n) \xleftarrow{j} 0 \xleftarrow{i} \ldots$$

$$\xleftarrow{j} 0 \xleftarrow{i} H_{k-1}(C(S^{n-1})_*) \xleftarrow{\partial} H_k(C(S^n)_*) \xleftarrow{j} 0 \ldots$$

$$0 \xleftarrow{i} H_{n-1}(C(S^{n-1})_*) \xleftarrow{\partial} H_n(C(S^n)_*) \xleftarrow{j} 0 \ldots$$

Therefore $\forall\ k > 1$ there is a short exact sequence:

$$0 \xleftarrow{i} H_{k-1}(C(S^{n-1})_*) \xleftarrow{\partial} H_k(C(S^n)_*) \xleftarrow{j} 0$$

this implies that $H_{k-1}(C(S^{n-1})_*) = H_k(C(S^n)_*)$.

Finally, one only needs to compute the homology of the following sequence:

$$0 \longleftarrow H_0(S^n) = \mathbb{Z} \xleftarrow{j} H_0(C(S^n_+)_* \oplus C(S^n_-)_*) = \mathbb{Z}^2 \xleftarrow{i} H_0(C(S^{n-1})) = \mathbb{Z} \xleftarrow{\partial} H_1(S^n) \xleftarrow{j} 0$$

Suppose that we have proven:

$$\text{im } i = \ker j = \{(\sigma, \widetilde{\sigma}),\ j((\sigma, \widetilde{\sigma}) = \sigma - \widetilde{\sigma} = 0\} = \mathbb{Z}$$

Therefore, we have the short exact sequence:

im $i = \mathbb{Z} \xleftarrow{i} H_0(C(S^{n-1})) = \mathbb{Z} \xleftarrow{\partial} H_1(S^n) \xleftarrow{j} 0$

In this case the extension problem is trivial and $H_1(S^n) = 0$. Summarizing, for $n > 1$, $H_{k-1}(C(S^{n-1})_*) = H_k(C(S^n)_*)$ for $k > 1$ and $H_1(S^n) = 0$.

Finally, for $n = 1$:

$$0 \longleftarrow H_0(S^1) = \mathbb{Z} \xleftarrow{j} H_0(C(S^1_+)_* \oplus C(S^1_-)_*) = \mathbb{Z}^2 \xleftarrow{i} H_0(C(S^0)) = \mathbb{Z}^2 \xleftarrow{\partial} H_1(S^1) \xleftarrow{j} 0$$

gives the short exact sequence:

$$\ker j = \mathbb{Z} \xleftarrow{i} H_0(C(S^0)) = \mathbb{Z}^2 \xleftarrow{\partial} H_1(S^1) \xleftarrow{j} 0$$

therefore $H_1(S^1) = \mathbb{Z}$.

$$H_n(C(S^n)_*) = \mathbb{Z}$$

3.1.4 Limitations: the extension problem

The technique for computing the homology based on the long exact sequence of the theorem 3.1.1 is **not an algorithm**. It is **not a constructive method**. Due to the extension problem, generally the unknown homology group cannot be obtained. .

Moreover, even if sometimes the homology group can be determined, generally the generators cannot.

This method is very useful for finding out the homology of particular spaces, but does not solve the general problem of determining the homology and the generators for a given space. Therefore, when the extension problem cannot be solved, other techniques must be sought.

In the next part, the theory of Constructive Homology is introduced. A constructive version of the theorem of the long exact sequence 3.1.1 will be presented, and we will use it to write a Constructive Mayer-Vietoris algorithm.

3.2 Smith Reduction algorithm

3.2.1 Presentation

The Smith Reduction is an algebraic method for computing the homology of a chain-complex. With this method, the homology of any **finite simplicial complex** X can be computed (at least theoretically).

This algorithm computes the Betti numbers, the torsion coefficients and the homology generators. For each k, it is obtained:

$$H_k(X) = \mathbb{Z}[\gamma_1] \oplus ... \oplus \mathbb{Z}[\gamma_r] \oplus \mathbb{Z}/\lambda_1\mathbb{Z}[\beta_1] \oplus ... \oplus \mathbb{Z}/\lambda_s\mathbb{Z}[\beta_s]$$

where r is the Betti number and $\lambda_1, ..., \lambda_s$ are the torsion coefficients. $\gamma_1, ..., \gamma_r$ are representatives generators of the free part of H_k (the part with coefficients in $\mathbb{Z}$) and $\beta_1, ..., \beta_s$ are representative generators of the torsion part of H_k (the part with coefficients in $\mathbb{Z}/\lambda_j\mathbb{Z}$ for some $\lambda_j > 1$).

In this chapter, the method is presented algebraically. **For a specific explication on the implementation and further details, see annex B at page 101.**

The goal of the method is to compute directly the quotient between the submodules $\ker d_k$ and im d_{k+1} that defines the homology:

$$H_k(X) = \ker d_k / \text{im } d_{k+1}$$

To compute the quotient H_k, one must be capable of:

1. determine:
   - the $\ker d_k$ for each k,
   - the im d_{k+1} for each k,
2. perform the quotient between them both.

The Smith Reduction algorithm determines explicitly the submodules im d_{k+1} and $\ker d_k$ of each k. Moreover, im d_{k+1} and $\ker d_k$ are expressed in a base that allows us to perform the quotient.

3.2.2 General description

In the Smith Reduction algorithm, the boundary operator d_k is expressed as a matrix D_k in a canonical base of the simplicial complex X for each dimension k. Then, a change of base is applied to obtain a matrix N_k with the following structure:

$$N_k = \begin{pmatrix} 0 & \boldsymbol{\lambda} & 0 \\ 0 & 0 & Id \\ 0 & 0 & 0 \end{pmatrix}$$

where

$$\boldsymbol{\lambda} = \begin{pmatrix} \lambda_r & 0 & 0 & 0 \\ 0 & \lambda_{r-1} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \lambda_1 \end{pmatrix}$$

is a diagonal matrix with $\lambda_r, ..., \lambda_1$ in the diagonal ($\lambda_i \in \mathbb{Z}$ and $\lambda_i > 1$ and λ_i divides λ_{i+1}). N_k is said to be in **Smith Normal Form** (See [Mun99, Ago76, Pel06])

With the Smith Reduction, the same chain-complex is obtained but expressed in a different base at each dimension; we call this new base **Smith base** . **The rows of the matrix**

N_{k+1} **are expressed in the same base than the columns of** N_k**, for each** k. We denote the matrix of change of base as $P_k : C_k \to C_k$:

$$D_k = (P_{k-1})N_k(P_k)^{-1}$$

Remark 3.2.1. *The method (one of them) to obtain the change of base matrices* P_k *is explained in annex B.*

We have the diagram:

$$\begin{array}{ccccccccc}
\cdots & \longleftarrow & X_{k-1}{=}[\boldsymbol{\sigma}^{k-1}] & \xleftarrow[D_k]{} & X_k{=}[\boldsymbol{\sigma}^k] & \xleftarrow[D_{k+1}]{} & X_{k+1}{=}[\boldsymbol{\sigma}^{k+1}] & \longleftarrow & \cdots \\
 & & \uparrow{\scriptstyle P_{k-1}} & & \uparrow{\scriptstyle P_k} & & \uparrow{\scriptstyle P_{k+1}} & & \\
\cdots & \longleftarrow & X_{k-1} & \xleftarrow[N_k]{} & X_k & \xleftarrow[N_{k+1}]{} & X_{k+1} & \longleftarrow & \cdots
\end{array}$$

where $\boldsymbol{\sigma}^k = [\sigma_1^k, \ldots, \sigma_{l(k)}^k]$ is the canonical base of X at dimension k. The chain-complex at the botton is expressed in the Smith base.

How to find out the homology

Using two consecutive incidence matrices in Smith Normal Form, N_k, N_{k+1}, one can determine the Betti numbers and the homology generators:

- the dimension of $\ker N_k$ is equal to the number p of zero-columns of N_k. Suppose that these columns are expressed in the base $\{\gamma_1^k, \ldots, \gamma_p^k\}$
- the dimension of $\text{im } N_{k+1}$ is equal to number q of non-zero-rows of N_{k+1}. Suppose that the rows with the $\lambda > 1$ coefficients are expressed in the base $\{\gamma_1^k, \ldots, \gamma_q^k\}$ and the rows with the coefficient 1 are expressed in the base $\{\gamma_{q+1}^k, \ldots, \gamma_t^k\}$

Therefore, the Betti number of H_k is the difference $p - (t - q)$ and

$$\ker N_k = [\gamma_1^k, \ldots, \gamma_p^k]$$

$$\text{im } N_{k+1} = [\lambda_q \gamma_1^k, \ldots, \lambda_1 \gamma_q^k, \gamma_{q+1}^k, \ldots, \gamma_t^k]$$

Observe that $\ker d_k = \ker N_k$ and $\text{im } N_{k+1} = \text{im } d_{k+1}$ are expressed explicitly and in the same base ($\text{im } d_{k+1} \subset \ker d_k$). Therefore, to perform the quotient is trivial:

$$H_k = \ker N_k / \text{im } N_{k+1} = \mathbb{Z}/\lambda_q\mathbb{Z}[\gamma_1^k] \oplus \ldots \oplus \mathbb{Z}/\lambda_1\mathbb{Z}[\gamma_q^k] \oplus \mathbb{Z}[\gamma_{(t-q+1)}^k] \oplus \ldots \mathbb{Z}[\gamma_p^k]$$

3.2.3 Limitations

Theoretically the Smith algorithm solves completely our problem of determining the homology of a finite simplicial complex. Unfortunately, the complexity of this algorithm makes it impossible to use it in practice. Firstly, the process of transforming a matrix into its Smith Normal Form requires a lot of operations. Secondly, when we are working with simplicial complexes with thousands of vertices, the dimension of the matrices overflows the memory.

Consequently, this method can only be used in 'small' cases. That is why is important to have an algorithm that computes the union of simplices without repeating the computation.

For a survey of the existing optimizations refer to Section 0.3.

Conclusion of part I

If we retake the schema presented in the figure 4, we have an overview of the relations of all the concepts presented in this part.

The concept of simplicial complex has been defined. Simplicial homology, which is the property of simplicial complexes that we want to find out, has been explained. The classical methods to compute the homology has been described. We have seen the limitations of these methods.

In the next part, a constructive version of the long exact sequence is presented which will solve the extension problem, and a reduction of the size of the incidence matrices is proposed, which will reduce the number of operations in the Smith Reduction algorithm.

As a final remark, there exists another classical method to compute the homology called **spectral sequences**. It was invented after the long exact sequence method, but as it predecessor, it is not a constructive method. We do not explain here the notion of spectral sequence because it is not used in our application and its theoretical basis are not involved in our methods. For a description of spectral sequences see [SR06, Section 3.2].

Part II

Constructive Homology

In this part, we present the concepts of Constructive Homology used in the **Constructive Mayer-Vietoris algorithm**, namely: the reduction, the cone of a morphism, and the effective short exact sequence of Mayer-Vietoris.

In the previous part, the limitations of the classical methods to compute the homology of a simplicial complex have been explained. On one hand, the long exact sequence is not constructive and, due to the extension problem, the method is, in general, ineffective. On the other hand, the Smith Reduction algorithm can compute the homology of any finite simplicial complex but its computational complexity makes it applicable only to small simplicial complexes.

In this part, using the methods of Constructive Homology:

- a constructive version of the long exact sequence is presented, which will solve the extension problem, and, consequently, a **constructive version of the Mayer-Vietoris long exact sequence** is proposed (**effective short exact sequence and Lemma 82**).
- a method to **reduce the size of the incidence matrices** is explained (**Homological Smith Reduction**) to make applicable the Smith Reduction algorithm.

The scheme in the figure 3.1 presents the main concepts of this part.

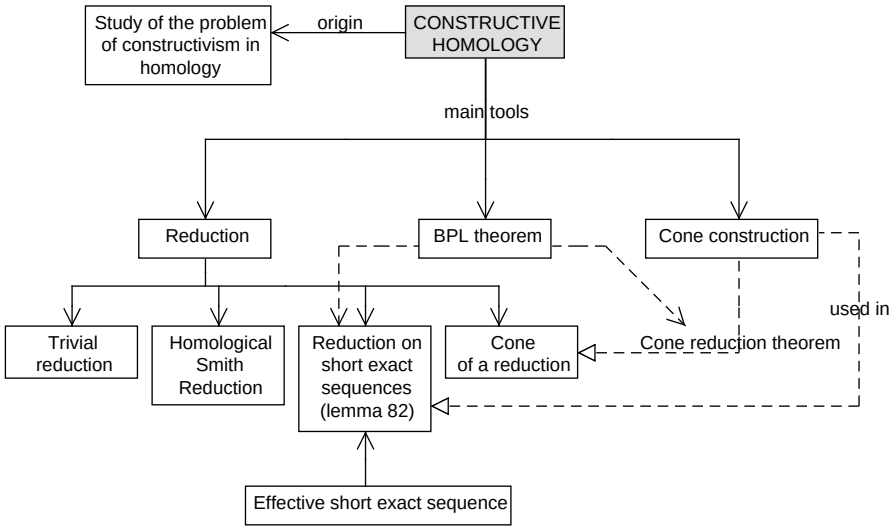


Figure 3.1: Diagram of concepts of Constructive Homology. *The BPL theorem (Basic Perturbation Lemma) is the key result to prove the Cone Reduction Theorem and Lemma 82.*

Chapter 4

The constructiveness problem

Constructive Homology is born from the study of the constructiveness problem of homology. The lack of constructiveness in homology has its roots in the definitions that use existential quantifiers ($\exists$) [SR06], like in the definition of a boundary:

$$c \in C_k \text{ is a boundary if } \exists c' \in C_{k+1} \text{ so that } d_{k+1}(c') = c$$

This definition requires to proof the existence of such a chain c' to state that c is a boundary. However, this definition does not require to find a particular chain c'. Therefore, the methods in classical homological algebra, prove the existence of the chain c' without giving an effective method to find it. The consequence is the non-constructiveness of the methods to compute the homology, like in the case of the long exact sequence.

In [SR06, section 4.1] there is a beautiful introduction to constructive mathematics. By a simple example, it is explained how classical homology is based in the logic axioms of Zermelo-Fraenkel and why non-constructive theories are born from them.

Constructive Homology studies why classical homology is not constructive and provides the elements to make it constructive. The constructiveness requirements are gathered in the **solution S of the homological problem for a chain complex** (see [SR06]); from it, the concept of **reduction**, presented in the next chapter, is derived.

The reduction concept goes beyond to what it is explained in this document. In fact, with the help of the reduction, Sergeraert et. al. have been able to compute the homology of chain complexes with modules of *infinite* dimension.

In this project, a particular application of constructive homology is presented. However, the potential of this theory goes far beyond.

Let us finish with the following citation, from Sergeraert's first paper [Ser94] on the computability problem of homology:

"Hilbert's dream actually was a dream. But a new subject for research was thus opened: studying what mathematical theories are in fact opened to algorithmic processes.[...] This

paper is devoted to such a result. Here it is proved that almost every reasonable computability problem in homological algebra and algebraic topology has a positive solution."

Chapter 5

Reduction

5.1 Intuitive introduction

The reduction concept is introduced intuitively as a **simplification**. In the case of a simplicial complex $\widetilde{X}$, a simplification means to obtained a reduced simplicial complex X by elimination of cells that preserve the homology of $\widetilde{X}$; i.e, X and $\widetilde{X}$ have *equivalent homology.* Classically, the basic operation to 'simplify' a space is to perform consecutive retractions (see figure 5.1). *Simplification*

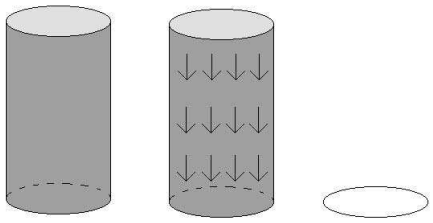


Figure 5.1: Retraction of the cylinder onto a circle

However, a reduction operates directly on the chain complex and not on the simplicial complex; i.e, a reduction of a chain-complex $\widetilde{C}_*$ is another chain-complex C_* with equivalent homology (and with modules of inferior dimension that $\widetilde{C}_*$). How such a reduction can be obtained?

On one hand, a method of simplification can be conceived by answering the next question: *First viewpoint*
What is it needed in order to determine the homology H_k at a given dimension k?
A group of generators and their coefficient spaces.

Concretely, one needs the generators of the free part of H_k (that have coefficients in $\mathbb{Z}$) and the generators of the torsion group of H_k (that have coefficients in $\mathbb{Z}/\lambda\mathbb{Z}$ for some torsion coefficient $\lambda > 1$) along with their torsion coefficients.

Definition 21. Homological cycles, weak-boundaries (notation).
We will call ***homological cycles*** *the generators of the free part of H_k and* ***weak boundaries*** *the generators of the torsion part of H_k.*

If the modules of a chain complex $\widetilde{C}_*$ are expressed in bases where the homological cycles and the weak-boundaries are explicit and closed by the boundary operator (i.e, their image by the boundary operator are homological cycles and weak-boundaries of inferior dimension), then a new chain-complex C_* can be built formed by the subbases $\boldsymbol{\gamma}$ constituted by the homological cycles and the weak-boundaries. The boundary operator of C_* would be the boundary operator of $\widetilde{C}_*$ restricted to the subbases $\boldsymbol{\gamma}$.

Second viewpoint

On the other hand, another method of simplification can be conceived by answering the next question:
How homology H_k is defined for a given dimension k?

$$H_k = \ker d_k / \text{im } d_{k+1}$$

where d_* is the boundary operator.

Definition 22. Boundary, pure-boundary.
We defined in part I of this document a boundary as a chain σ that $\sigma \in B_{k+1} := im\ d_{k+1}$. Here we abuse notation and call a ***boundary*** *an element of B_{k+1} that is not a weak-boundary; i.e, a boundary is a chain $\sigma \in B_{k+1} := im\ d_{k+1}$ so that not exist a chain $\widetilde{\sigma}$ and $a\lambda \in \mathbb{Z}$, $\lambda > 1$, $\sigma = \lambda\widetilde{\sigma}$. In the cases, where it can be confused with the classical definition of boundary it will be called* ***pure-boundary****.*

When the quotient $H_k = \ker d_k / \text{im } d_{k+1}$ is performed, the cycles that are boundaries are 'canceled'. Therefore, the simplification consists in the following: given all the cycles, ignore those that are boundaries and then perform the quotient of modules to obtain H_k. With the previous simplification, the quotient will only be performed over the weak-boundaries because the boundaries would have already been eliminated.

Summing up, in this case, the idea is to simplify a chain-complex $\widetilde{C}_*$ by 'eliminating' its boundaries.

These are intuitive approaches of what will be detailed in the next chapters; a special kind of reduction (simplification) using the Smith Reduction, the Homological Smith Reduction.

First, let us formalise the concept of reduction as defined by Sergeraert et. al. in [SR06].

5.2 Formal definition

The reduction concept is formalised. The definition and proposition of this section are from [SR06, Section 4.6].

Definition 23. Reduction.

A ***reduction*** $\rho : \widehat{C}_* \Rrightarrow C_*$ *is a diagram:*

$$\rho = \quad \widehat{C}_* \circlearrowright h \qquad g \uparrow \downarrow f \qquad C_*$$

where:

1. $\widehat{C}_*$ *and* C_* *are chain-complexes.*
2. f *and* g *are chain-complex morphisms (def. in page 25).*
3. h *is a homotopy operator (def. in page 26).*
4. *These relations are satisfied:*
   (a) $fg = id_{C_*}$
   (b) $gf + dh + hd = id_{\widehat{C}_*}$
   (c) $fh = hg = hh = 0$

A reduction is a particular **homology equivalence** between a **big chain-complex** $\widehat{C}_*$ and a **small one** C_*.

Proposition 5.2.1. *Let* $\rho : \widehat{C}_* \Rrightarrow C_*$ *be a reduction. This reduction is equivalent to a decomposition:* $\widehat{C}_* = A_* \oplus B_* \oplus C'_*$*:*

1. $\widehat{C}_* \supset C'_* = im\ g$ *is a subcomplex of* $\widehat{C}_*$
2. $A_* \oplus B_* = \ker f$ *is a subcomplex of* $\widehat{C}_*$.
3. $\widehat{C}_* \supset A_* = \ker f \cap \ker h$ *is not in general a subcomplex of* $\widehat{C}_*$.
4. $\widehat{C}_* \supset B_* = \ker f \cap \ker d$ *is a subcomplex of* $\widehat{C}_*$ *with null differentials.*
5. *The chain-complex morphisms* f *and* g *are inverse isomorphisms between* C'_* *and* C_*.
6. *The arrows* d *and* h *are module isomorphisms of respective degrees -1 and +1 between* A_* *and* B_*.

Observe that:

- B_k is a collection of cycles so that $B_k \subset \mathrm{im}\ d_k$,
- there exists a bijection between A_{k+1} and B_k through d and h,
- **the component** C'_k **is a copy of** C_k **and their homological natures therefore are the same.**

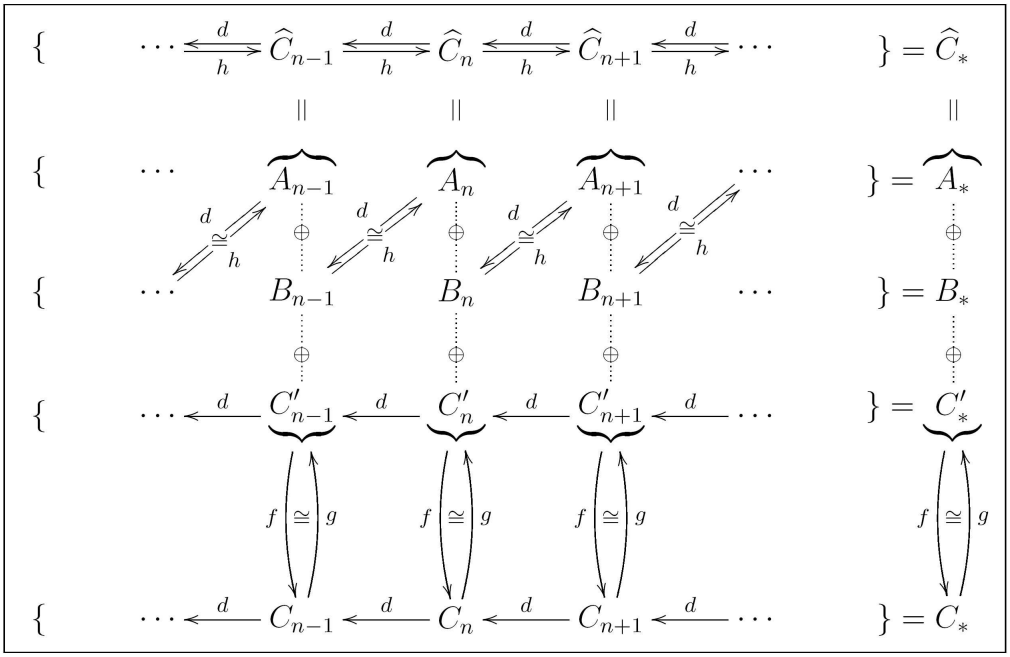


Figure 5.2: Reduction diagram (from [SR06])

Example 7. Trivial example of a reduction

There is a reduction that fulfills trivially all the conditions to be a reduction, i.e, a reduction that does 'nothing':

Definition 24. Trivial reduction.

Given a chain complex C_*, its trivial reduction ρ is a reduction where the small chain is C_* itself, f and g are morphisms identity and h are 0 morphisms.

Proposition 5.2.2. *The trivial reduction of a chain is a reduction.*

Proof. Trivially all conditions in the definition of a reduction are fulfilled. □

The definition of a reduction is quite abstract. Next, we present a particular case of reduction called **Homological Smith Reduction**.

5.3 Special case: Homological Smith Reduction

As explained in the intuitive introduction, supposed that, for a chain-complex $\widehat{C}_*$, there exist bases in which the homological cycles and the weak-boundaries are expressed explicitly (and these elements are closed by the boundary operator). Then, we can build a new chain

complex C_*, its modules would be expressed in the bases $\boldsymbol{\gamma}$ constituted by the homological cycles and the weak-boundaries. The boundary operator of C_* would be the boundary operator of $\widehat{C}_*$ restricted to the subbases $\boldsymbol{\gamma}$. In this chapter, we follow this idea.

5.3.1 Base classification

Suppose that the incidence matrices $(D_k)_{k=1,\dots,n}$ of a chain complex $\widehat{C}_*$ are expressed in Smith Normal Form N_k for each k (as explained in 3.2). Then, we perform a classification of the bases in which the matrices N_k are expressed.

Consider two consecutive incidence matrices in Smith normal form, N_k, N_{k+1}. Suppose that $\{\gamma_1, \dots, \gamma_{l(k)}\}$ is the base in which are written the columns of N_k and the rows of N_{k+1}.

Suppose that:

- the zero columns of N_k are expressed in the subbase $\{\gamma_1, \dots, \gamma_\bullet\}$. Then $\ker N_k = \ker d_k = [\gamma_1, \dots, \gamma_\bullet]$,
- the columns of N_k with a coefficient 1 are expressed in the subbase $\boldsymbol{pb}^k = \{pb_1^k, \dots, pb_\bullet^k\}$, and that the columns with a $\lambda > 1$ are expressed in the subbase of $\boldsymbol{pw}^k = \{pw_1^k, \dots, pw_\bullet^k\}$.

We split now the subbase $\ker d_k = [\gamma_1, \dots, \gamma_\bullet]$. Looking at the matrix N_{k+1}, let the rows of D_{k+1} with a coefficient 1 be expressed in the subbase $\boldsymbol{b} = \{b_1^k, \dots, b_\bullet^k\}$, and the rows with a $\lambda > 1$ be expressed in the subbase $\boldsymbol{w} = \{w_1^k, \dots, w_\bullet^k\}$, the rest of the subbase of $\ker d_k$ is the subbase of $\boldsymbol{c} = \{c_1^k, \dots, c_\bullet^k\}$

Definition 25. Weak-boundaries, boundaries, homological cycles, pre-weak-boundaries, pre-boundaries.

With the previous notations, the chains of submodule generated by $\boldsymbol{w}$ *is called* ***weak-boundaries****; those generated by* $\boldsymbol{b}$ *are called* ***boundaries****; those generated by* $\boldsymbol{c}$ *are called* ***homological cycles****; those generated by* $\boldsymbol{pw}$ *are called* ***pre-weak-boundaries****; and finally, those generated by* $\boldsymbol{pb}$ *are called* ***pre-boundaries****.*

Example 8. base classification

$$N_k = \begin{array}{c} \begin{array}{cccccccccc} \gamma_1 & \gamma_2 & \gamma_3 & \gamma_4 & \gamma_5 & \gamma_6 & \gamma_7 & \gamma_8 & \gamma_9 & \gamma_{10} \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \qquad N_{k+1} = \begin{array}{c} \begin{array}{c} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \gamma_4 \\ \gamma_5 \\ \gamma_6 \\ \gamma_7 \\ \gamma_8 \\ \gamma_9 \\ \gamma_{10} \end{array} \\ \begin{pmatrix} 0 & 0 & 6 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Imagine that the columns of N_k and the rows of N_{k+1} are expressed in the base $\{\gamma_1, \dots, \gamma_{10}\}$. Then:

- Observe the matrix N_{k+1}
  - The rows with a 1 in them are expressed in the base $\{\gamma_3, \gamma_4, \gamma_5\} \Rightarrow \boldsymbol{b} = \{\gamma_3, \gamma_4, \gamma_5\}$
  - The rows with a $\lambda > 1$ are rows 1 and 2 with $\lambda = 2, 6$, respectively. This rows are expressed in the subbase $\{\gamma_1, \gamma_2\} \Rightarrow \boldsymbol{w} = \{\gamma_1, \gamma_2\}$
- Observe matrix N_k.
  - The null columns of N_k (the $\ker d_k$) is generated by $\{\gamma_1, \ldots, \gamma_7\}$. $\boldsymbol{c}$ is formed by $\{\gamma_1, \ldots, \gamma_7\} \backslash (\boldsymbol{b} \cup \boldsymbol{w}) = \{\gamma_6, \gamma_7\}$
  - The columns of N_k with a 1 are expressed in the subbase $\{\gamma_9, \gamma_{10}, \gamma_{11}\} \Rightarrow \boldsymbol{pb} = \{\gamma_9, \gamma_{10}\}$
  - The columns of D_k with $\lambda > 1$ is only column 8 with $\lambda = 3 \Rightarrow \boldsymbol{pw} = \{\gamma_8\}$

Result

$\boldsymbol{pb}$	$\boldsymbol{b}$	$\boldsymbol{pw}$	$\boldsymbol{w}$	$\boldsymbol{c}$
γ_9, γ_{10}	$\gamma_3, \gamma_4, \gamma_5$	γ_8	γ_1, γ_2	γ_6, γ_7

The torsion coefficients are 2 and 6.

Remark 5.3.1. *The base of vertices and the base of the dimension n of the chain complex are particular cases. One must perform the classification in the same way but keeping in mind that the morphisms at dimension* 0 *and at dimension* $n+1$ *are the zero morphisms. Therefore, in the base of dimension n there are only cycles, pre-boundaries and pre-weak-boundaries but not weak-boundaries nor boundaries. In the vertices base there are not pre-boundaries, weak-boundaries nor pre-weak-boundaries. There are only cycles and boundaries.*

where $\boldsymbol{pb}^k$ is what we called the 'pre-boundaries'; $\boldsymbol{b}^k$, the 'boundaries'; $\boldsymbol{pw}^k$, the pre-weak-boundaries; $\boldsymbol{w}^k$, the weak-boundaries; $\boldsymbol{c}^k$, the homological cycles.

5.3.2 Chain-complex reduction

We have seen that the incidence matrices D_k in Smith Normal Form, N_k, are expressed in the bases $\{\boldsymbol{w}^k, \boldsymbol{b}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k, \boldsymbol{pb}^k\}$. We also know the relation between these subbases in N_k:

Subbase on the columns	Corresponding image subbase on the rows	Submatrix
$\boldsymbol{pb}^k$	$\boldsymbol{b}^{k-1}$	Id
$\boldsymbol{pw}^k$	$\boldsymbol{w}^{k-1}$	Diagonal matrix with the torsion coefficients $\boldsymbol{\lambda}$

The form of the Smith matrix is as follows:

$$N_k = \begin{array}{c} \\ \boldsymbol{w}^{k-1} \\ \boldsymbol{b}^{k-1} \\ \boldsymbol{c}^{k-1} \\ \boldsymbol{pw}^{k-1} \\ \boldsymbol{pb}^{k-1} \end{array} \begin{array}{c} \begin{array}{ccccc} \boldsymbol{w}^k & \boldsymbol{b}^k & \boldsymbol{c}^k & \boldsymbol{pw}^k & \boldsymbol{pb}^k \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & \boldsymbol{\lambda} & 0 \\ 0 & 0 & 0 & 0 & Id \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

The goal is to construct a new chain complex EC_* from the original one C_* by reduction. To build a new chain complex EC_*, we need a base for each dimension and a border operator. Since we want a chain complex with equivalent homology, we need to take the elements $\boldsymbol{w}^k, \boldsymbol{c}^k$ for each dimension (weak-boundaries and homological cycles). However, observing the matrix, we see that $\boldsymbol{w}^{k-1}$ is related to $\boldsymbol{pw}^k$ by the torsion coefficients submatrix $\boldsymbol{\lambda}$. We need to keep this submatrix and, therefore, also the subbase $\boldsymbol{pw}^k$ for each k.
In the end, the only subbase that we are not considering are $\boldsymbol{pb}^k$ and $\boldsymbol{b}^k$.
Observe that:

- the submodule generated by $\boldsymbol{pb}^k$ is isomorph to the submodule generated by $\boldsymbol{b}^{k-1}$ (since the identity submatrix relates them),
- the subbasis $\boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k$ are closed by the boundary operator; i.e, the image of this subbasis by the boundary operator is included in the subbasis $\boldsymbol{w}^{k-1}, \boldsymbol{c}^{k-1}, \boldsymbol{pw}^{k-1}$ of inferior dimension.

Definition 26. Small chain-complex of the Homology Smith Reduction.
With the previous notations, the reduced chain-complex EC_* of the Homological Smith Reduction is defined as follows:

- *at each dimension k, the k-module of EC_* is defined as:*

$$EC_k := [\boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k]$$

- *at each dimension k, its incidence matrix is:*

$$EN_k := \begin{array}{c} \\ \boldsymbol{w}^{k-1} \\ \boldsymbol{c}^{k-1} \\ \boldsymbol{pw}^{k-1} \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{w}^k & \boldsymbol{c}^k & \boldsymbol{pw}^k \end{array} \\ \begin{pmatrix} 0 & 0 & \boldsymbol{\lambda} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

Remark 5.3.2. *It is immediate to prove that $N_{k-1}N_k = 0 \ \forall k$, therefore, EC_* is, effectively, a chain-complex.*

Definition 27. Smith base and Reduced Smith base.
We define, with the previous notations, the ***Smith base*** *as the base* $\{\boldsymbol{w}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{pw}, \boldsymbol{pb}\}$ *and the* ***Reduced Smith base*** *as the base* $\{\boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k\}$.

5.3.3 The Homological Smith Reduction algorithm

The algorithm to obtain the Homological Smith Reduction is as follows:

Input: a list of incidence matrices $(D_k)_{k=1,\ldots,n}$ of a chain complex $\widehat{C}_*$

1. For k from 1 to n, transform the matrix D_k into its Smith Normal Form N_k and construct the matrix of change of base P_k, P_k^{-1} as explained in annex B.
2. Perform the Smith base classification of N_k $\boldsymbol{w}^k, \boldsymbol{b}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k, \boldsymbol{pb}^k$ for each k.
3. Using this classification construct the matrices EN_k for each k.

Finally, we define:
Definition 28. Homological Smith Reduction.
Using the previous notations, given a chain-complex $\widetilde{C}_*$*, its Homological Smith Reduction is:*

$$\rho = \quad \widehat{C}_* \circlearrowright h \qquad g \uparrow \downarrow f \qquad EC_*$$

where EC_* *is the small chain of the Homological Smith Reduction (definition 26):*

$$EC_* = [\boldsymbol{w}_*, \boldsymbol{c}_*, \boldsymbol{pw}_*]$$

and

$$f : \widehat{C}_* \to EC_*, \quad f = \{f_k = r_k(P_k)^{-1} = (P_k)^{-1}|_{\boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k}\}$$

$$g : EC_* \to \widehat{C}_*, \quad g = \{g_k = P_k r_k^T = P_k|_{\boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k}\}$$

$$h : \widehat{C}_* \to \widehat{C}_{*+1}, \quad h = \{h_k = P_k \widetilde{h}_k (P_{k-1})^{-1}\}$$

(Observe that the restrictions on the matrices sometimes is carried out on the columns and sometimes on the rows (which is the case of the inverses).)

The matrix r_k *is the projection of the Smith base* $\{\boldsymbol{w}^k, \boldsymbol{b}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k, \boldsymbol{pb}^k\}$ *into the Reduced Smith one* $\{\boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k\}$*. So, it is defined as follows over the Smith base:*

$$r_*(u) = \begin{cases} 0 & \text{if } u \in \boldsymbol{pb}_*, \boldsymbol{b}_*; \\ u & \text{if } u \in \boldsymbol{w}_*, \boldsymbol{c}_*, \boldsymbol{pw}_*. \end{cases}$$

and extended linearly.

In matrix form:[1]

$$r_k = \begin{array}{c} \\ \boldsymbol{w}^k \\ \boldsymbol{c}^k \\ \boldsymbol{pw}^k \end{array} \begin{array}{c} \begin{array}{ccccc} \boldsymbol{w}^k & \boldsymbol{b}^k & \boldsymbol{c}^k & \boldsymbol{pw}^k & \boldsymbol{pb}^k \end{array} \\ \begin{pmatrix} Id & 0 & 0 & 0 & 0 \\ 0 & 0 & Id & 0 & 0 \\ 0 & 0 & 0 & Id & 0 \end{pmatrix} \end{array}$$

The morphism h is defined by the matrix resulting of the following matrix product:

$$h_k = \begin{array}{c} \\ \sigma_1^k \\ \vdots \\ \sigma_{l(k)}^k \end{array} \begin{array}{c} \boldsymbol{pb}^k \\ \begin{pmatrix} \\ P_k|_{\boldsymbol{pb}} \\ \\ \end{pmatrix} \end{array} * \boldsymbol{b}^{k-1} \begin{pmatrix} \sigma_1^{k-1} & \cdots & \sigma_{l(k-1)}^{k-1} \\ & P_{k-1}^{-1}|_{\boldsymbol{b}^{k-1}} & \end{pmatrix}$$

In fact,

$$\widetilde{h}_k = \begin{array}{c} \\ \boldsymbol{w}^k \\ \boldsymbol{b}^k \\ \boldsymbol{c}^k \\ \boldsymbol{pw}^k \\ \boldsymbol{pb}^k \end{array} \begin{array}{c} \begin{array}{ccccc} \boldsymbol{w}^{k-1} & \boldsymbol{b}^{k-1} & \boldsymbol{c}^{k-1} & \boldsymbol{pw}^{k-1} & \boldsymbol{pb}^{k-1} \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & Id & 0 & 0 & 0 \end{pmatrix} \end{array}$$

[1] We define r_k^T as $r_k^T = \begin{array}{c} \\ \boldsymbol{w}^k \\ \boldsymbol{b}^k \\ \boldsymbol{c}^k \\ \boldsymbol{pw}^k \\ \boldsymbol{pb}^k \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{w}^k & \boldsymbol{c}^k & \boldsymbol{pw}^k \end{array} \\ \begin{pmatrix} Id & 0 & 0 \\ 0 & 0 & 0 \\ 0 & Id & 0 \\ 0 & 0 & Id \\ 0 & 0 & 0 \end{pmatrix} \end{array}$

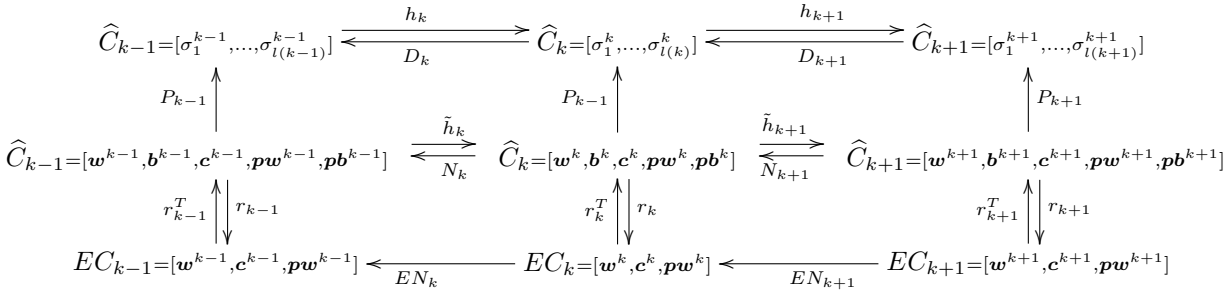


Figure 5.3: Schema of the Homological Smith Reduction

We have to prove that the Homological Smith Reduction is effectively a reduction, $\rho = (f, g, h) = \widehat{C}_* \Rrightarrow EC_*$

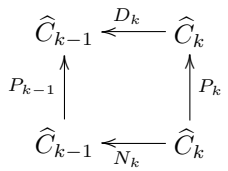
Proposition 5.3.3 (The Homological Smith Reduction is a reduction)**.**

Proof. Lets prove that these elements fulfilled the properties of the definition of reduction.

1. $\widehat{C}_*, EC_*$ **are chain-complexes.**
   $(\widehat{C}_*, \widehat{d}_*)$ is a chain-complex by hypothesis.
   We define $C_k = [\boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k]$ and $d_k = f_{k-1}\widehat{d}_k g_k$.
   By construction $d_k(EC_k) \subset d_{k-1}$. Then, EC_k define a module for each k, its boundary operator d_* is well defined (is closed) and $d_{k-1} \circ d_k = 0$ for each k. So, EC_*, d_* is a chain-complex.[2]

2. ***f* and *g* are chain-complex morphisms.**
   We want to prove that $f = \{f_k : \widehat{C}_k \to EC_k\}$ is a linear morphism and $f_{k-1}d_k = d_k f_k$. Analogously for g.

   - f is a linear morphism because is composition of linear morphism (change of base and projection), we have already expressed it as a product of matrices. $f_k = r_k P_k^{-1}$.
   - Observe that:



[2] $d_{k-1} \circ d_k = f_{k-2}\widehat{d}_{k-1}g_{k-1}f_{k-1}\widehat{d}_k g_k = f_{k-2}\widehat{d}_{k-1}(P_{k-1})r_{k-1}^T r_{k-1}(P_{k-1})^{-1}\widehat{d}_k g_k = f_{k-2}\widehat{d}_{k-1}\widehat{d}_k g_k = 0.$

the diagram commutes by definition; $N_k = (P_{k-1})^{-1} D_k P_k$ (since the change of base is an isomorphism with the same chain-complex). So we only need to prove that the next diagram commutes and we will have proved the general commutativity:

$$\begin{array}{ccc} \widehat{C}_{k-1} & \xleftarrow{N_k} & \widehat{C}_k \\ {\scriptstyle r_{k-1}}\downarrow & & \downarrow{\scriptstyle r_k} \\ EC_{k-1} & \xleftarrow[EN_k]{} & EC_k \end{array}$$

We will see the commutativity for the elements of the base (Smith base of $\widehat{C}_*$). We know that for $u \in \{\boldsymbol{pb}_*, \boldsymbol{b}_*, \boldsymbol{pw}_*, \boldsymbol{w}_*, \boldsymbol{c}_*\}$:

$$N_*(u) = \begin{cases} b_j^{*-1} & \text{if } u = pb_j^*; \\ \lambda^i w_i^{*-1} & \text{if } u = pw_i^*; \\ 0 & \text{elsewhere.} \end{cases}$$

$$r_{*-1}N_*(u) = \begin{cases} \lambda^i w^{*-1} & \text{if } u = pw_i^*; \\ 0 & \text{elsewhere.} \end{cases}$$

$$N_* r_{*-1}(u) = \begin{cases} \lambda^i w_i^{*-1} & \text{if } u = pw_i^*; \\ 0 & \text{elsewhere.} \end{cases}$$

So $r_{*-1}N_*(u) = N_* r_{*-1}(u)$ for each element of the base, by linearity, we have the commutativity of the diagram. Then, f is chain-complex morphism.

- We prove in the same manner that g is a chain complex morphism. This time proving the commutativity of the diagram:

$$\begin{array}{ccc} \widehat{C}_{k-1} & \xleftarrow{N_k} & \widehat{C}_k \\ {\scriptstyle r^T_{k-1}}\uparrow & & \uparrow{\scriptstyle r^T_k} \\ EC_{k-1} & \xleftarrow[EN_k]{} & EC_k \end{array}$$

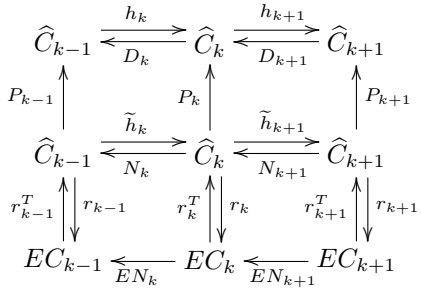
3. ***h* is a homotopy operator.**
   This is true by construction $h_* = P_* \widetilde{h_*} (P_{*-1})^{-1} : \widehat{C}_{*-1} \longrightarrow \widehat{C}_{*+1}$

$$\begin{array}{ccc} \widehat{C}_{k-1} & \longrightarrow & \widehat{C}_k \\ {\scriptstyle P_k}\uparrow & & \uparrow{\scriptstyle P_{k+1}} \\ EC_{k-1} & \xrightarrow{\widetilde{h}_{k+1}} & EC_k \end{array}$$

4. **These relationships are satisfied:**

   (a) $\boldsymbol{fg = id_{EC_*}}$ This is clearly true. An element of the base EC_* is an element of the reduced Smith base. By applying g we are expressing it in canonical coordinates, when applying f we transform it again in the Smith base and restrict it to EC_*. $fg = r_k(P_k^{-1})(P_k)r_k^T = r_k(r_k^T) = Id$

   (b) $\boldsymbol{gf + dh + hd = id_{\widehat{C}_*}}$ Observe this diagram:



   Observe that (*) $\widetilde{h}_k N_k + N_{k+1}\widetilde{h}_{k+1} + r_k^T r_k = Id_{\widehat{C}_k}$, because:

   i. $r_k^T r_k$ is an identity over the reduced base, $\boldsymbol{w}_k, \boldsymbol{c}_k, \boldsymbol{pw}_k$;
   ii. $\widetilde{h}_k N_k$ is the identity over $\boldsymbol{pb}^k$,
   iii. and $N_{k+1}\widetilde{h}_{k+1}$ is a identity over $\boldsymbol{b}^k$.

   Therefore, $\widetilde{h}_k N_k + N_{k+1}\widetilde{h}_{k+1} + r_k^T r_k = Id|_{\boldsymbol{pb}^k} + Id|_{\boldsymbol{b}^k} + Id|_{\boldsymbol{pw}_k, \boldsymbol{w}_k, \boldsymbol{c}_k} = Id_{\widehat{C}_k}$.

   Now,

   i. $h_k D_k = P_k \widetilde{h}_k (P_{k-1})^{-1} D_k \overset{(1)}{=} P_k \widetilde{h}_k N_k (P_k)^{-1}$
   ii. $D_{k+1} h_{k+1} = D_{k+1} P_{k+1} \widetilde{h}_{k+1} (P_k)^{-1} \overset{(2)}{=} P_k N_{k+1} \widetilde{h}_{k+1} (P_k)^{-1}$
   iii. $g_k f_k = P_k r_k^T r_k (P_k)^{-1}$

   $\xrightarrow{(i+ii+iii)}$

   $$h_k D_k + D_{k+1} h_{k+1} + g_k f_k = P_k(\widetilde{h}_k N_k + N_{k+1}\widetilde{h}_{k+1} + r_k^T r_k)(P_k)^{-1} \overset{(3)}{=} Id_{\widehat{C}_k}$$

   (1),(2) definition of N_k
   (3) by observation(*).

   (c) $\boldsymbol{fh = hg = hh = 0}$

i. $f_{k+1}h_k = f_{k+1}P_{k+1}\widetilde{h}_{k+1}(P_k)^{-1} = r_{k+1}(P_{k+1})^{-1}P_{k+1}\widetilde{h}_{k+1}(P_k)^{-1} = r_{k+1}\widetilde{h}_{k+1}(P_k)^{-1} \overset{(1)}{=} 0$.

(1) because $\widetilde{h}_{k+1}(P_k)^{-1}$ has its image in $[\boldsymbol{pb}^{k+1}]$

ii. $hg = 0$ (idem)

iii. $hh = 0$ because $h_{k+1}h_k = (P_{k+1}\,\widetilde{h}_{k+1}\,P_k^{-1})(P_k\,\widetilde{h}_k\,P_{k-1}^{-1}) = P_{k+1}\,\widetilde{h}_{k+1}\,\widetilde{h}_k P_{k-1}^{-1}$ and clearly $\widetilde{h}_{k+1}\,\widetilde{h}_k = 0$.

□

5.3.4 Interpretation of the elements in the reduction

Chain complexes and morphisms f and g

To define a Homological Smith Reduction, we need two chains, the big one $\widehat{C}_*$, the small one EC_*, and two morphisms between them f and g. The two chains have the same homology and the morphism f and g establish isomorphisms between the reduced chain and a subchain of the original one with equivalent homology.

Proposition 5.2.1 in the case of Homological Smith Reduction

Let us revisit the proposition 5.2.1 and interpret each element in the case of the Homological Smith Reduction:

Let $\rho : \widehat{C}_* \to C_*$ be a reduction (in this case, the homological smith reduction). This reduction is equivalent to a decomposition: $\widehat{C}_* = A_* \oplus B_* \oplus C'_*$. In the Homological Smith Reduction:

1. $\widehat{C}_* \supset C'_* = im\ g$ *is a subcomplex of* $\widehat{C}_*$
   $\widehat{C}_* \supset C'_* = \operatorname{im} g = [\boldsymbol{w}^*, \boldsymbol{c}^*, \boldsymbol{pw}^*]$
2. $A_* \oplus B_* = \ker f$ *is a sumbcomplex of* $\widehat{C}_*$.
   $A_* \oplus B_* = \ker f = [\boldsymbol{pb}^*, \boldsymbol{b}^*]$
3. $\widehat{C}_* \supset A_* = \ker f \cap \ker h$ *is not in general a subcomplex of* $\widehat{C}_*$.
   $A_* = \ker f \cap \ker h = [\boldsymbol{pb}^*]$
4. $\widehat{C}_* \supset B_* = \ker f \cap \ker d$ *is a subcomplex of* $\widehat{C}_*$ *with null differentials.*
   $B_* = \ker f \cap \ker d = [\boldsymbol{b}^*]$
5. *The chain-complex morphisms* f *and* g *are inverse isomorphisms between* C'_* *and* C_*.
   Effectively, remember that $f_k = P_k^{-1}|_{\boldsymbol{pw^k},\boldsymbol{w^k},\boldsymbol{c^k}}$, $g_k = P_k|_{\boldsymbol{pw^k},\boldsymbol{w^k},\boldsymbol{c^k}}$ (the first restriction

is on the rows and the second on the columns). Clearly their composition in any order gives the morphism identity in C'_* or C_*.

6. *The arrows d and h are module isomorphisms of respective degrees -1 and +1 between A_* and B_*.*
   This means that given an element σ of B_* we can find, by h a chain whose border is σ.

Why is the homotopy h needed?

This homotopy operator plays a very special role. Remember that the method based on the long exact sequence of the homology presents extensions problems that prevent, in general, to compute the homology. One of the reasons why the classical homology is not constructive is due to the definition of boundary:

$$c \in C_k \text{ is a boundary, if } \exists z \in C_{k+1} \text{ so that } d_{k+1}(z) = c.$$

This definition is not constructive; no method is given to find out the chain z. Thanks to the homotopy operator h, with the properties defined in the reduction, given a boundary c, we can find the chain z to which it is the image.

This is essential to compute the homology of the union. Let us consider the following example:

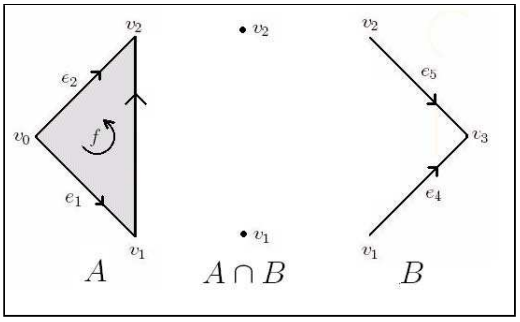


Figure 5.4: Decomposition of a simplicial complex X in two simplicial complexes, $X = A \cup B$ with $A \cup B \neq \emptyset$

It is clear from the picture that no one of the simplicial complexes, A, B, $A \cap B$, has an homological 1-cycle. However, the union of the elements has obviously one. When computing the homology of the union we need to find this 1-cycle.

If we perform the Homological Smith Reduction on A_*, B_*, $(A \cap B)_*$, we cannot obtain the 1-cycle directly from the reduced chains EA_*, EB_*, $E(A \cap B)_*$; observe on the figure

5.5 the reduced subcomplexes $\widetilde{A}$, $\widetilde{B}$ associated to the small chains EA_*, EB_* (all the details are in annex D). $\widetilde{A}$, $\widetilde{B}$ are retractions of A and B respectively:

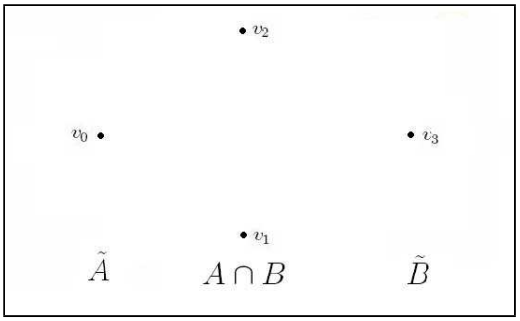


Figure 5.5: Simplicial complexes associated to the Smith small chains EA_*, EB_*, $E(A \cap B)_*$

On the figure 5.5, there is no 1-cycle, so, we need extra information to find the cycle of $A \cup B$ from the small chains EA_*, EB_*, $E(A \cap B)_*$. This information is in the original chain. We will find a chain in A_* that has as boundary $v_1 - v_2$ and a chain in B_* with the same boundary. The combination of these two chains will give us the searched cycle. But for this, we need a method to find the pre-image of the boundary operator. The homotopy h gives us this information.
Intuitively, the homotopy operator in the Homological Smith Reduction, captures all the information about the (pure-)boundaries and the pre-boundaries and ignores the rest. It is the constructive version of the definition of boundary (in the sense given in the definition 22; pure-boundary).

5.4 Composition of reductions

From [SR06, Prop. 59 page 56]

Proposition 5.4.1 (Composition of reductions)**.** *Let $\rho = (f, g, h) : C_* \Rrightarrow D_*$ and $\rho' = (f', g', h') : C'_* \Rrightarrow D'_*$ be two reductions. These reductions can be composed, producing the reduction $\rho'' = (f'', g'', h'') : C_* \Rrightarrow C''_*$ with:*

$$f'' = f'f$$

$$g'' = gg'$$

$$h'' = h + gh'f$$

Since in [SR06] the proof is left as an exercise, we give it here:

Proof. 1. C_* and C''_* are by hypothesis chain-complexes.

2. f'' and g'' are chain-complex morphisms because they are the composition of chain-complex morphisms.
3. h'' is obviously an homotopy operator.
4. The following relations hold:
   - $f''g'' = Id_{C''}$?
     $f''g'' = f'fgg' = f'Id_{C'}g' = f'g' = Id_{C''}$
   - $g''f'' + dh'' + h''d = Id_C$?
     $g''f'' + dh'' + h''d = g(Id_{C'} - dh' + h'd)f + dh + hd + dgh'f + gh'fd = gf - g(dh' + h'd)f + dh + hd + dgh'f + gh'fd = Id_C$
   - $f''h'' = h''g'' = h''h'' = 0$
     - $f''h'' = f'f(h + gh'f) = f'(fh) + f'(fg)h'f = 0 + f'Id_{C'}h'f = 0$
     - $h''g'' = (h + gh'f)gg' = (hg)g' + gh'(fg)g' = 0 + gh'(Id_{C'})g' = 0$
     - $h''h'' = (h + gh'f)(h + gh'f) = hh + (hg)h'f + gh'(fh) + gh'(fg)h'f = 0$

□

5.5 Equivalence of reductions

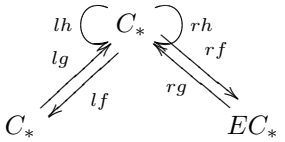
[SR06, Section 4.9, page 52]

Definition 29. Equivalence of reductions or strong homology equivalence.
A equivalence of reductions or strong homology equivalence $\epsilon : C_* \Longleftrightarrow D_*$ *between two chain-complexes* C_*, D_*, *is a pair of reductions connecting* C_* *and* D_* *through a third chain-complex* $\widehat{C}_*$*:*

$$\epsilon = C_* \overset{\rho_l}{\Leftarrow} \widehat{C}_* \overset{\rho_r}{\Rightarrow} D_*$$

Example 9. Reduction equivalence

We can consider for example, the next reduction equivalence:



where, for example, the left reduction is the trivial reduction and the right reduction is the Homological Smith Reduction.

We will see that the Cone Equivalence Theorem uses this structure.

Chapter 6

Cone construction

Remember that, given the homology of the simplicial complexes A, B and $A \cap B \neq \emptyset$, we want to compute the homology of the union $X = A \cup B$. At some point we need to 'mix' the homology information of A, B and $A \cap B$ in an 'appropriate' way in order to obtain the homology of the union. This appropriate way of 'mixing' the homology information is **the cone construction**; in this case, the homology information of A, B and $A \cap B$ is expressed as a reduction (for example, a reduction equivalence with the form presented in the example 9).

The cone is a concept vastly used because of its simplicity and the number of operations that can be performed with it. It appears in different contexts and with slightly different meaning but always based in the same idea.

In this chapter we will defined the cone concept and we will give its properties related with the reduction concept and the (effective) short sequence.

6.1 The concept of a cone

6.1.1 Topological cone

Depending on the context, the concept of cone is different, but always based in the same idea. Lets take, for example, the concept of a geometric cone, which is the cone we are used to deal with.

In topology there is an equivalent definition of cone. If X is a simplicial complex, we define the cone of X as:

$$CX := (X \times I)/(X \times \{0\})$$

where $I = [0, 1]$. This corresponds with creating a cylinder of base X and collapsing one of the two bases.

This construction has two main properties: is simple and has trivial homology (observe that

the cone is contractible to the vertex point by homotopy). This makes it very useful in algebraic topology, it embeds a space as a subspace of a contractible space.

6.1.2 Mapping cone and cone of a morphism

There is also a construction related with the topological cone known as a **mapping cone**, [Hat02]. Consider two topological spaces, for example, two simplicial complexes X and Y which are related by a map $f : X \to Y$. From these two spaces and f we construct a new topological space, the mapping cone, $C_f^{X,Y}$.

The process of construction is:

1. Create the cylinder of base X; $X \times [0, 1]$
2. Collapse the extreme $X \times \{0\}$ into a point to obtain the cone; CX.
3. Consider $X \times \{1\}$. We know that there is a map $f : X \times \{1\} \to Y$ that $(x, 1) \mapsto f(x)$. Consider the disjoint union of CX and Y and identify each point $(x, 1)$ of CX with $f(x) \in Y$. The result of this process is a new topological space, $C_f^{X,Y}$.

More concisely, we defined the mapping cone of $f : X \to Y$ as:

$$C_f^{X,Y} = Y \sqcup_f CX = Y \sqcup ((X \times [0, 1])/(X \times \{0\}))$$

where CX is the cone of X and $\sqcup_f$ represents attachment of Y along $X \times 1$ through the identifications $(x, 1) \approx f(x)$.

Suppose that we want to study the homology of the mapping cone of a morphism $f : X \to Y$. The steps to carry out are:

- consider the map $f : X \to Y$ relating the two simplicial complexes,
- construct $C_f^{X,Y}$, the cone of the map f,
- construct the chain complex of the cone, $(C_f^{X,Y})_*$,
- finally, compute its homology, $H_*(C_f^{X,Y})$.

Which is the relation between the chain-complexes X_*, Y_* and the chain-complex $(C_f^{X,Y})_*$
The cone theorem answers this question and, also, the following one:
Suppose that two chain complexes, X_* and Y_*, are given and a chain-complex morphism $f : X_* \to Y_*$, can we construct the chain complex $(C_f^{X,Y})_*$ directly using only this information (i.e, without referring to the underlying simplicial complexes)?
i.e:

- We have the chain complexes X_*, Y_*, associated to the two simplicial complexes, X, Y.

- We have a chain complex morphism $f : X_* \to Y_*$ induced by the simplicial map $f : X \to Y$.
- We want to construct $(C_f^{X,Y})_*$ with these elements and compute its homology.

In this case, when the chain complex $(C_f^{X,Y})_*$ is constructed from two chain-complexes X_* and Y_* and a chain complex morphism $f : X_* \to Y_*$, $(C_f^{X,Y})_*$ is called **cone morphism of** f and is denoted $Cone(f)$.

6.2 Cone of a morphism

Definition 30. Cone of a morphism.
([SR06]) Suppose that we have two chain complexes X_*, Y_* and a chain complex morphism $f : X_* \to Y_*$ between them. From these complexes and morphism f we define a new chain complex, denoted $Cone(f) = C_*$.
At each dimension $C_k := Y_k \oplus X_{k-1}$ and its boundary operator is:

$$D_{C_*} := \begin{bmatrix} D_{Y_*} & f_{*-1} \\ 0 & -D_{X_{*-1}} \end{bmatrix}$$

- *What does it mean $Y_k \oplus X_{k-1}$?*

  The elements of $C_k := Y_k \oplus X_{k-1}$ are elements of Y_k or elements of X_{k-1} or **formal** sums of these elements. Y_k and X_{k-1} are considered as 'disjoint' modules, i.e, if, for example, we must give a base of C_k, it will be formed by a base of Y_k and a base of X_{k-1}. Imagine that there is a common element to these two bases; should we eliminate one of them from the base of C_k? The answer is no. Even if Y_k and X_{k-1} have elements in common, in C_k they are considered disjointly.

- *In C_k, why do we consider the module X_{k-1} of inferior dimension than Y_k?*
  The answer is in the cone construction. While performing the cone of the morphism we are implicitly performing the topological cone of the underlying space X. Consider the basis $X \times \{1\}$ in the cone CX; this is a copy of X. Consider a face σ of $X \times \{1\}$ along with all the lines that unite that face with the vertex point of the cone, call it σ_C. σ_C is of one dimension higher than σ and is a face of CX. That is why we consider X_{k-1} instead of X_k, because implicitly we are considering the cone of X and in the cone, the faces of X_{k-1} correspond to faces of dimension k in CX.

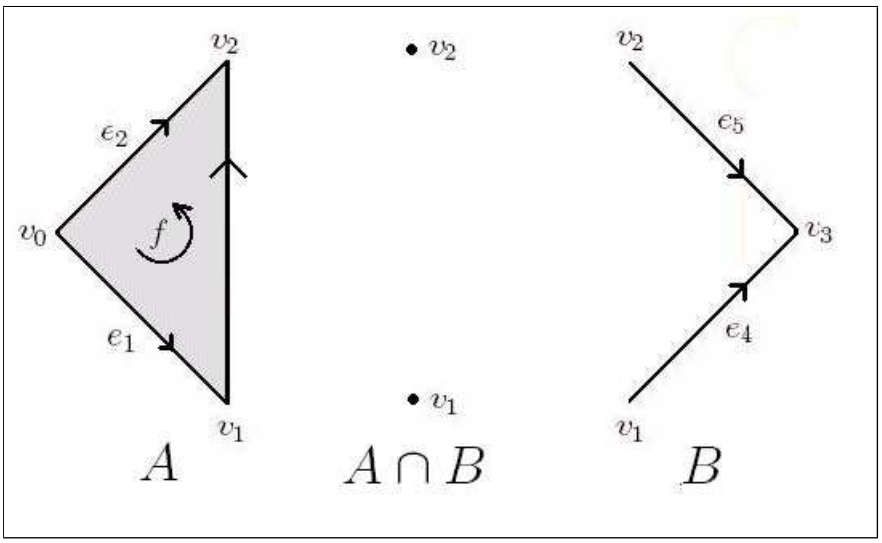
Example 10. Cone of the morphism inclusion in Mayer-Vietoris
We have two simplicial complexes A and B and their intersection $A \cap B \neq \emptyset$. We consider the application $i : A \cap B \to A \sqcup B$ (the second element is the disjoint union of A and B) that sends

$\sigma \mapsto (\sigma, \sigma)$. This application induces a chain-complex morphism $i_A \oplus i_B : (A \cap B)_* \to A_* \oplus B_*$. Then, the $Cone(i) = C_*$ is $C_k = (A_k \oplus B_k) \oplus (A \cap B)_{k-1}$. A basis of C_k is $\boldsymbol{\sigma}_k^A \oplus \boldsymbol{\sigma}_k^B \oplus \boldsymbol{\sigma}_{k-1}^{A \cap B}$ i.e the 'disjoint' union of the basis of A, B and $A \cap B$. The border operator matrix expressed in this base is:

$$D_k = \begin{pmatrix} D_{A_k} & 0 & i_{A_{k-1}} \\ 0 & D_{B_k} & i_{B_{k-1}} \\ 0 & 0 & -D_{(A \cap B)_{k-1}} \end{pmatrix}$$

Observe that i_A and i_B are the matrices associated to the morphisms i_A, i_B (abuse of notation).

In the particular case presented in the following figure, we have:



$$A_* : 0 \longleftarrow A_0 = [v_0, v_1, v_2] \longleftarrow A_1 = [e_1, e_2, e_3] \longleftarrow A_2 = [f] \longleftarrow 0$$

$$B_* : 0 \longleftarrow B_0 = [v_1, v_2, v_3] \longleftarrow B_1 = [e_4, e_5] \longleftarrow 0$$

$$(A \cap B)_* : 0 \longleftarrow (A \cap B)_0 = [v_1, v_2] \longleftarrow 0$$

In the annex B we study this example; the incidence matrices are:

$$D_1^A = \begin{array}{c} \\ v_0 \\ v_1 \\ v_2 \end{array} \begin{array}{c} \begin{array}{ccc} e_1 & e_2 & e_3 \end{array} \\ \begin{pmatrix} -1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & 1 \end{pmatrix} \end{array} \qquad D_2^A = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \end{array} \begin{array}{c} \begin{array}{c} f \end{array} \\ \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix} \end{array}$$

$$D_1^B = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{cc} e_4 & e_5 \end{array} \\ \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \end{pmatrix} \end{array}$$

The chain complex of the $Cone(i_A \oplus i_B)$ is

$$0 \longleftarrow A_0 \oplus B_0 \longleftarrow A_1 \oplus B_1 \oplus (A \cap B)_0 \longleftarrow A_2 \longleftarrow 0$$

, i.e

$$Cone(i_A \oplus i_B) : 0 \leftarrow C_0 = [v_0, v_1, v_2] \oplus [v_1, v_2, v_3] \leftarrow C_1 = [e_1, e_2, e_3] \oplus [e_4, e_5] \oplus [v_1, v_2] \leftarrow C_2 = [f] \leftarrow 0$$

and the incidence matrices are:

$$D_1^C = \begin{array}{c} \\ v_0 \\ v_1 \\ v_2 \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{ccccccc} e_1 & e_2 & e_3 & e_4 & e_5 & v_1 & v_2 \end{array} \\ \begin{pmatrix} -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

$$D_2^C = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ v_1 \\ v_2 \end{array} \begin{array}{c} \begin{array}{c} f \end{array} \\ \begin{pmatrix} 1 \\ -1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \end{array}$$

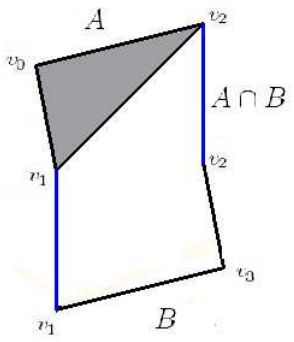


Figure 6.1: Cone of the example in the figure 10

6.3 Cone and reductions

Imagine that we have two chain complexes C_* and C'_* with respective reductions: $\rho' = (f', g', h') : C'_* \Rrightarrow D'_*$, $\rho = (f, g, h) : C_* \Rrightarrow D_*$. Suppose too, that we have a chain-morphism $\phi : C'_* \Rrightarrow C_*$, relating these two chains.
We can construct $Cone(\phi)$. Can we obtain a reduction of this cone from the reduction of C_*, C'_*? The Cone Reduction Theorem gives a positive answer ([SR06, Section 5.5]).

Theorem 6.3.1 (Cone Reduction Theorem)**.** *Let $\rho = (f, g, h) : C_* \Rrightarrow D_*$ and $\rho' = (f', g', h') : C'_* \Rrightarrow D'_*$ be two reductions and $\phi : C'_* \Rrightarrow C_*$ a chain-complex morphism. Then there is a canonical reduction:*

$$\rho'' = (f'', g'', h'') : Cone(\phi) \Rrightarrow Cone(f\phi g')$$

$$\text{with } f'' = \begin{bmatrix} f & -f\phi h' \\ 0 & f' \end{bmatrix}$$
$$g'' = \begin{bmatrix} g & -h\phi g' \\ 0 & g' \end{bmatrix}$$
$$h'' = \begin{bmatrix} h & h\phi h' \\ 0 & -h' \end{bmatrix}$$

Example 11. Cone reduction and Mayer-Vietoris

We work again in the context of Mayer-Vietoris, i.e, with a finite simplicial complex $X = A \cup B$ which is union of two subcomplexes, A and B with non-empty intersection $A \cap B$. Consider the chain-morphism $i_A \oplus i_B : (A \cap B)_* \longrightarrow A_* \oplus B_*$. Consider the Homological Smith Reductions of $(A \cap B)_*$ and A_*, B_*.

$$\rho^A = (f_A, g_A, h_A) : A_* \Rrightarrow EA_*$$

,

$$\rho^B = (f_B, g_B, h_B) : B_* \Rrightarrow EB_*$$

,

$$\rho^{A\cap B} : (f_{A\cap B}, g_{A\cap B}, h_{A\cap B}) = (A\cap B)_* \Rrightarrow E(A\cap B)_*$$

Observe that for $A_* \oplus B_*$, we have the reduction: $\rho^{A\oplus B} = (f_A \oplus f_B, g_A \oplus g_B, h_A \oplus h_B) : A_* \oplus B_* \Rrightarrow EA_* \oplus EB_*$.
We have the next diagram:

$$\begin{array}{ccc} (A\cap B)_* & \xrightarrow{i_A\oplus i_B} & A_* \oplus B_* \\ {\scriptstyle g_{A\cap B}}\uparrow\downarrow{\scriptstyle f_{A\cap B}} & & {\scriptstyle g_{A\oplus B}}\uparrow\downarrow{\scriptstyle f_{A\oplus B}} \\ E(A\cap B)_* & \longrightarrow & EA_* \oplus EB_* \end{array}$$

The Cone Reduction Theorem tell us that there is a canonical reduction:

$$\rho^C = (f_C, g_C, h_C) : Cone(i_A \oplus i_B) \Rrightarrow Cone((f_A \oplus f_B)(i_A \oplus i_B)g_{A\cap B})$$

Define:

$$EC_k := Cone((f_A \oplus f_B)(i_A \oplus i_B)g_{A\cap B})_k = EA_k \oplus EB_k \oplus E(A\cap B)_{k-1}$$

with border operator:

$$d_{EC_k} = \begin{bmatrix} d_{EA_k} \oplus d_{EB_k} & (f_A \oplus f_B)(i_A \oplus i_B)g_{(A\cap B)} \\ 0 & -d_{E(A\cap B)_{k-1}} \end{bmatrix}$$

In matrix form:

$$D_{EC_k} = \begin{pmatrix} EN_k^A & 0 & \phi_{k-1}^A \\ 0 & EN_k^B & \phi_{k-1}^B \\ 0 & 0 & -EN_{k-1}^{A\cap B} \end{pmatrix}$$

where

$$\phi_k^A = f_A\ i_A\ g_{A\cap B} = (r_k^A)(P_k^A)^{-1} i_k^A (P_k^{A\cap B})(r_k^{A\cap B})^T$$

ϕ_k^B is defined analogously. i^A is the matrix representing the inclusion of $(A\cap B)_*$ in A_*.

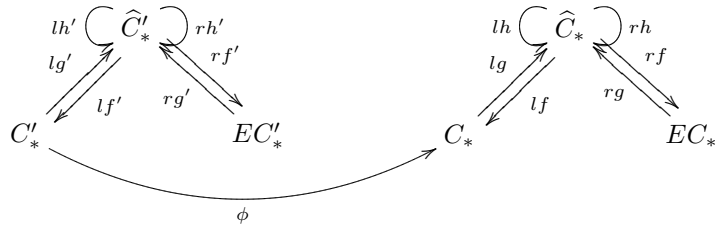
The morphism g that gives the homology of $Cone(i_A\oplus i_B)$ from the homology of $Cone((f_A\oplus f_B)(i_A \oplus i_B)g_{A\cap B})$ is:

$$g = \begin{pmatrix} (g_A \oplus g_B) & -(h_A \oplus h_B)(i_A \oplus i_B)g_{A\cap B} \\ 0 & g_{A\cap B} \end{pmatrix} = \begin{pmatrix} g_A & 0 & -h_A\ i_A\ g_{A\cap B} \\ 0 & g_B & -h_B\ i_B\ g_{A\cap B} \\ 0 & 0 & g_{A\cap B} \end{pmatrix}$$

6.4 Cone equivalences

[SR06, Section 6.2, page 70]

Theorem 6.4.1 (Cone Equivalence Theorem)**.** *Let $\phi : C'_* \longrightarrow C_*$ be a chain-complex morphism between two chain-complexes*[1]*. Then a general algorithm computes a version of the $Cone(\phi)$ (and the modules of the chain are of finite type).*

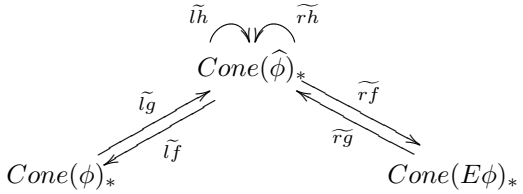


and we have the equivalence:

$$Cone(C'_* \xrightarrow{\phi} C_*) \overset{\rho_l}{\Leftarrow} Cone(\widehat{C}'_* \xrightarrow{\widehat{\phi}} \widehat{C}_*) \overset{\rho_r}{\Rightarrow} Cone(EC'_* \xrightarrow{E\phi} EC_*)$$

Remark 6.4.2. *Observe that the reduction equivalence obtained from the Cone equivalence is the result of performing twice the Cone Reduction Theorem; once over the left reductions and another time over the right reductions.*

The result of the cone equivalence is a new reduction equivalence:



where $\widehat{\phi} = (lg)\ \phi\ (lf')$ and $E\phi = (rf)\ (lg)\ \phi\ (lf')\ (rg')$
and

$$\widetilde{rg} = \begin{bmatrix} rg & -(rh)\widehat{\phi}(rg') \\ 0 & rg' \end{bmatrix} \quad \widetilde{rf} = \begin{bmatrix} rf & -(rf)\widehat{\phi}(rh') \\ 0 & rf' \end{bmatrix} \quad \widetilde{rh} = \begin{bmatrix} rh & (rh)\widehat{\phi}(rh') \\ 0 & -(rh') \end{bmatrix}$$

Analogously:

$$\widetilde{lg} = \begin{bmatrix} lg & -(lh)\widehat{\phi}(lg') \\ 0 & lg' \end{bmatrix} \quad \widetilde{lf} = \begin{bmatrix} lf & -(lf)\widehat{\phi}(lh') \\ 0 & lf' \end{bmatrix} \quad \widetilde{lh} = \begin{bmatrix} lh & (lh)\widehat{\phi}(lh') \\ 0 & -(lh') \end{bmatrix}$$

[1] We are always supposing that the modules of the chain complexes are finite

Chapter 7

Effective short exact sequence

Remember that the classical Mayer-Vietoris method is the result of a short exact sequence that for each dimension k is of the form (see example 3 at page 26):

$$0 \longleftarrow (A \cup B)_k \xleftarrow{j_A \ominus j_B} A_k \oplus B_k \xleftarrow{i_A \oplus i_B} (A \cap B)_k \longleftarrow 0$$

From this set of short exact sequences, the classical Mayer-Vietoris exact long sequence is constructed (page 29). However, this method for computing the homology is not constructive.

In constructive homology it is defined the **effective short exact sequence** which is an exact short sequence with two graded module morphisms:

Definition 31. Effective short exact sequence.
An effective short exact sequence of chain-complexes is a diagram:

$$0 \longleftarrow R_* \underset{j}{\overset{\nu}{\rightleftarrows}} S_* \underset{i}{\overset{\rho}{\rightleftarrows}} T_* \longleftarrow 0$$

where i and j are chain-complex morphisms, ρ (retraction) and ν (section) are graded module morphisms satisfying:

- $\rho i = Id_{T_*}$
- $i\rho + \nu j = id_{S_*}$
- $j\nu = id_{R_*}$

Example 12. Mayer-Vietoris is a effective short exact sequence
The **effective short exact sequence of Mayer-Vietoris** is defined as:

$$0 \longleftarrow (A \cup B)_* \underset{j}{\overset{\nu}{\rightleftarrows}} A_* \oplus B_* \underset{i}{\overset{\rho}{\rightleftarrows}} (A \cap B)_* \longleftarrow 0$$

with:

$$\begin{array}{rccc} i = i_A \oplus i_B : & (A \cap B)_* & \longrightarrow & A_* \oplus B_* \\ & \sigma & \longmapsto & (\sigma, \sigma) \end{array}$$

$$\begin{array}{rccc} j = j_A \ominus j_B : & A_* \oplus B_* & \longrightarrow & (A \cup B)_* \\ & (\sigma, \tilde{\sigma}) & \longmapsto & \sigma - \tilde{\sigma} \end{array}$$

$$\begin{array}{rccc} \nu : & (A \cup B)_* & \longrightarrow & A_* \oplus B_* \\ & \sigma & \longmapsto & (\sigma|_A, -\sigma|_B + \sigma|_{A\cap B}) \end{array}$$

$$\begin{array}{rccc} \rho : & A_* \oplus B_* & \longrightarrow & (A \cap B)_* \\ & (\sigma, \tilde{\sigma}) & \longmapsto & \tilde{\sigma}|_{A\cap B} \end{array}$$

Proof. Lets prove that the properties in the definition are satisfied:

- $\rho i = id_{(A\cap B)_*}$
  $\rho(i_A \oplus i_B)(\sigma) = \rho(\sigma, \sigma) = \sigma|_{A\cap B} = \sigma = id_{A\cap B}(\sigma)$
- $i\rho + \nu j = id_{(A\oplus B)_*}$.
  Suppose $\sigma \in A_*$ and $\tilde{\sigma} \in B_*$:
  1. $(i_A \oplus i_B)\rho(\sigma, \tilde{\sigma}) = (\tilde{\sigma}|_{A\cap B}, \tilde{\sigma}|_{A\cap B})$
  2. $\nu(j_A \ominus j_B)](\sigma, \tilde{\sigma}) = \nu(\sigma - \tilde{\sigma}) = ((\sigma - \tilde{\sigma})|_A, -(\sigma - \tilde{\sigma})|_B + (\sigma - \tilde{\sigma})|_{A\cap B}) = (\sigma - \tilde{\sigma}|_A, -\sigma|_B + \tilde{\sigma} + \sigma|_B - \tilde{\sigma}|_{A\cap B}) = (\sigma - \tilde{\sigma}|_A, \tilde{\sigma} - \tilde{\sigma}|_{A\cap B}) = (\sigma - \tilde{\sigma}|_{A\cap B}, \tilde{\sigma} - \tilde{\sigma}|_{A\cap B})$

  $\Longrightarrow [(i_A \oplus i_B)\rho + \nu(j_A \ominus j_B)](\sigma, \tilde{\sigma}) = (\sigma, \tilde{\sigma})$
- $j\nu = id_{(A\cup B)_*}$
  $[(j_A \ominus j_B)\nu](\sigma) = (j_A \ominus j_B)(\sigma|_A, -\sigma_B + \sigma|_{A\cap B}) = \sigma|_A + \sigma_B - \sigma|_{A\cap B} = \sigma = id_{A\cup B}(\sigma)$

□

7.1 Cone, effective short exact sequences and Lemma 82

The next lemma is the constructive version of the long exact sequence, it is the **Lemma 82** in the reference [SR06, Section 6.2] which, in the same reference, is a result of Theorem 81 (SES theorems).

Lemma 7.1.1 (Lemma 82)**.** *The effective exact sequence*

$$0 \longleftarrow R_* \underset{j}{\overset{\nu}{\rightleftarrows}} S_* \underset{i}{\overset{\rho}{\rightleftarrows}} T_* \longleftarrow 0$$

produces a reduction: $\rho = (f, g, h) : Cone(i) \Rrightarrow R_*$. Furthermore, the border operator of R_* given by this reduction is by chance the original border operator d_{R_*} of R_*.

- $f = j$
- $h = \rho$
- $g = \nu - \rho\ d_{S_*}\ \nu$

We will call this lemma, **Lemma 82**

Example 13. Mayer-Vietoris and Lemma 82

In the Mayer-Vietoris case, this lemma tells us how to construct a reduction:

$$\rho = (f, g, h) : Cone(i_A \oplus i_B) \Rrightarrow (A \cup B)_*$$

with $f = j_A \ominus j_B$.

This means that if we know the homology of $Cone(i_A \oplus i_B)$, then we can evaluate the homology of $(A \cup B)_*$. What is needed is the computation of the image of each element of the homology of $Cone(i_A \oplus i_B)$ by the morphism $f = j_A \ominus j_B$.

$$0 \longleftarrow (A \cup B)_* \underset{j_A \ominus j_B}{\overset{\nu}{\rightleftarrows}} A_* \oplus B_* \underset{i_A \oplus i_B}{\overset{\rho}{\rightleftarrows}} (A \cap B)_* \longleftarrow 0$$

Conclusion of part II

In this part, we have seen a constructive version of the Mayer-Vietoris long exact sequence using the Lemma 82 (example 13) and a method of reducing the incidence matrices without modifying the homology (Homological Smith Reduction).

These are the tools used in the Constructive Mayer-Vietoris algorithm. See in the schema of the figure 7.1 the relations between Classical Homology and Constructive Homology.

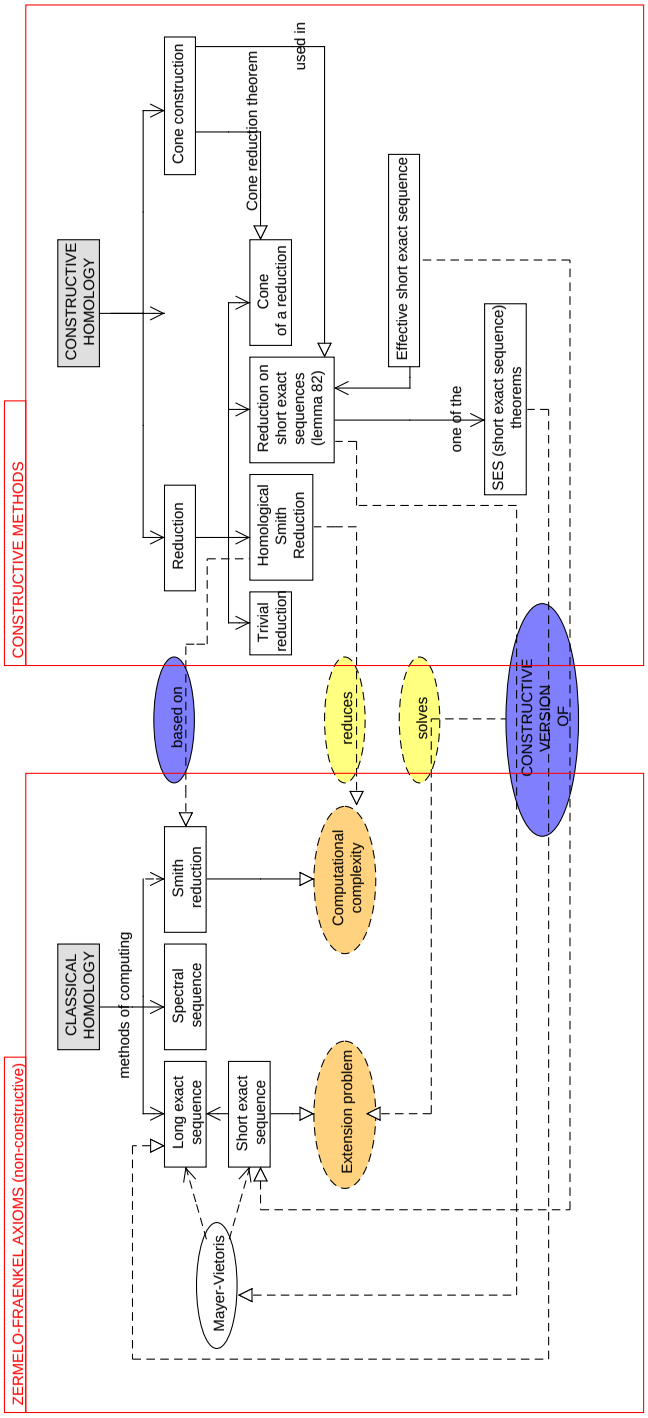


Figure 7.1: Diagram relating the concepts of Classical Homology, part I, and Constructive Homology, part II

Part III

Constructive Mayer-Vietoris Algorithm

Chapter 8

Theoretical algorithm

Now we are prepared to explain the Constructive Mayer-Vietoris Algorithm. Firstly, we explain the one step algorithm, i.e, an algorithm for computing the union homology of two simplicial complexes. The one step algorithm is not an iterative algorithm. Afterwards, we explain the iterative algorithm.

We make the distinction between this two methods because the optimizations that can be applied to each one are different.

First, let us define:

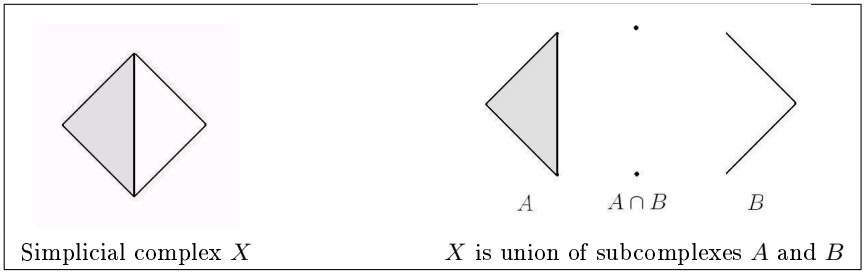
Definition 32. Homological information of a simplicial complex.

We call homological information of a simplicial complex X a reduction equivalence where one of the reductions is of type Homological Smith Reduction, and the small chain of the other reduction is the chain complex X_*

Proposition 8.0.2 (Homology from the homological information of a simplicial complex)**.** *The homological information of a simplicial complex allows to compute its homology generators and torsion coefficients.*

Proof. Immediate, because the reductions establish isomorphisms between the homology groups and the Homological Smith Reduction provides the homology of one of the chains. (See the step 6 of the one step algorithm in the next section). □

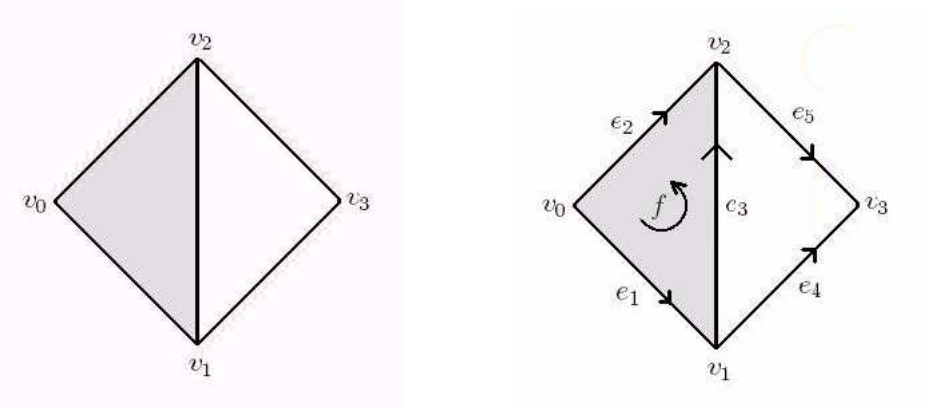
8.1 One step algorithm



Simplicial complex X — X is union of subcomplexes A and B

Input A simplicial complex X decomposed in two subcomplexes A and B, so that $X = A \cup B$ and $A \cap B \neq \emptyset$.

Step 1 Order the vertices of X and enumerate the cells of X at each dimension.



(a) Order the vertices of X (b) Enumerate the faces of X

Figure 8.1: Step 1

Step 2 Compute the Homological Smith Reduction for A, B and $A \cap B$.

$$\rho^A = (f_A, g_A, h_A) : A_* \Rrightarrow EA_*$$

$$\rho^B = (f_B, g_B, h_B) : B_* \Rrightarrow EB_*$$

$$\rho^{A \cap B} : (f_{A \cap B}, g_{A \cap B}, h_{A \cap B}) = (A \cap B)_* \Rrightarrow E(A \cap B)_*$$

The steps are:

1. For A, B and $A \cap B$ compute the list of incidence matrices (associated to the enumeration given to the cells; canonical basis $\{\boldsymbol{\sigma}^k\}_k$).

$$D_k = \begin{array}{c} \\ \sigma^{k-1} \\ \vdots \\ \sigma^{k-1}_{l(k-1)} \end{array} \begin{array}{c} \begin{array}{ccc} \sigma^k_1 & \dots & \sigma^k_{l(k)} \end{array} \\ \begin{pmatrix} & & \\ & \dots & \\ & & \end{pmatrix} \end{array}$$

2. Perform the Smith Reduction as explained in annex B. (Obtain the matrices of change of base P_k, P_k^{-1}, and the incidence matrices in Smith Normal Form N_k for each k).

$$N_k = \begin{pmatrix} 0 & \boldsymbol{\lambda} & 0 \\ 0 & 0 & Id \\ 0 & 0 & 0 \end{pmatrix} \qquad P_k = \begin{array}{c} \\ \sigma^k_1 \\ \vdots \\ \sigma^k_{l(k)} \end{array} \begin{array}{c} \begin{array}{ccccc} \boldsymbol{w}^k & \boldsymbol{b}^k & \boldsymbol{c}^k & \boldsymbol{pw}^k & \boldsymbol{pb}^k \end{array} \\ \begin{pmatrix} & & & & \\ & & \dots & & \\ & & & & \end{pmatrix} \end{array} \qquad P_k^{-1} = \begin{array}{c} \\ \boldsymbol{w}^k \\ \boldsymbol{b}^k \\ \boldsymbol{c}^k \\ \boldsymbol{pw}^k \\ \boldsymbol{pb}^k \end{array} \begin{array}{c} \begin{array}{ccc} \sigma^k_1 & \dots & \sigma^k_{l(k)} \end{array} \\ \begin{pmatrix} & & \\ & \dots & \\ & & \end{pmatrix} \end{array}$$

3. Compute the Homological Smith Reduction using the information of the previous step, (see section 5.3).

Output: For $\bullet = A,\ B,\ A \cap B$ the matrices that represent the morphisms in the Homological Smith Reduction:

$$\begin{array}{ccc} \overset{h_{A\cap B}}{\circlearrowleft} & \overset{h_A}{\circlearrowleft} & \overset{h_B}{\circlearrowleft} \\ (A\cap B)_* & A_* & B_* \\ {\scriptstyle g_{A\cap B}}\uparrow\downarrow{\scriptstyle f_{A\cap B}} & {\scriptstyle g_A}\uparrow\downarrow{\scriptstyle f_A} & {\scriptstyle g_B}\uparrow\downarrow{\scriptstyle f_B} \\ E(A\cap B)_* & EA_* & EB_* \end{array}$$

$$f_\bullet = \{(P_k)^{-1}|_{\boldsymbol{pw}^k, \boldsymbol{w}^k, \boldsymbol{c}^k}\}_k$$

$$g_\bullet = \{P_k|_{\boldsymbol{pw}^k, \boldsymbol{w}^k, \boldsymbol{c}^k}\}_k$$

$$h_\bullet = \left\{ \begin{array}{c} \\ \sigma^k_1 \\ \vdots \\ \sigma^k_{l(k)} \end{array} \begin{array}{c} \begin{array}{c} \boldsymbol{pb}^k \end{array} \\ \begin{pmatrix} \\ P_k|_{\boldsymbol{pb}} \\ \\ \end{pmatrix} \end{array} * \boldsymbol{b}^{k-1} \begin{array}{c} \begin{array}{ccc} \sigma^{k-1}_1 & \dots & \sigma^{k-1}_{l(k-1)} \end{array} \\ \begin{pmatrix} & P^{-1}_{k-1}|_{\boldsymbol{b}^{k-1}} & \end{pmatrix} \end{array} \right\}_k$$

and the list of incidence matrices of the small chains:

$$EN_{k,\bullet} := \begin{array}{c} \\ \boldsymbol{w}^{k-1} \\ \boldsymbol{c}^{k-1} \\ \boldsymbol{pw}^{k-1} \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{w}^k & \boldsymbol{c}^k & \boldsymbol{pw}^k \end{array} \\ \begin{pmatrix} 0 & 0 & \boldsymbol{\lambda} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

Step 3 Consider the inclusion morphism $i : (A \cap B)_* \longrightarrow A_* \oplus B_*$ and construct the Cone Reduction (Cone Reduction Theorem, theorem 6.3.1)

$$\rho^C = (f_C, g_C, h_C) : Cone(i_A \oplus i_B) \Rrightarrow Cone((f_A \oplus f_B)(i_A \oplus i_B)g_{A\cap B})$$

(see example 11). The steps are:

1. Sum[1] of the reductions of A and B:

$$\rho^{A\oplus B} = (f_A \oplus f_B, g_A \oplus g_B, h_A \oplus h_B) : A_* \oplus B_* \Rrightarrow EA_* \oplus EB_*$$

$$\begin{array}{c} \overset{h_{A\oplus B}}{\curvearrowleft} \\ A_* \oplus B_* \\ {\scriptstyle g_{A\oplus B}} \Big\Uparrow \Big\Downarrow {\scriptstyle f_{A\oplus B}} \\ EA_* \oplus EB_* \end{array}$$

2. Consider the morphism inclusion between the reductions $\rho^{A\cap B}$ and $\rho^{A\oplus B}$.

$$\begin{array}{ccc} \overset{h_{A\cap B}}{\curvearrowleft} & & \overset{h_{A\oplus B}}{\curvearrowleft} \\ (A \cap B)_* & \xrightarrow{i_A \oplus i_B} & A_* \oplus B_* \\ {\scriptstyle g_{A\cap B}} \Big\Uparrow \Big\Downarrow {\scriptstyle f_{A\cap B}} & & {\scriptstyle g_{A\oplus B}} \Big\Uparrow \Big\Downarrow {\scriptstyle f_{A\oplus B}} \\ E(A \cap B)_* & & EA_* \oplus EB_* \end{array}$$

3. Compute the cone morphism of $i = i_A \oplus i_B$, $Cone(i)$ and the canonical reduction given by the Cone Reduction Theorem, (see the formulas for computing the morphisms f_C, g_C, h_C of the reduction at page 66, theorem 6.3.1). The small chain is $Cone(Ei)_* := Cone((f_A \oplus f_B)(i_A \oplus i_B)g_{A\cap B})_* = EA_* \oplus EB_* \oplus E(A \cap B)_{*-1}$

$$\rho^C = \begin{array}{c} \overset{h_C}{\curvearrowleft} \\ Cone(i)_* \\ {\scriptstyle g_C} \Big\Uparrow \Big\Downarrow {\scriptstyle f_C} \\ Cone(Ei)_* \end{array}$$

[1] $f_A \oplus f_B = \begin{bmatrix} f_A & 0 \\ 0 & f_B \end{bmatrix}$

Step 4 Compute the homological information of $(A \cup B)_*$. The steps are

1. Compute the Homological Smith Reduction of $Cone(Ei)$

$$\rho^{EC} = \quad \overset{h'}{\curvearrowleft} Cone(Ei)_* \quad \underset{g'}{\overset{f'}{\Updownarrow}} \quad E(Cone(Ei))_*$$

2. Compose the reductions ρ^C and ρ^{EC} (section 5.4):

$$\overset{h_C}{\curvearrowleft} Cone(i)_* \quad \underset{g_C}{\overset{f_C}{\Updownarrow}} \quad \overset{h'}{\curvearrowleft} Cone(Ei)_* \quad \underset{g'}{\overset{f'}{\Updownarrow}} \quad E(Cone(Ei))_* \qquad \Longrightarrow \qquad \rho^r = \quad \overset{rh}{\curvearrowleft} Cone(i)_* \quad \underset{rg}{\overset{rf}{\Updownarrow}} \quad E(Cone(Ei))_*$$

3. Compute the reduction associated to the effective short exact sequence over $Cone(i)$ (Lemma 82, page 70, section 7.1).

$$\rho^l = \quad \overset{lh}{\curvearrowleft} Cone(i)_* \quad \underset{lg}{\overset{lf}{\Updownarrow}} \quad (A \cup B)_*$$

Output: Homological information of $(A \cup B)_*$; i.e, the reduction equivalence

$$\epsilon : \rho^l \Longleftrightarrow \rho^r$$

$$\overset{lh}{\curvearrowleft}\overset{rh}{\curvearrowright} Cone(i)_* \quad \underset{lf}{\overset{lg}{\rightleftarrows}} \quad (A \cup B)_* \qquad\qquad Cone(i)_* \quad \underset{rg}{\overset{rf}{\rightleftarrows}} \quad E(Cone(Ei))_*$$

Step 6 Compute the generators of $H_k((A \cup B)_*)$. For each cycle $c \in \boldsymbol{c}^k$ and each weak-boundary $w \in \boldsymbol{w}^k$, compute its image by $(lf)\ (rg)$. $(lf)\ (rg)(c)$ is a homological cycle of the union and $(lf)\ (rg)(w)$ is a weak-boundary of the union, its torsion coefficient is the correspondent torsion coefficient of w in $E(Cone(Ei))$.

8.1.1 Computations simplification

In practice, some of the information computed in the one step algorithm is not used. We can, therefore, simplify the algorithm. We do not need to compute:

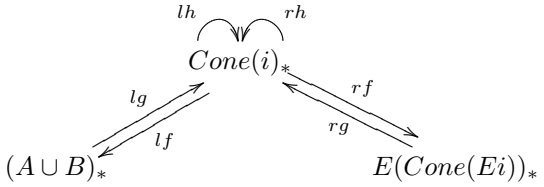
1. $h_{A\cap B}$
2. The morphisms f_C, h_C in the reduction of the $\rho^C : Cone(i) \Rrightarrow Cone(Ei)$.
3. The morphisms h', f' in the reduction $\rho^{EC} : Cone(Ei) \Rrightarrow E(Cone(Ei))$.
4. The morphisms rf, rh in the composition of reductions $\rho^C\ \rho^{EC} : Cone(i) \Rrightarrow Cone(Ei) \Rrightarrow E(Cone(Ei))$.
5. The morphism lg, lh in the reduction $\rho^l : Cone(i) \Rrightarrow (A \cup B)_*$

To see a handwritten example of the simplified one step algorithm see annex D.

8.2 Iterative algorithm

Now we describe the iterative algorithm. We need to add some changes to the one step algorithm.

The output in the one step algorithm is a reduction equivalence which is the homological information of the union $A \cup B$ of simplicial complexes A, B:



A reduction equivalence is not the input of the one step algorithm, therefore, the iteration is not possible. To achieve an iterative algorithm, we need the Cone Equivalence Theorem (theorem 6.4.1)

Input: A simplicial complex X and a decomposition in subcomplexes $A_1, \ldots, A_p$, so that, $X = A_1 \cup \ldots \cup A_p$ and for all r there is an s so that $A_r \cap A_s \neq \emptyset$.

Step 1 Order the vertices of X and enumerate the cells of X at each dimension (the subcomplexes $(A_r)_r$ inherit the vertex ordering and the cells enumeration).

Step 2 Compute the **homology information** of each subcomplex A_r, i.e, for each r, compute a reduction equivalence

$$\epsilon^r : (A_r)_* \overset{trivial}{\Leftarrow} (A_r)_* \overset{Smith}{\Rightarrow} E(A_r)_*$$

where the right reduction is the Homological Smith Reduction of A_r and the left reduction is a trivial reduction of A_r(see section 7).

Step 3 Create a list of *non_treated_simplices* and initialize it with the list $\{A_r\}_r$

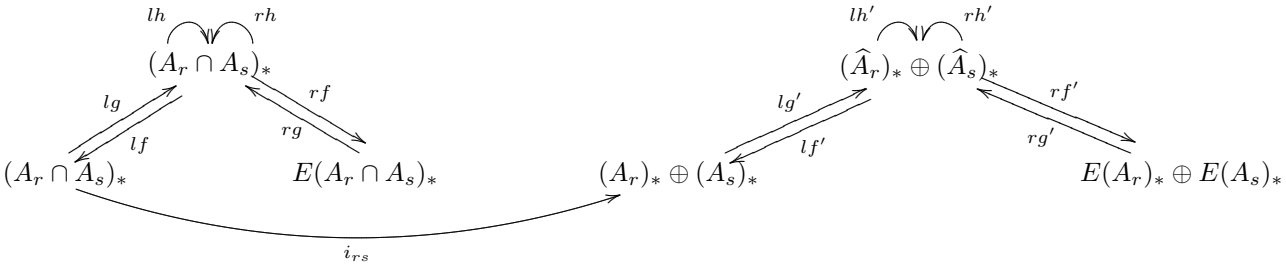
Loop

- If *non_treated_simplices* contains a single element X, end of the loop.
- Let A_r, A_s be two simplicial complexes in the *non_treated_simplices* list so that $A_r \cap A_s \neq \emptyset$. Do:

1. Compute the intersection of A_r, A_s.
2. Compute the homology information, ϵ^{rs} of $A_r \cap A_s$ (compute a reduction equivalence as in the Step 1,

$$\epsilon^{rs} : (A_r \cap A_s)_* \overset{trivial}{\Leftarrow} (A_r \cap A_s)_* \overset{Smith}{\Rightarrow} E(A_r \cap A_s)_*$$

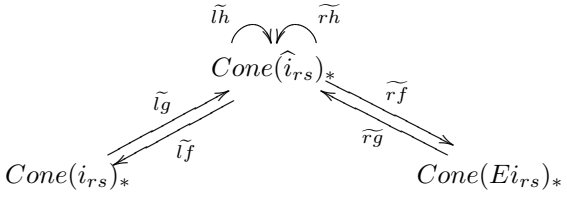
).
3. Compute the cone equivalence (see section 6.4) of the reduction equivalences ϵ^{rs} and $\epsilon^r \oplus \epsilon^s$, through the morphism inclusion $i_{rs} = i_{A_r} \oplus i_{A_s}$:



The result is a reduction equivalence

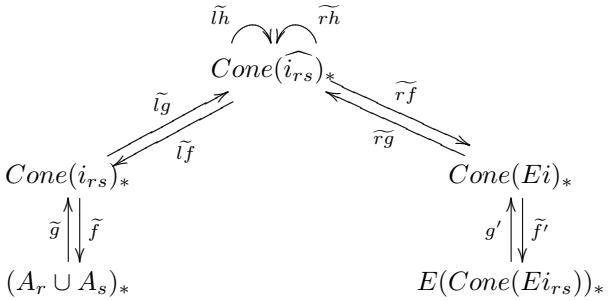
$$\widetilde{\epsilon} : Cone(i_{rs}) \overset{\rho^l}{\Leftarrow} Cone(\widehat{i}_{rs}) \overset{\rho^r}{\Rightarrow} Cone(Ei_{rs})$$

$\widetilde{\epsilon} =$

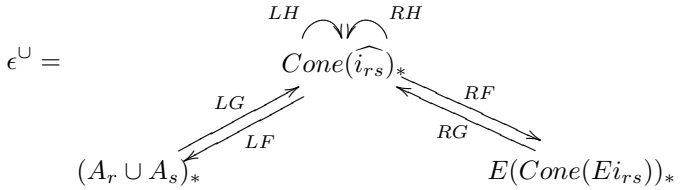


Remember that $\widehat{i} = (rg)\ i_{rs}\ (lf)$ and $Ei_{rs} = (rg)\ (rf)\ i_{rs}\ (lf)\ (lg)$

4. Create a new simplicial complex $A_r \cup A_s$; the union of A_r and A_s
5. Compute the reduction associated to effective short exact sequences (ie. lemma 82) over $Cone(i_{rs})$; $\rho^{lemma} : Cone(i_{rs}) \Rrightarrow (A_r \cup A_s)_*$ and the Homological Smith Reduction over $Cone(Ei_{rs})$; $\rho^{EC} : Cone(Ei_{rs}) \Rrightarrow E(Cone(Ei_{rs}))$
6. Compose the reductions: ρ^l with ρ^{lemma} and ρ^r with ρ^{EC}. The result is the reduction equivalence $\epsilon^{\cup} : (A_r \cup A_s) \Leftrightarrow E(Cone(Ei_{rs}))$ which is the **homology information** of $(A_r \cup A_s)$.



$\Downarrow$ composition of reductions



7. Associate the reduction equivalence $\epsilon^{\cup}$ with the simplicial complex $A_r \cup A_s$.
8. Remove A_r, A_s from the *non_treated_simplices* list and add $A_r \cup A_s$.

Last step The element X at the end of the loop is the original simplicial complex, it has associated its homology information, therefore, the generators and torsion coefficients can be computed.

8.2.1 Reminder of the formulas in the algorithm

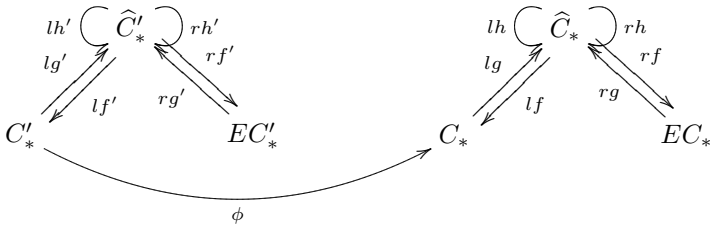
Cone of a morphism

Input: two chain complexes X_*, Y_* and a chain complex morphism $f : X_* \to Y_*$
Output: Chain complex $Cone(f) = C_* = Y_* \oplus X_{*-1}$ with incidence matrix:

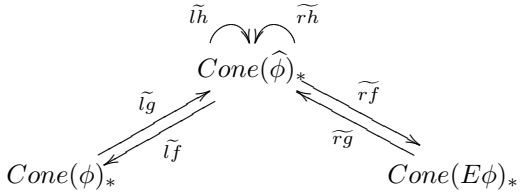
$$D_{C_*} := \begin{bmatrix} D_{Y_*} & f_{*-1} \\ 0 & -D_{X_{*-1}} \end{bmatrix}$$

Cone equivalence

Input:



Output:



where $\widehat{\phi} = (lg)\ \phi\ (lf')$ and $E\phi = (rf)\ (lg)\ \phi\ (lf')\ (rg')$
and

$$\widetilde{rg} = \begin{bmatrix} rg & -(rh)\widehat{\phi}(rg') \\ 0 & rg' \end{bmatrix} \quad \widetilde{rf} = \begin{bmatrix} rf & -(rf)\widehat{\phi}(rh') \\ 0 & rf' \end{bmatrix} \quad \widetilde{rh} = \begin{bmatrix} rh & (rh)\widehat{\phi}(rh') \\ 0 & -(rh') \end{bmatrix}$$

Analogously:

$$\widetilde{lg} = \begin{bmatrix} lg & -(lh)\widehat{\phi}(lg') \\ 0 & lg' \end{bmatrix} \quad \widetilde{lf} = \begin{bmatrix} lf & -(lf)\widehat{\phi}(lh') \\ 0 & lf' \end{bmatrix} \quad \widetilde{lh} = \begin{bmatrix} lh & (lh)\widehat{\phi}(lh') \\ 0 & -(lh') \end{bmatrix}$$

Lemma 82 (reduction in a exact short sequence)

Input: effective short exact sequence

$$0 \longleftarrow R \underset{j}{\overset{\nu}{\rightleftarrows}} S \underset{i}{\overset{\rho}{\rightleftarrows}} T \longleftarrow 0$$

Output: $\rho = (f, g, h) : Cone(i) \Rrightarrow R_*$

- $f = j$
- $h = \rho$
- $g = \nu - \rho\, d_{S_*}\, \nu$

Lemma 82 and Mayer-Vietoris

Input: effective short exact sequence of Mayer-Vietoris:

$$0 \longleftarrow (A \cup B)_* \underset{j_A \ominus j_B}{\overset{\nu}{\rightleftarrows}} A_* \oplus B_* \underset{i_A \oplus i_B}{\overset{\rho}{\rightleftarrows}} (A \cap B)_* \longleftarrow 0$$

$$\begin{array}{rccc} i = i_A \oplus i_B : & (A \cap B)_* & \longrightarrow & A_* \oplus B_* \\ & \sigma & \longmapsto & (\sigma, \sigma) \end{array}$$

$$\begin{array}{rccc} j = j_A \ominus j_B : & A_* \oplus B_* & \longrightarrow & (A \cup B)_* \\ & (\sigma, \tilde{\sigma}) & \longmapsto & \sigma - \tilde{\sigma} \end{array}$$

$$\begin{array}{rccc} \nu : & (A \cup B)_* & \longrightarrow & A_* \oplus B_* \\ & \sigma & \longmapsto & (\sigma|_A, -\sigma|_B + \sigma|_{A \cap B}) \end{array}$$

$$\begin{array}{rccc} \rho : & A_* \oplus B_* & \longrightarrow & (A \cap B)_* \\ & (\sigma, \tilde{\sigma}) & \longmapsto & \tilde{\sigma}|_{A \cap B} \end{array}$$

Output in matrix form:

$$f_k = \boldsymbol{\sigma}^k_{A \cup B} \begin{array}{c} \begin{array}{ccc} \boldsymbol{\sigma}^k_A & \boldsymbol{\sigma}^k_B & \boldsymbol{\sigma}^{k-1}_{A \cap B} \end{array} \\ \begin{pmatrix} j_A & -j_B & 0 \end{pmatrix} \end{array}$$

$$h_k = \begin{array}{c} \begin{array}{ccc} \boldsymbol{\sigma}^k_A & \boldsymbol{\sigma}^k_B & \boldsymbol{\sigma}^{k-1}_{A \cap B} \end{array} \\ \begin{array}{c} \boldsymbol{\sigma}^{k+1}_A \\ \boldsymbol{\sigma}^{k+1}_B \\ \boldsymbol{\sigma}^k_{A \cap B} \end{array} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ \rho & \rho & 0 \end{pmatrix} \end{array}$$

$$g_k = \begin{array}{c} \begin{array}{c} \boldsymbol{\sigma}^k_{A \cup B} \end{array} \\ \begin{array}{c} \boldsymbol{\sigma}^k_A \\ \boldsymbol{\sigma}^k_B \\ \boldsymbol{\sigma}^{k-1}_{A \cap B} \end{array} \begin{pmatrix} \nu \\ \nu \\ -\rho\, d_{A_* \oplus B_*}\, \nu \end{pmatrix} \end{array}$$

Homological Smith Reduction

Input: a chain-complex $\widehat{C}_*$
Output: a reduction ρ

$$\rho = \quad \widehat{C}_* \circlearrowright h \quad \underset{g}{\uparrow}\!\!\downarrow_{f} \quad EC_*$$

$$EN_k = \begin{array}{c} \\ \boldsymbol{w}^{k-1} \\ \boldsymbol{c}^{k-1} \\ \boldsymbol{pw}^{k-1} \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{w}^k & \boldsymbol{c}^k & \boldsymbol{pw}^k \end{array} \\ \begin{pmatrix} 0 & 0 & \boldsymbol{\lambda} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \end{array}$$

$$g_k := P_k = \begin{array}{c} \\ \sigma_1^k \\ \vdots \\ \sigma_{l(k)}^k \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{w}^k & \boldsymbol{c}^k & \boldsymbol{pw}^k \end{array} \\ \begin{pmatrix} & & \\ & & \\ & \dots & \end{pmatrix} \end{array} \qquad f_k := P_k^{-1} = \begin{array}{c} \\ \boldsymbol{w}^k \\ \boldsymbol{c}^k \\ \boldsymbol{pw}^k \end{array} \begin{array}{c} \begin{array}{ccc} \sigma_1^k & \dots & \sigma_{l(k)}^k \end{array} \\ \begin{pmatrix} & & \\ & \dots & \\ & & \end{pmatrix} \end{array}$$

$$h_k = \begin{array}{c} \\ \sigma_1^k \\ \vdots \\ \sigma_{l(k)}^k \end{array} \begin{array}{c} \begin{array}{c} \boldsymbol{pb}^k \end{array} \\ \begin{pmatrix} \\ P_k|_{\boldsymbol{pb}} \\ \\ \end{pmatrix} \end{array} * \boldsymbol{b}^{k-1} \begin{array}{c} \begin{array}{ccc} \sigma_1^{k-1} & \dots & \sigma_{l(k-1)}^{k-1} \end{array} \\ \begin{pmatrix} & P_{k-1}^{-1}|_{\boldsymbol{b}^{k-1}} & \end{pmatrix} \end{array}$$

Sum of reduction equivalences

Input: two reduction equivalences

$$\epsilon^A : A'_* \overset{\rho'_A}{\Leftarrow} \widehat{A}_* \overset{\rho_A}{\Rightarrow} A_*$$

$$\epsilon^B : B'_* \overset{\rho'_B}{\Leftarrow} \widehat{B}_* \overset{\rho_B}{\Rightarrow} B_*$$

Output: $\epsilon^A \oplus \epsilon^B : A'_* \oplus B'_* \overset{\rho'_A \oplus \rho'_B}{\Leftarrow} \widehat{A}_* \oplus \widehat{B}_* \overset{\rho_A \oplus \rho_B}{\Rightarrow} A_* \oplus B_*$
The sum of reductions $\rho_A \oplus \rho_B$ is the sum of their morphisms and their chains (sum of the incidence matrices). If f_A, f_B are morphisms (or incidence matrices), then:

$$f_A \oplus f_B = \begin{bmatrix} f_A & 0 \\ 0 & f_B \end{bmatrix}$$

Composition of reductions

Input: $\rho = (f, g, h) : C_* \Rrightarrow D_*$ and $\rho' = (f', g', h') : C'_* \Rrightarrow D'_*$
Output: $\rho'' = (f'', g'', h'') : C_* \Rrightarrow C''_*$

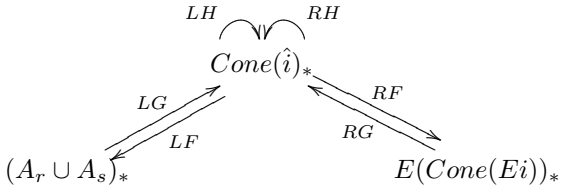
$$f'' = f'f$$

$$g'' = gg'$$

$$h'' = h + gh'f$$

8.3 Proof of the correctness of the algorithm

The proof is immediate from the theoretical basis. We only need to prove that the reduction equivalence computed at the end of each step of the loop is effectively a reduction equivalence:



but the input of the algorithm are reduction equivalences and all the operations performed (sum of reductions, cone equivalence, composition of reductions) preserve the structure of reduction equivalence. Therefore at each step of the loop a reduction equivalence is obtained.

Moreover, observe that the chain-complex $(A_r \cup A_s)_*$ comes from the reduction of Lemma 82 and this lemma assures that the differential operator coincides with the boundary operator of $(A_r \cup A_s)_*$.

Then, the homology of $E(Cone(Ei))_*$ is equivalent to the homology of $(A_r \cup A_s)_*$. We have an isomorphism between $E(Cone(Ei))_*$ and a subchain of $(A_r \cup A_s)_*$ with equivalent homology. This isomorphism is given by the maps

$$(A_r \cup A_s)_* \xrightarrow{(RF)(LG)} E(Cone(Ei))_*$$

and

$$(A_r \cup A_s)_* \xleftarrow{(LF)(RG)} E(Cone(Ei))_*$$

In the same way the correctness of the one step algorithm is proved, because all the operations performed conserve the reduction structure of the input.

Chapter 9

Observations on the complexity

We suppose that the only chain-complex that is computed is the small chain-complex corresponding to the right reduction (the big chain-complex of the reduction equivalence and the small chain-complex of the left reduction are not computed, since we do not require them, see the proposed implementation in annex C).
Observations:

1. The operations carried out in the algorithm are:
   - computation of the Homological Smith Reduction,
   - build matrices by block,
   - composition of morphisms, which corresponds to multiply their matrices.
2. To compute the homology of the union of A_* and B_* we do not perform the Smith Reduction over the union but on the $Cone(Ei)_* = EA_* \oplus EB_* \oplus E(A \cap B)_{*-1}$. The incidence matrices of the $Cone(Ei)$ are proportional to the size of the homology groups of A, B and $A \cap B$.

   When we perform the Homological Smith Reduction to obtain EA_*, EB_* and $E(A \cap B)_*$ we only keep the subbasis correspondent to the homological cycles, the weak-boundaries and the pre-weak-boundaries. Therefore, the dimension of the the incidence matrices of $Cone(Ei)$ are proportional to these elements.

Let n_k^A be the number of k-cells of A and m_k^A be the dimension of the reduced Smith basis of A (i.e, the basis obtained after the Homological Smith Reduction $\{\boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k\}$). Now let notate $smith(m)$ the number of operations required for computing the Smith Normal Form of a matrix with m elements.

We suppose that the homology information of A, B, and $A \cap B$ is already known, i.e, that we already have their Homological Smith Reductions. How many operations are needed to compute the homology of the union using this information?

1. *Compute the Cone Equivalence.* This corresponds to compute two Cone reductions. At each dimension, we need to construct six morphisms by block and one incidence matrix by block. To construct each one of the block matrices we need to multiply two matrices and copy three matrices.

   Then:

   $2 * 7$ matrix multiplications
   $3 * 7$ matrices copied

   In the matrix multiplications always one of the matrices has the number of the columns equal to the number of cells (of the corresponding dimension) of the intersection. Therefore, the multiplications require a number of operations of the order of $O((n_k^A + n_k^B)n_k^{A\cap B})$ for the left reductions and $O((m_k^A + m_k^B)m_k^{A\cap B})$ for the right reductions and for each k.

2. *Compute the reduction* $\rho^{lemma} : Cone(\widehat{\phi}_{rs}) \Rrightarrow (A_r \cup A_s)$ *of Lemma 82.*

   - We need to compute the morphisms i, j, ν and ρ of the effective Mayer-Vietoris short exact sequence. This is done by lineal search on the cells of A_r, A_s, $A_r \cup A_s$, $A_r \cap A_s$.
   - We need to compute the incidence matrices in the canonical bases of $(A_r)_* \oplus (A_s)_*$.
   - We need to perform copies of the morphisms i, j, ν and ρ to construct the morphisms of the reduction.
   - We have to multiply $\rho\ d_{(A_r)_* \oplus (A_s)_*}\ \nu$

3. *Compute the Homological Smith Reduction of* $Cone(Ei)$. As stated before, this is proportional to the dimension of the homology groups and their torsion coefficients. The complexity of this part is $\sum_{k=0}^{n-1} smith((m_k^A + m_k^B + m_k^{A\cap B})(m_{k+1}^A + m_{k+1}^B + m_{k+1}^{A\cap B}))$.

4. *Composition of reductions.* Implies 4 multiplications of matrices and a sum.

In conclusion, the complexity of the algorithm depends highly in the number of cells of the intersection. Therefore, the effectiveness of the algorithm depends on the dimension of the intersection.

If the number of cells of the intersection is small and the reduced Smith bases of A, B, $A \cap B$ are small compared to their number of cells, then the Constructive Mayer-Vietoris algorithm is much faster than computing directly the Smith reduction of the union because the dimensions of the matrices in which the Smith Normal Form must be computed are much smaller.

9.1 Drawbacks

- We need to use the Smith Reduction, even if it is with small matrices, and the complexity of that algorithm is very high (as shown at the end of the part I).

- Memory optimizations can be performed but in any case the matrix corresponding to the homotopy operator of the Homological Smith Reduction can be simplified. The dimensions of this matrix corresponds to the sum of the number of cells of A, B, $A \cap B$.
- The homology generators obtained are usually ill-shaped.

Chapter 10

Optimizations

We can optimize the memory used in the left reduction of the reduction equivalence, i.e, the trivial reductions and the reduction corresponding to Lemma 82. For the right reductions (Homological Smith reductions) we cannot reduce the size of the matrix morphisms and it is not necessary if the homology groups are 'small'.

Here we list some possible optimizations:

- The matrix is not the only possible representation of a morphism. We can also represent a morphism as a lit of instructions. Given an input vector it produces and output vector. So we could represent the trivial reduction in this way. The morphisms f and g could be represented like functions that return their input vector, and h returns the zero-vector. In the same way, we can represent the Mayer-Vietoris morphism i, j, ν and ρ because of their simplicity. This way, the expense in memory is minimal (linear on the number of cells).

  For example, the matrix corresponding to a inclusion i can be expressed as a permutation which tells where to copy the elements and where to put zeros. To represent a permutation we only need to store a vector.

  Then, for the left reductions we should build nested functions in substitution of the matrices by block.

- The matrix of the homotopy operator of the Homological Smith Reduction is very big in general. Unfortunately, we need all the information stored in it. However, with an additional conditions, we can simplify its size.

  The additional condition that we need is that we will not iterate the algorithm to compute, afterwards, a new union homology. If this is the case, then, we can know in advance all the cells involved in the intersections. Observe that we apply the homotopy operator h associated to a reduction only to the intersections. Then, the only relevant information in rh are the columns corresponding to the cells of the intersections, the rest of the columns are not used.

- The incidence matrices of $Cone(Ei)$ are sparse and most of the columns are zero. This could be used to ameliorate the performance of the Smith Reduction.
- The loop can be parallelized.

Chapter 11

Conclusion and future works

In the Constructive Mayer-Vietoris algorithm, in order to be efficient, we need the intersection to be 'small'; this means, basically, that the number of cells of the intersection must be rather inferior to the number of cells of the two subcomplexes. We have seen that the complexity of the algorithm is basically in proportion to the size of the intersection.

This algorithm is a direct application of one of the constructive versions of the long exact sequence of homology, the Lemma 82 [SR06]. The algorithm permits to compute the homology generators and the torsion coefficients of the union of simplicial complexes.
The extensions of this work are:

- This algorithm can be applied to any effective short exact sequence like the one defined by the relative homology (see [Mun99]). Therefore, this algorithm could be conceive in more abstractly.
- Possible memory optimizations.
- Optimization of the computation of the Smith Reduction of $Cone(Ei)$, given the special form of its incidence matrices.
- Adaptation to other combinatorial objects as CW-complex (see [Mun99]).
- To find well-shaped generators.

Part IV

Annexe

Appendix A

List of symbols

$*$	added at the end of something means 'all the dimensions considered'
C_*	chain complex
d	boundary operator (d_k, boundary operator of degree k, $d_k : C_k \to C_{k-1}$)
(C_*, d_*)	chain complex with boundary operator d_*
$C(A)_*$	chain complex associated to the topological space A
A_*	chain complex associated to the topological space A
D_k	matrix of boundary operator of dimension k expressed in canonical basis.
N_k	matrix of boundary operator of dimension k expressed in Smith normal form.
P_k	matrix of change of basis with input basis the Smith basis an output basis the canonical basis (in dimension k).
EN_k	boundary matrix after Homological Smith reduction
EC_*	reduced chain of C_* (chain obtained after the Homological Smith reduction of the chain C_*)
$Cone(\phi)$	cone of the morphism ϕ
$\rho = (f, g, h)$	reduction with morphism f, g and homotopy operator h
f, g	chain morphism associated to a reduction
h	homotopy operator (normally associated to a reduction)
$\widehat{C} \Rrightarrow C$	reduction
$\{\sigma_1^k, ...\sigma_{l(k)}^k\}$	canonical basis of dimension k
$\{\boldsymbol{b}^k, \boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k, \boldsymbol{pb}^k\}$	Smith basis of dimension k (basis in which the Smith normal form is expressed),($\boldsymbol{pb}^k = \{pb_1^k, ...pb_{l(k)}^k\}$)
$\{\boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k\}$	Reduced Smith basis of dimension k (basis in which EN_k is expressed)
$C_k = [\sigma_1^k, ..., \sigma_{l(k)}^k]$	module C_k generated by $\sigma_1^k, ...\sigma_{k(l)}^k$

Appendix B

How to compute the Smith Normal Form

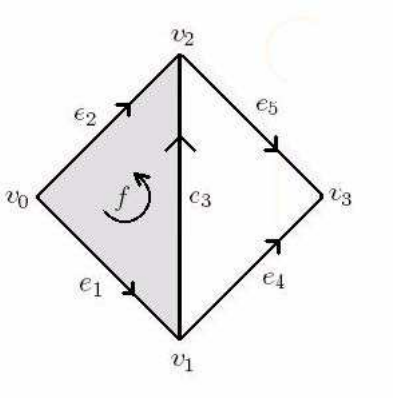
B.1 Reminder about chain complex

Given a simplicial complex X of dimension n (finite), for each $k < n$ the module formed by the k-faces, is denoted $C_k(X)$ or simply X_k. If $\{\sigma_1^k, \ldots, \sigma_{l(k)}^k\}$ are the set of all k-faces with a given orientation of X, then:

$$X_k = \mathbb{Z}\sigma_1 \oplus \ldots \oplus \mathbb{Z}\sigma_{l(k)}$$

. So, $\{\sigma_1^k, \ldots, \sigma_{l(k)}^k\}$ is the basis of the module X_k and $\mathbb{Z}$ is its ring of coefficients. So X_k is formed by a finite linear combination of k-faces with coefficients in $\mathbb{Z}$.

Let us notice that normally an element of X_k has no geometrical meaning. For example, in the figure below $z = 4e_1 - e_2$ makes no geometrical sense, though this is algebraically correct. There is a special case of interest: when a **cycle** is formed; for example, in the same figure, that would lead to $e_1 + e_3 - e_2$.



B.2 How to construct the chain complex and the border operator.

Input: finite simplicial complex X of dimension n. To define the chain complex and construct the border operator matrix is achieved through the following steps:

1. Orient the faces of the simplicial complex (give an order to the vertices),
2. Define the modules of the chain complex at each dimension using a 'canonical' basis,
3. Express the border operator in matrix form using the 'canonical basis'.

B.2.1 How to give an orientation to the faces of a simplicial complex

Each k-face (k-simplex, $k < n$) is determined by a set of $(k+1)$ vertices, say $\{v_0, \ldots, v_k\}$. When defining a k-face, not only the set of vertices is important but also the **order** in which they are given. For example, $\{v_0, v_1, v_2, \ldots, v_k\}$ is the same set as $\{v_1, v_0, v_2, \ldots, v_k\}$ and defines the same k-face, but not the same **orientation**. From now on, we will only consider oriented faces.

Remark 0. There are only two possible orientations for a given face. A permutation of two elements changes the orientation.

So, **to give an orientation to the faces is equivalent to give an order of the vertices**. This is achieved globally. Given a simplicial complex X with a set of vertices δ, we give an order to the vertices, i.e., $\forall u, v \in \delta, \quad u < v$ or $v < u$. This order of the vertices **induces an orientation** on all the faces for all the dimensions since we only have to consider the vertices defining that face, in the order of the prescribed in the set. For example, if a 3-face t, is determined by vertices $[u, v, s, p]$, this means that $u < v < s < p$ in the order of δ.

Remark 1. Remember that giving an orientation of the faces does not mean that the simplicial complex is **oriented**. In fact, there are some simplicial complexes that are not even orientable, like the Moebius strip.

Remark 2. Faces must be oriented because a module is constructed from them. This module has the faces as elements and $\mathbb{Z}$ as coefficients. Given an orientation of the faces, multiplying by (-1) a face has the following meaning: it produces a orientation change of the face.

B.2.2 How to define a module: give a basis

For each dimension $k \leq dim(X)$, the objective is to define $C_k(X)$, to this end, only a basis needs be specified since here (finite simplicial complex), $C_k(X)$ is free, i.e., is generated by a finite basis.
The natural basis for each dimension will be given by the faces of that dimension. First, an order of the vertices of X must be set, that induces an order in the faces. For each dimension k, the set of oriented k-faces gives a basis of $C_k(X)$ that is called the **'canonical' or natural** basis.

Remark. Observe that the canonical basis is **not** unique, given a different order of the faces, another basis is produced. However, this does not alter at all the homology and in any case, even if the choice of order is not unique, the construction of the basis is still 'natural'. So, from now on, when talking about a face, implicitly about it is an oriented face; and when talking about a canonical basis it means one of the possible canonical bases.

For each dimension k, a canonical basis $\{\sigma_1^k, ..., \sigma_{l(k)}^k\}$ is formed by the oriented k-faces of X. **Remember that a basis is not only a set but the order in which this set is given is also important**, i.e., as it happened with the vertices, the basis $\{\sigma_1^k, \sigma_2^k, \sigma_3^k, \ldots, \sigma_{l(k)}^k\}$ is not the same as $\{\sigma_2^k, \sigma_1^k, \sigma_3^k, \ldots, \sigma_{l(k)}^k\}$ because the two first elements have been permuted. In this case, the order of the faces is not related to the orientation but to the way of representing an element of $C_k(X)$. This is of capital importance when constructing the boundary matrix on this basis.

Example. Let's assume that a basis of $C_2(X)$ is $\{e_1, e_2, e_3, e_4\}$ and suppose that $z = 4e_1 - e_2 + 5e_4$, then z can be written as $z = (4, -1, 0, 5)$ in this basis and z is represented as a vector. Observe that z can be written as $z = 5e_4 - e_2 + 4e_1$, i.e., changing the order of the summands produces the same element. However, $z = (4, -1, 0, 5) \neq (5, -1, 0, 4) = 5e_1 - e_2 + 4e_4$. Each basis comes with an order of the elements that permits us to represent the elements of the module as arrays. This leads us to represent the boundary operator as a matrix.

B.2.3 Boundary matrices of border operators

Considering again the example of the previous subsection, given the canonical basis of a given dimension k, $\{\sigma_1^k, \ldots, \sigma_{l(k)}^k\}$, each element of the module $C_k(X)$, $z = a^1\sigma_1^k + a^2\sigma_2^k + \ldots + a^{l(k)}\sigma_{l(k)}^k$ can be represented as an array: $(a^1, a^2, \ldots, a^{l(k)})$.

To construct the border matrix of dimension k, let's consider the k-dimensional canonical basis and, for each element of it, σ_i^k, we have to compute its value by the boundary operator $d_k(\sigma_i^k) = \sum_j b_i^j \sigma_j^{k-1}$. Finally, the i_{th}-column of the border matrix will be $(b_i^1, \ldots, b_i^{l(k-1)})$.

For a given (oriented) k-face $\sigma_i^k = [v_0, \ldots, v_k]$ the border operator is defined as follows:

$$d_k([v_0, \ldots, v_k]) = \sum_{i=0}^{k} (-1)^k [v_0, \ldots, v_{i-1}, v_{i+1}, \ldots v_k]$$

for each i. $[v_0, \ldots, v_{i-1}, v_{i+1}, \ldots, v_k]$ is an element of the canonical basis of $C_{k-1}(X)$. In matrix form, it comes:

$$D_k = \begin{array}{c} \\ \sigma_1^{k-1} \\ \vdots \\ \sigma_{l(k)}^{k-1} \end{array} \begin{array}{c} \begin{array}{ccc} \sigma_1^k & \ldots & \sigma_{l(k)}^k \end{array} \\ \begin{pmatrix} & \vdots & \\ \ldots & \eta_{j,i}^k & \ldots \\ & \vdots & \end{pmatrix} \end{array}$$

where:

$$\eta_{j,i}^k = \begin{cases} 0 & \text{if } \sigma_j^{k-1} \text{ is not in the boundary of } \sigma_i^k; \\ 1 & \text{if } \sigma_j^{k-1} \text{ is in the boundary of } \sigma_i^k; \\ -1 & \text{if } -\sigma_j^{k-1} \text{ is in the boundary of } \sigma_i^k. \end{cases}$$

B.3 How to find the homological information

B.3.1 Smith normal form

This section is derived from [Mun99]. Tables are from [Pel06] (with a minor correction in one of them).

General algebraic context: existence of a Smith normal form in a module

Remember that the objective is to express the border operator matrix in the form:

$$\begin{pmatrix} 0 & \boldsymbol{\lambda} & 0 \\ 0 & 0 & Id \\ 0 & 0 & 0 \end{pmatrix}$$

where $\boldsymbol{\lambda}$ is a diagonal matrix with $\lambda_r, \ldots, \lambda_1$ in the diagonal ($\lambda_i \in \mathbb{Z}$ and $\lambda_i > 1$ and λ_i divides λ_{i+1}).

However, the classical result of Smith is to find a matrix of the form:

$$\begin{pmatrix} Id & 0 & 0 \\ 0 & \boldsymbol{\lambda} & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

.
The changes from one to the other are minor, but essential to have the good basis to study homology. We will describe how to obtain the 'classical' Smith normal form.
The existence of this form is an algebraic theorem, the proof of it can be found, for example, in [Mun99]
To obtain this normal form, a change of basis has to be performed, which is equivalent to perform changes in the rows and columns of the matrix. However, attention must be paid to perform 'legal' changes, i.e., changes that can be 'undone' in our coefficient space $\mathbb{Z}$. For example, if a row is multiplied by 2, this is an action that cannot be undone in general, since to undo it the row should be divided by 2 but $1/2 \notin \mathbb{Z}$. Hence, the elementary rows operations are:

1. Exchange rows i and j,
2. Multiply a row i by (-1) (observe that (-1) is invertible in $\mathbb{Z}$),
3. Replace a row i by $row_i + q\ row_j$, where q is an integer and $j \neq i$.

The same elementary operations can be carried out on the columns.

Observe that each one of these elementary operations produce a change of basis. For example, if the columns of the matrix are expressed in the basis $\{\sigma_1^k, \ldots, \sigma_s^k\}$ then, permuting two columns means to permute the two respective elements of the basis. Hereunder, a table summerizes the transformations of the change of basis matrix.

Operation on the matrix	**Operation on the $(k-1)$-basis**	**Operation on the k-basis**
Exchange(row_i, row_j)	Exchange$(\gamma_i^{k-1}, \gamma_j^{k-1})$	
$row_i \leftarrow -row_i$	$\gamma_i^{k-1} \leftarrow -\gamma_i^{k-1}$	
$row_i \leftarrow row_i + q\ row_j$	$\gamma_i^{k-1} \leftarrow \gamma_j^{k-1} - q\ \gamma_i^{k-1}$	
$col_i \leftrightarrow col_j$		$\gamma_i^k \leftrightarrow \gamma_j^k$
$col_i \leftarrow -col_i$		$\gamma_i^k \leftarrow -\gamma_i^k$
$col_i \leftarrow col_i + q\ col_j$		$\gamma_i^k \leftarrow \gamma_i^k + q\ \gamma_j^k$

For example, imagine that a k basis is $\{\gamma_1^k, \gamma_2^k, \gamma_3^k, \gamma_4^k\}$ and the next changes are performed consecutively in the matrix:

1. Exchange$(column_2, column_4)$,
2. $column_2 \leftarrow column_2 - 3column_1$,
3. $column_4 \leftarrow -column_4$.

The changes in the basis will be:

1. $\{\gamma_1^k, \gamma_4^k, \gamma_3^k, \gamma_2^k\}$,
2. $\{\gamma_1^k, \gamma_4^k - 3\gamma_1, \gamma_3^k, \gamma_2^k\}$,
3. $\{\gamma_1^k, \gamma_4^k - 3\gamma_1, \gamma_3^k, -\gamma_2^k\}$.

In computational terms, we need to store all these changes of basis in an effective way. This is achieved with the **change of basis matrix** P_k. In this matrix, the columns are expressed in the Smith basis and the rows expressed in the canonical basis.

$$P_k = \begin{array}{c} \\ \sigma_1 \\ \vdots \\ \sigma_{l(k)} \end{array} \begin{array}{c} \begin{array}{ccc} s_1 & \dots & s_{l(k)} \end{array} \\ \begin{pmatrix} & & \\ & & \\ & & \end{pmatrix} \end{array}$$

.

This matrix tells how to express an element of the Smith basis s_i in terms of the canonical basis. If the i_{th} column has coefficients $(\tau_{i,1}, \tau_{i,2}, \dots, \tau_{i,l(k)})$, then $s_i = \tau_{i,1}\ \sigma_1 + \tau_{i,2}\ \sigma_2 + \dots + \tau_{i,l(k)}\ \sigma_{i,l(k)}$.

Every elementary operation in the rows or columns of the border matrix will be translated in an operation in the change of basis matrix. P_k is the matrix of change of basis of dimension k. The next table expresses the changes to be performed in those matrices when the border matrix is modified:

Operation on N_k	**Operation on P_{k-1}**	**Operation on P_k**
$row_i \leftrightarrow row_j$	$col_i \leftrightarrow col_j$	
$row_i \leftarrow -row_i$	$col_i \leftarrow -col_i$	
$row_i \leftarrow row_i + k\ row_j$	$col_i \leftarrow col_j - k\ col_i$	
$col_i \leftrightarrow col_j$		$col_i \leftrightarrow col_j$
$col_i \leftarrow -col_i$		$col_i \leftarrow -col_i$
$col_i \leftarrow col_i + k\ col_j$		$col_i \leftarrow col_i + k\ col_j$

Initially, the starting point is the border operator matrix expressed in the canonical basis. So, initially no change of basis has been perform. That is why initially $P_k = Id$ for each k, when performing the transformation this matrix will be transformed too.

Using the previous example, the k basis is $\{\gamma_1^k, \gamma_2^k, \gamma_3^k, \gamma_4^k\}$ and the next changes are performed consecutively in the matrix:

1. Exchange$(column_2, column_4)$,
2. $column_2 \leftarrow column_2 - 3column_1$,
3. $column_4 \leftarrow -column_4$.

Then, if we start with P_k as the identity, it comes:

Input:

$$P_k = \begin{array}{c} \\ \gamma_1^k \\ \gamma_2^k \\ \gamma_3^k \\ \gamma_4^k \end{array} \begin{array}{c} \begin{array}{cccc} \gamma_1^k & \gamma_2^k & \gamma_3^k & \gamma_4^k \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

The changes performed produce:

1. Exchange($column_2, column_4$)

$$P_k = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

2. $column_2 \leftarrow column_2 - 3column_1$

$$P_k = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -3 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

3. $column_4 \leftarrow -column_4$

$$P_k = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & -3 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

The columns of this matrix are expressed in the new basis, say $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$, and reading the columns we find that: $\alpha_1 = \gamma_1, \alpha_2 = -3\ \gamma_3 + \gamma_4, \alpha_3 = \gamma_3, \alpha_4 = -\gamma_2$, which is exactly the basis found before.

Steps to obtain the Smith normal form (extracted from [Mun99])

Extracted from Munkres: Given a matrix $A = (a_{ij})$ of integers, not all zero, let $\alpha(A)$ denote the smallest non-zero element of the set of numbers $|a_{ij}|$. a_{ij} is called a minimal entry of A if $|a_{ij}| = \alpha(A)$.
The reduction procedure consists of two steps. The first one brings the matrix to a form where $\alpha(A)$ is as small as possible. The second one reduces the dimensions of the matrix involved.

1. We seek to modify the matrix by elementary operations so as to decrease the value of the function α. We prove the following:

If the number $\alpha(A)$ fails to divide some entry of A, then it is possible to decrease the value of α by applying elementary operations to A; and conversely.

The converse is easy. If the number $\alpha(A)$ divides each entry of A, then it will divide each entry of any matrix B obtained by applying elementary operations to A. In this situation, it is possible to reduce the value of α by applying elementary operations.

To prove the result itself, we suppose $a_{i,j}$ is a minimal entry of A that fails to divide some entry of A. If the entry $a_{i,j}$ fails to divide some entry $a_{k.j}$ in its column, then we perform a division, writing $\frac{a_{k,j}}{a_{i,j}} = q + \frac{r}{a_{i,j}}$ where $0 < |r| < |a_{i,j}|$. Signs do not matter here; q and r may be either positive or negative. We then replace (row k) of A by $(row\ k) - q(row\ i)$. The result is to replace the entry $a_{k,j}$ in the k^{th} row and j^{th} column of A by $a_{kj} - q\ a_{ij} = r$. The value of α for this new matrix is at most $|r|$, which is less than $\alpha(A)$.

A similar argument applies if a_{ij} fails to divide some entry in its row. Finally, suppose a_{ij} divides each entry in its row and each entry in its column, but fails to divide the entry a_{st}, where $s \neq i$ and $t \neq j$. Consider the following four entries of A:

$$\begin{matrix} a_{ij} & \dots & a_{it} \\ \vdots & & \vdots \\ a_{sj} & \dots & a_{st} \end{matrix}$$

Because a_{ij} divides a_{st}, we can by elementary operations bring the matrix to the form where the entries in these four places are as follows:

$$\begin{matrix} a_{ij} & \dots & a_{it} \\ \vdots & & \vdots \\ 0 & \dots & a_{st} + la_{it} \end{matrix}$$

If we then replace (row i) of this matrix by (row i)+(row s), we are back to the precious situation, where a_{ij} fails to divide some entry in its row.

Step 2. At the beginning of this step, we have a matrix A whose minimal entry divides every entry of A. Apply elementary operations to bring a minimal entry of A to the upper left corner of the matrix and to make it positive. Because it divides all entries in its row and column, we can apply elementary operations to make all the other entries in its row and column into zeros. Note that at the end of this process, the entry in the upper left corner divides all entries of the matrix.

One now begins Step 1 again, applying it to the smaller matrix obtained by ignoring the first row and first column of our matrix.

Step 3. The algorithm terminates either when the smaller matrix is the zero matrix or when it disappears. At this point our matrix is in normal form.

Last step. In order to have the good basis, we need a minor change in the algorithm. We must perform the same algorithm starting from the upper-right corner of the matrix and following the antidiagonal. Next, we need to permute the non-zero rows.

Remark. The steps described to obtain the Smith normal form are not deterministic, i.e., there are different stages where there is more than one possible choice. This does not alter the final matrix since it is unique. However, this implies that **the Smith basis** (and, so, the change of basis matrices) **are not unique**.

B.4 Example

Example 14. The Smith Reduction algorithm

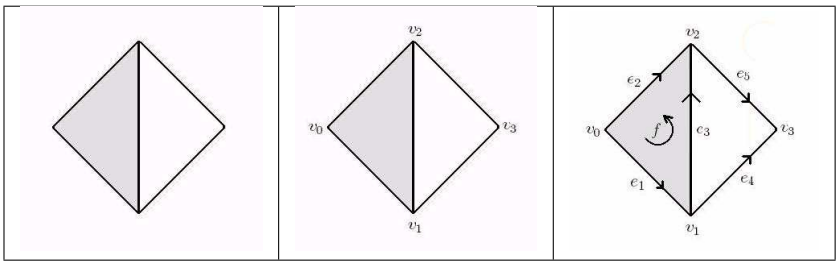


Table B.1: Complex description

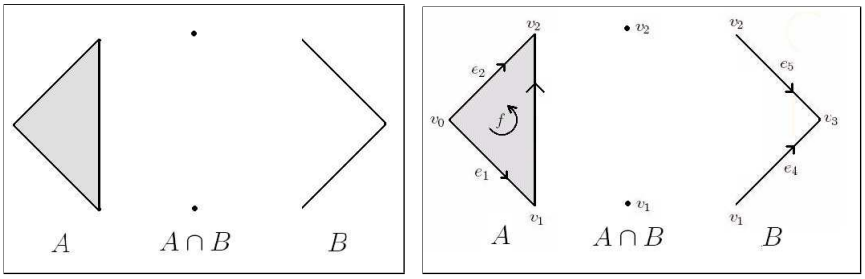


Table B.2: Subcomplex description

$$D_1^A = \begin{array}{c} \\ v_0 \\ v_1 \\ v_2 \end{array} \begin{array}{c} \begin{array}{ccc} e_1 & e_2 & e_3 \end{array} \\ \begin{pmatrix} -1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & 1 \end{pmatrix} \end{array}$$

From left to right:

$row_1 \leftrightarrow row_3$	$column_2 \leftarrow column_2 - column_3$
$\begin{matrix} \\ v_2 \\ v_1 \\ v_0 \end{matrix} \begin{pmatrix} e_1 & e_2 & e_3 \\ 0 & \mathbf{1} & 1 \\ 1 & 0 & -1 \\ -1 & -1 & 0 \end{pmatrix}$	$\begin{matrix} \\ v_2 \\ v_1 \\ v_0 \end{matrix} \begin{pmatrix} e_1 & e_2 - e_3 & e_3 \\ 0 & 0 & 1 \\ 1 & 1 & \mathbf{-1} \\ -1 & -1 & 0 \end{pmatrix}$
$row_2 \leftarrow row_2 + row_1$	$column_1 \leftarrow column_1 - column_2$
$\begin{matrix} \\ v_2 - v_1 \\ v_1 \\ v_0 \end{matrix} \begin{pmatrix} e_1 & e_2 - e_3 & e_3 \\ 0 & 0 & 1 \\ \mathbf{1} & 1 & 0 \\ -1 & -1 & 0 \end{pmatrix}$	$\begin{matrix} \\ v_2 - v_1 \\ v_1 \\ v_0 \end{matrix} \begin{pmatrix} e_1 - e_2 + e_3 & e_2 - e_3 & e_3 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & \mathbf{-1} & 0 \end{pmatrix}$
$row_3 \leftarrow row_3 + row_2$	$row_2 \leftrightarrow row_1$
$\begin{matrix} \\ v_2 - v_1 \\ v_1 - v_0 \\ v_0 \end{matrix} \begin{pmatrix} e_1 - e_2 + e_3 & e_2 - e_3 & e_3 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{matrix} \\ v_1 - v_0 \\ v_2 - v_1 \\ v_0 \end{matrix} \begin{pmatrix} e_1 - e_2 + e_3 & e_2 - e_3 & e_3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$

$$N_1^A = \begin{matrix} \\ v_1 - v_0 \\ v_2 - v_1 \\ v_0 \end{matrix} \begin{pmatrix} e_1 - e_2 + e_3 & e_2 - e_3 & e_3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

Here we have made explicit the changes of basis while transforming the matrix. So the change of basis obtained are: We see that the rows of the Smith matrix, N_1^A, are expressed in the basis $\{v_1 - v_0, v_2 - v_1, v_0\}$. Denoting its elements as $x_0 = v_1 - v_0, x_1 = v_2 - v_1, x_2 = v_0$ we obtain the matrix of change of basis in A_0:

$$P_0^A = \begin{matrix} \\ v_0 \\ v_1 \\ v_2 \end{matrix} \begin{pmatrix} x_0 & x_1 & x_2 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

In the same way, the Smith basis of A_1 is $\{\alpha_1 = e_1 - e_2 + e_3, \alpha_2 = e_2 - e_3, \alpha_3 = e_3\}$ and the matrix of change of basis is:

$$P_1^A = \begin{matrix} \\ e_1 \\ e_2 \\ e_3 \end{matrix} \begin{pmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ 1 & 0 & 0 \\ -1 & 1 & 0 \\ 1 & -1 & 1 \end{pmatrix}$$

We see that during the process of the Smith algorithm we can also obtain the change of basis matrices. I the same way we can also obtain their inverses.

$$(P_0^A)^{-1} = \begin{array}{c} \\ x_0 \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{ccc} v_0 & v_1 & v_2 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \end{array} \qquad (P_1^A)^{-1} = \begin{array}{c} \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{array} \begin{array}{c} \begin{array}{ccc} e_1 & e_2 & e_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \end{array}$$

Next, we have to obtain the Smith matrix of D_2^A. First, we change the basis of the rows of the matrices by multiplying by the $(P_1^A)^{-1}$:

$$D_2^A = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \end{array} \begin{array}{c} \begin{array}{c} f \end{array} \\ \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix} \end{array} \qquad N_2^A = (P_1^A)^{-1} D_2^A = \begin{array}{c} \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{array} \begin{array}{c} \begin{array}{c} f \end{array} \\ \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \end{array}$$

and in this case we obtain the matrix in Smith form and we do not need to perform any additional change of basis.
$P_2^A = 1$ (1×1 identity matrix).

Example 15. Smith Reduction algorithm with change of basis

$$D_1^B = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{cc} e_4 & e_5 \end{array} \\ \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \end{pmatrix} \end{array} \quad P_0^B = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \end{array} \quad P_1^B = \begin{array}{c} \\ e_4 \\ e_5 \end{array} \begin{array}{c} \begin{array}{cc} e_4 & e_5 \end{array} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{array}$$

Smith algorithm:

operation on matrix	**Change in P_0^B**	**Change in P_1^B**
$row_1 \leftrightarrow row_3$	$col_1 \leftrightarrow col_3$	
$\begin{pmatrix} \mathbf{1} & 1 \\ 0 & -1 \\ -1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$	
$col_1 \leftarrow col_1 - col_2$		$col_1 \leftarrow col_1 - col_2$
$\begin{pmatrix} 0 & 1 \\ 1 & \mathbf{-1} \\ -1 & 0 \end{pmatrix}$		$\begin{matrix} \\ e_4 \\ e_5 \end{matrix} \begin{pmatrix} \beta_1 & \beta_2 \\ 1 & 0 \\ -1 & 1 \end{pmatrix}$
$row_2 \leftarrow row_2 + row_1$	$col_1 \leftarrow col_1 - col_2$	
$\begin{pmatrix} 0 & 1 \\ 1 & 0 \\ \mathbf{-1} & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 1 \\ -1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$	
$row_3 \leftarrow row_3 + row_2$	$col_2 \leftarrow col_2 - col_3$	
$\begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & -1 & 1 \\ -1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$	
$row_2 \leftrightarrow row_1$	$col_2 \leftrightarrow col_1$	
$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix}$	$\begin{matrix} \\ v_1 \\ v_2 \\ v_3 \end{matrix} \begin{pmatrix} \gamma_1 & \gamma_2 & \gamma_3 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & 0 \end{pmatrix}$	

Appendix C

Proposed data structure for the implementation

CHomology
private:
cells (enumerated) [simplicial complex description]
dimension: integer
/ dimension */*
reductions: CReduction_equivalence
/ information about the chain complex */*
generators_free: pointers to cells;
generators_torsion: pointers to cells
public:
/ Constructors */*
CHomology();
/ initialise everything with null */*
CHomology($vector < cells >$);
/ init cells and dimension*/*
CHomology(CHomology A, CHomology B);
/ init cells and dimension of union*/*
void init_reductions();
/ left_reduction is trivial, right_reduction is Smith */*
Matrix get_incidence_matrix_of_dimension(integer k);
$vector < Matrix >$ construct_all_incidence_matrix();
get_dimension();
get_reductions();
void compute_generators();
*/*Cond: right-hand reduction of equiv is of Smith type */*

draw_homology();
*/*Cond: hom is initialised */*
S̃implicial_complex();

CReduction

private:
small_chain : $vector < Matrix* > [dimension - 1]$
h : $vector < Matrix* > [dimension]$
f : $vector < Matrix* > [dimension]$
g : $vector < Matrix* > [dimension]$
type : char
/ Possible values: c (cone), t (trivial), s (smith), l (lemma), u (unkwon) */*
w,pw,c : pair_of_integers
*/*Cond: type=s */*
Matrix get_small_chain_at(int i);
void set_small_chain_sum($vector < Matrix* >$ Ch1, $vector < Matrix* >$ Ch2);
*/*sum Ch1 with Ch2 */*
void set_small_chain_cone($vector < Matrix* >$ morphism, $vector < Matrix* >$ Ch1, $vector < Matrix* >$ Ch2);
/ cone of Ch1, Ch2 */*
public:
CReduction();
/ init everything with null */*
CReduction($vector < nbcells > [n]$)
*/*trivial reduction */*
CReduction(CReduction* R1, CReduction* R2, $vector < Matrix* > [n]$ morphism)
/ Cone */*
CReduction(CExact_sequence* ES);
*/*lemma 82 */*
CReduction($vector < Matrix* >$ Incidence_matrix);
*/*Smith reduction */*
CReduction(CReduction* R1, CReduction* R2);
/ somme R1 and R2 */*
void composition_of_reductions(CReduction* R1, CReduction* R2);
*/*R1 is modified with the composition of R1 and R2 */*

CReduction_equivalence

private:
right_reduction : CReduction
left_reduction : CReduction

public:
CReduction_equivalence();
*/*init everything with null */*
CReduction_equivalence(CReduction* Rleft, CReduction* Rright);
CReduction_equivalence(CReduction_equivalence* A, CReduction_equivalence* B);
*/*sum of A, B */*
CReduction_equivalence(CReduction_equivalence* A, CReduction_equivalence* B,
$vector < Matrix* >$ Inclusion);
/ cone */*

CExact_sequence
private:
CH1: CHomology*
CH2: CHomology*
m_nu : Morphism
m_j : Morphism
m_i : Morphism
m_rho : Morphism
public:
CExact_sequence_Mayer_Vietoris(CHomology *A,CHomology *B,
CHomology *AinterB, CHomology *AunionB);

See the diagram in figure C.1

C.0.1 Proposed functions implementation

CHomology

CHomology();

/ initialise the cells and dimension, this function depends only on the data type and the way data is stored */ (in our case IS data)*
/ initialise everything with null */*

CHomology(cells<vector>);
/ initialise the cells and dimension, this function depends only on the data type and the way data is stored */ (in our case IS data)*

CHomology (node A, node B); */* init cells and dimension of union A, B */*

void init_reductions()
if (cells not initialise) error; */*Chain complex with canonical incidence matrix */*
vector<Matrix>canonical_chain = construct_all_incidence_matrix() ; */* local variable */*
vector<int>number_of_cells_by_dimension = (information given by IS data)
set_reductions(CReduction_equivalence (CReduction(number_of_cells_by_dimension) , CReduction(canonical_chain)); */*trivial reduction at left and smith at right */*
}
*/*ATTENTION: we do not store the canonical_chain, once we get out of the function the information is lost, that is something to decide, if we store the canonical basis anywhere or not */*

void init_reductions(CReduction_equivalence equiv){
set_reductions(equiv);
}

compute_generators(){
/ condition */*
if (reductions.get_RR.get_type != s)
error("cannot compute homology; right_reduction is not of type Smith")
exit()
if(reductions.get_LR.get_ype != l and reductions.left_reduction.type != t)
error("cannot compute homology; left_reduction is not of correct type");
exit()

for each dimension k <dimension{
for each element c of Smith reduction do{
aux = Multiply (reductions.right_reduction.g(k) , c);
aux = Multiply (reductions.left_reduction.f(k), aux);
add aux to generators_cycles[k];
}
for each element (w,lambda) of Smith reduction do{
aux = Multiply (reductions.right_reduction.g(k), w);
aux = Multiply (reductions.left_reduction.f(k), aux);
add (aux, lambda) to generators_weak_cycles[k];
}
/ lambda is the torsion coefficient*/*
}

}

CReduction

CReduction(); */* init everything with null */*

CReduction(vector<int>number_of_cells_by_dimension){ */* Trivial reduction */*
set_type(t);
set_f(list_identity_matrix(number_of_cells_by_dimension));
set_g (list_identity_matrix(number_of_cells_by_dimension));
set_h (list_zero_matrix(number_of_cells_by_dimension));
set_small_chain (null);
}

CReduction(Morphism *mor, CReduction *R1, CSReduction *R2){ */* Cone reduction */*
set_type(c);
Morphism small_morph = Composition(&R2.get_f(), Composition(mor, &R1.get_g())); */* *Composition() = multiply vector<matrix>*/*
set_small_chain_cone(*small_morph, *(&R1.get_small_chain()), *(&R2.get_small_chain()
));

set_f(Block(&R1.get_f(k), Composition(-(&R1.get_f(k)),Composition(&mor, (&R2.get_h(k))));zero_ma
&R2.get_f(k)));
set_g(Block(&R1.get_g(k), Composition(-(&R1.get_h(k)),Composition(&mor, (&R2.get_g(k))));
zero_matrix_indicator, &R2.get_g(k)));
set_h(Block(&R1.get_h(k), Composition((&R1.get_h(k)),Composition(&mor, (&R2.get_h(k))));
zero_matrix_indicator, -&R2.get_f(k)));
}
}

CReduction(CExact_sequence *seq){ */* Reduction lemma 82 */*
set_type(l);
set_small_chain_sum (&seq.CH1.compute_all_incidence_matrix()), &seq.CH2.compute_all_incidence_matrix())
);
set_f(seq.get_m_j());
set_h(seq.get_m_rho());
set_g(seq.get_m_nu() - Composition(seq.get_m_rho() , Composition(get_small_chain,
seq.get_m_nu())); */* we need to define the operation '-' */*
free(small_chain);
small_chain = null;
}

```
    CReduction( vector<matrix>CC ){ /* Homological Smith Reduction */
type = s;
{N, P, P_inv, small_chain} = smith(CC);
{pb,b, pw, w, c} = basis_classification(N);
construct_f(P_inv);
construct_g(P);
construct_h(P, P_inv);
}


    CReduction(CReduction *R1, CReduction *R2){
if ( &R1.get_type == &R2.get_type) set_type(&R1.get_type);
else set_type(u);
set_small_chain_sum( &R1.get_small_chain, &R2.get_small_chain);
set_f( sum( &R1.get_f() , &R2.get_f() ) );
set_g( sum ( &R1.get_g() , &R2.get_g() ) );
set_h( sum( &R1.get_h() , &R2.get_h() ) );
}


    void composition_of_reductions(CReduction *R2){
set_type(&R2.get_type);
set_small_chain( &reduct2.get_small_chain);
if (&this.get_type = t) this = &R2; return;
if (&R2.get_type = t) return;
set_f( Composition( &R2.get_f() , this.get_f() ) );
set_g( Composition( this.get_g(), &R2.get_g() ) );
set_h( (this.get_h()) + Composition( this.get_g() , Composition( &R2.get_h(),        this.get_f()
) ) );
}


    set_small_chain_sum(vector<Matrix*>* CC1, vector<Matrix*>* *CC2){
if (CC1 == null or CC2 == null) { small_chain = null; return; }
small_chain = Block(CC1, matrix_zero_indicator; matrix_zero_indicator, CC2)
}
 set_small_chain_cone(vector<Matrix*>* Morphism, vector<Matrix*>* Ch1, vec-

tor<Matrix*>* Ch2){
if ( Ch1 == null or Ch2 == null ) small_chain = null;
small_chain = Block (&Ch2[k] , &mor[k]; matrix_zero_indicator, - &CH1[k-1] ];
```

```
}
```

get_N(integer k){
if (k<1 or k>dimension) return null_matrix;
return N[k]
}

CReduction_equivalence

CReduction_equivalence(); */* initialise everything with null */*

CReduction_equivalence(CReduction* Rleft, CReduction* Rright);

CReduction_equivalence(CReduction_equivalence *A, CReduction_equivalence *B, Morphism *morph){ {/* cone equivalence*/}
vector<Matrix*>big_morph = Composition(&B.get_LR.get_g() , Composition(&morph , &A.get_LR.get_f()));
set_LR(CReduction(big_morph, &A.get_LR, &B.get_LR)); */* cone reduction */*
set_RR(CReduction(big_morph, &A.get_RR, &B.get_RR));
}

CReduction_equivalence (CReduction_equivalence *A, CReduction_equivalence *B){ */*sum*/*
set_RR(CReduction (&A.get_RR, &B.get_RR)); {/* sum of reductions */}
set_LR(CReduction (&A.get_LR, &B.get_LR));
}

CExact_sequence

Exact_sequence(CHomology *A, CHomology *B, CHomology *AinterB, CHomology *AunionB){
if (AinterB == null) error
AunionB = CHomology(A,B);
CH1 = A;
CH2 = B;
create_morphism_i();
create_morphism_j();

```
create_morphism_rho();
create_morphism_nu();
}
```

GLOBAL FUNCTIONS

CHomology union = compute_homology_info_of_union (CHomology *A, CHomology *B, CHomology *AinterB){

```
CHomology AunionB = null;
CExact_sequence MV ( A, B, AinterB, AunionB); /* initialise cells of AunionB */


        CReduction_equivalence aux_equiv( &A.get_reductions , &B.get_reductions ); /*
creates A + B */
AunionB.set_reductions( ( *(AinterB.reductions), aux_equiv, MV.i) ); /* MV.i is the morphism
inclusion */

        CReduction reduct_lemma82 (MV);
CReduction reduct_Smith ( equiv_cone.get_RR.get_small_chain);
AunionB.get_reductions.get_LR.composition_of_reductions ( reduct_lemma82);
AunionB.get_reductions.get_RR.composition_of_reductions ( reduct_Smith);

        return AunionB;
}
```

GLOBAL ALGORITHM

```
   Input :
- list of Simplicial Complexes A1, A2,...
 /* Algorithm hypothesis: we know the homology of each input. For us, to "know the homol-
ogy" is equivalent of having a reduction equivalence were the left_reduction is of type Smith,
and the right_reduction is of type Lemma 82 or trivial */
/* Compute the homology of the parts */
for each CHomology X do{
CHomology HX(X);
HX.init_reductions(); /* procedure in CHomology class */
}
 /* At this point the algorithm fulfills the initial hypothesis */

 A1 = first CHomology of the list;

first CHomology of the list is treated;
```

```
 for each pair Ai, Aj of not treated CHomology of the list{


            CHomology Aij = compute_homology_info_of_union ( CHomology Ai, CHomology
Aj, CHomology Ai_inter_Aj )
Aij.compute_generators();
draw ( Aij.hom ); /* draw generators of homology*/
Ai, Aj are treated;
add Aij to the list;
}
```

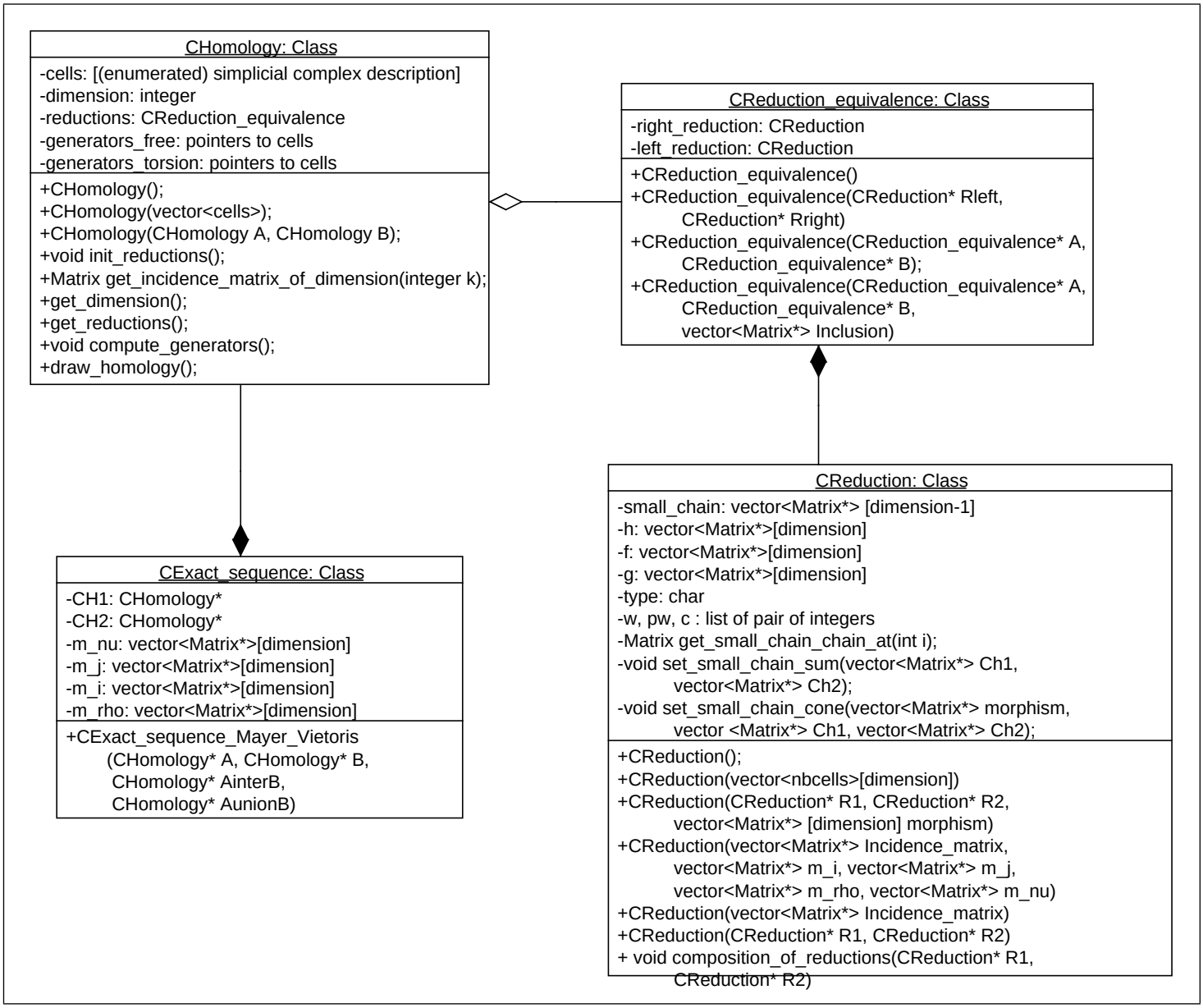


Figure C.1: Diagram of the data structure

Appendix D

Handwritten example of One step algorithm

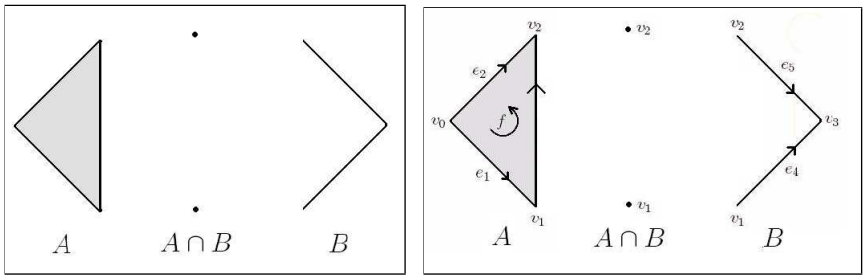


Table D.1: Subcomplex description

Input: chain complexes:

- Information on A_*: $(P_0^A, P_1^A, P_2^A, (P_0^A)^{-1}, (P_1^A)^{-1}, (P_2^A)^{-1}, N_1^A, N_2^A)$ (this information was obtained in example 14, 109).

Change of base of A

$$P_0^A = \begin{array}{c} \\ v_0 \\ v_1 \\ v_2 \end{array} \begin{array}{c} \begin{array}{ccc} x_0 & x_1 & x_2 \end{array} \\ \begin{pmatrix} -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & 0 \end{pmatrix} \end{array} \quad P_1^A = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \end{array} \begin{array}{c} \begin{array}{ccc} \alpha_1 & \alpha_2 & \alpha_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 1 & -1 & 1 \end{pmatrix} \end{array} \quad P_2^A = \begin{array}{c} f \\ \begin{pmatrix} 1 \end{pmatrix} \end{array}$$

The inverses of A

$$(P_0^A)^{-1} = \begin{array}{c} \\ x_0 \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{ccc} v_0 & v_1 & v_2 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \end{array} \quad (P_1^A)^{-1} = \begin{array}{c} \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{array} \begin{array}{c} \begin{array}{ccc} e_1 & e_2 & e_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \end{array} \quad (P_2^A)^{-1} = \begin{array}{c} f \\ \begin{pmatrix} 1 \end{pmatrix} \end{array}$$

Smith of A

$$N_1^A = \begin{array}{c} \\ x_0 \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{ccc} \alpha_1 & \alpha_2 & \alpha_3 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \end{array} \quad N_2^A = \begin{array}{c} \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{array} \begin{array}{c} \begin{array}{c} f \end{array} \\ \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \end{array}$$

- Information on B_*: $(P_0^B, P_1^B, (P_0^B)^{-1}, (P_1^B)^{-1}, N_1^B)$

  Change of base of B

$$P_0^B = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{ccc} \gamma_1 & \gamma_2 & \gamma_3 \end{array} \\ \begin{pmatrix} -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & 0 \end{pmatrix} \end{array} \quad P_1^B = \begin{array}{c} \\ e_4 \\ e_5 \end{array} \begin{array}{c} \begin{array}{cc} \beta_1 & \beta_2 \end{array} \\ \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} \end{array}$$

  The inverses of change of base of B

$$(P_0^B)^{-1} = \begin{array}{c} \\ \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \end{array} \quad (P_1^B)^{-1} = \begin{array}{c} \\ \beta_4 \\ \beta_5 \end{array} \begin{array}{c} \begin{array}{cc} e_1 & e_2 \end{array} \\ \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \end{array}$$

  Smith of B

$$N_1^B = \begin{array}{c} \\ \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{array} \begin{array}{c} \begin{array}{cc} \beta_1 & \beta_2 \end{array} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \end{array}$$

- Information on $(A \cap B)_*$: $(P_0^{(A \cap B)})$

$$P_0^{(A \cap B)} = \begin{array}{c} \\ v_1 \\ v_2 \end{array} \begin{array}{c} \begin{array}{cc} v_1 & v_2 \end{array} \\ \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{array}$$

  (There are no change of base because there are no border operators; we are working only with vertices).

Example 16. Homological Smith Reduction

1. Homological Smith Reductions of A, B and $A \cap B$

$$\begin{array}{ccc} (A\cap B)_* & A_* & B_* \\ {\scriptstyle g_{A\cap B}}\Uparrow\Downarrow{\scriptstyle f_{A\cap B}} & {\scriptstyle g_A}\Uparrow\Downarrow{\scriptstyle f_A} & {\scriptstyle g_B}\Uparrow\Downarrow{\scriptstyle f_B} \\ E(A\cap B)_* & EA_* & EB_* \end{array}$$

   (a) Classification of base in 'pre-boundaries' $\boldsymbol{pb}^k$, 'boundaries' $\boldsymbol{b}^k$, 'pre-weakboundaries' $\boldsymbol{pw}^k$, 'weak-boundaries' $\boldsymbol{w}^k$, 'homological cycles' $\boldsymbol{c}^k$ for each k.

   (b) Extract submatrix EN_k from N_k that is expressed in base $\{\boldsymbol{w}^{k-1}, \boldsymbol{c}^{k-1}, \boldsymbol{pw}^{k-1}\}$ and $\{\boldsymbol{w}^k, \boldsymbol{c}^k, \boldsymbol{pw}^k\}$

Output: base classification for each chain complex and the reduced incidence matrices EN_*^A, EN_*^B, $EN_*^{A\cap B}$

   (a) Classification of base of A_*

	pb	***b***	***pw***	***w***	***c***
$A_0 = [x_0, x_1, x_2]$		x_0, x_1			x_2
$A_1 = [\alpha_1, \alpha_2, \alpha_3]$	α_2, α_3	α_1			
$A_2 = [f]$	f				

   (b) Classification of base of B_*

	pb	***b***	***pw***	***w***	***c***
$B_0 = [\gamma_1, \gamma_2, \gamma_3]$		γ_1, γ_2			γ_3
$B_1 = [\beta_1, \beta_2, \alpha_3]$	β_1, β_2				

   (c) Classification of base of $(A \cap B)_*$

	pb	***b***	***pw***	***w***	***c***
$(A\cap B)_0 = [v_1, v_2]$					v_1, v_2

The reduction of the matrices are (with *null* we indicate the trivial morphism, i.e, 'the matrix is empty'):

   (a) $EN_1^A = null$, $EN_2^A = null$

   (b) $EN_1^B = null$

   (c) $EN_1^{(A\cap B)} = null$

Example 17. Cone Reduction Theorem

2. Consider the inclusion morphism $i : (A \cap B)_* \longrightarrow A_* \oplus B_*$ and construct the reduction of the cone.

$$\begin{array}{ccc} (A\cap B)_* & \xrightarrow{\ i\ } & A_*\oplus B_* \\ {\scriptstyle g_{A\cap B}}\uparrow\downarrow{\scriptstyle f_{A\cap B}} & & {\scriptstyle g_{A\oplus B}}\uparrow\downarrow{\scriptstyle f_{A\oplus B}} \\ E(A\cap B)_* & \xrightarrow{Ei} & EA_*\oplus EB_* \end{array} \quad \Longrightarrow \quad \begin{array}{c} Cone(i)_* \\ {\scriptstyle g_C}\uparrow\downarrow{\scriptstyle f_C} \\ Cone(Ei))_* = U_* \end{array}$$

$Ei = f_{A\oplus B}\ i\ g_{A\cap B}$

(a) Construct the incidence matrix $Cone(Ei) = U_*$. It is a matrix by blocks, 3×3.

$$D_1^U = \begin{bmatrix} EN_1^A & 0 & i_0^A \\ 0 & EN_1^B & i_0^B \\ 0 & 0 & -EN_0^{A\cap B} \end{bmatrix}$$

We compute the value of i_0^A and i_0^B (since $EN_1^A = EN_1^B = EN^{A\cap B} = null$):

$$i_0^A = \underbrace{(r_0^A)(P_0^A)^{-1}}_{(E.1)}\underbrace{I_0^A}_{(E.2)}\underbrace{P_0^{A\cap B}(r_0^{A\cap B})^T}_{(E.3)} = x_2 \begin{array}{c} \begin{array}{cc} v_1 & v_2 \end{array} \\ \begin{pmatrix} 1 & 1 \end{pmatrix} \end{array}$$

(D.1) $$(r_0^A)(P_0^A)^{-1} = x_2 \begin{array}{c} \begin{array}{ccc} v_0 & v_1 & v_2 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \end{array}$$

(D.2) $$I_0^A = \begin{array}{c} \\ v_0 \\ v_1 \\ v_2 \end{array} \begin{array}{c} \begin{array}{cc} v_1 & v_2 \end{array} \\ \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \end{array}$$

(D.3) $$P_0^{A\cap B}(r_0^{A\cap B})^T = P_0^{A\cap B}$$

We can also compute this matrix i_0^A by computing an image of the base $[v_1, v_2]$.
For example, for i_0^B: $v_1 \longmapsto (r_0^B)(P_0^B)^{-1}I_0^B P_0^{A\cap B}(r^{A\cap B})^T(v_1) = (r_0^B)(P_0^B)^{-1}I_0^B P_0^{A\cap B}(v_1) = (r_0^B)(P_0^B)^{-1}I_0^B(v_1) = (r_0^B)(P_0^B)^{-1}(v_1) = (r_0^B)(\gamma_3) = \gamma_3$

$v_2 \longmapsto (r_0^B)(P_0^B)^{-1}I_0^B P_0^{A\cap B}(r^{A\cap B})^T(v_2) = (r_0^B)(P_0^B)^{-1}I_0^B P_0^{A\cap B}(v_2) = (r_0^B)(P_0^B)^{-1}I_0^B(v_2) = (r_0^B)(P_0^B)^{-1}(v_2) = (r_0^B)(\gamma_1 + \gamma_3) = \gamma_3$

Therefore,

$$I_0^B = \gamma_3 \begin{matrix} \\ \end{matrix} \begin{matrix} v_1 & v_2 \end{matrix} \begin{pmatrix} 1 & 1 \end{pmatrix}$$

.

Summing up,

$$D_1^U = \begin{pmatrix} I_0^A \\ I_0^B \end{pmatrix} = \begin{matrix} x_2 \\ \gamma_3 \end{matrix} \begin{matrix} v_1 & v_2 \end{matrix} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$D_k^U = null$ for $k > 1$.

3. Compute the homology of $Cone(i)$.

   (a) Compute the homology (Smith Reduction) of $U_* = Cone(E(i))$.

$$\text{input: } D_1^U = \begin{matrix} x_2 \\ \gamma_3 \end{matrix} \begin{matrix} v_1 & v_2 \end{matrix} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \quad \text{output: } N_1^U = \begin{matrix} s_1 \\ s_2 \end{matrix} \begin{matrix} l_1 & l_2 \end{matrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

$$P_0^U = \begin{matrix} x_2 \\ \gamma_3 \end{matrix} \begin{matrix} s_1 & s_2 \end{matrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \quad P_1^U = \begin{matrix} v_1 \\ v_2 \end{matrix} \begin{matrix} l_1 & l_2 \end{matrix} \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}$$

$$(P_0^U)^{-1} = \begin{matrix} s_1 \\ s_2 \end{matrix} \begin{matrix} x_2 & \gamma_3 \end{matrix} \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} \quad (P_1^U)^{-1} = \begin{matrix} l_1 \\ l_2 \end{matrix} \begin{matrix} v_1 & v_2 \end{matrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

Determine the generators of the homology, $\boldsymbol{w}^U, \boldsymbol{c}^U$.

Classification of the base of U_*

	pb	***b***	***pw***	***w***	***c***
$U_0 = [s_1, s_2]$		s_1			s_2
$U_1 = [l_1, l_2]$	l_2				l_1

The generators of the homology are $H_0^U = [s_2]$ and $H_1^U = [l_1]$.

   (b) Express the homology generators s_2, l_1 in the canonical base of $U_* = Cone(Ei)$
   The canonical base of U_* is:

$$\{(\boldsymbol{w}^{A_k}, \boldsymbol{c}^{A_k}, \boldsymbol{pw}^{A_k}), (\boldsymbol{w}^{B_k}, \boldsymbol{c}^{B_k}, \boldsymbol{pw}^{B_k}), (\boldsymbol{w}^{(A\cap B)_{k-1}}, \boldsymbol{c}^{(A\cap B)_{k-1}}, \boldsymbol{pw}^{(A\cap B)_{k-1}})\}$$

Therefore, if $z \in U_*$, $z = z_k^A + z_k^B + z_{k-1}^{A\cap B}$. $H_0^U = [s_2] \Longrightarrow s_2 \rightarrow z^0 = P_0^U(s_2) = 0 + \gamma_3 + 0$ $H_1^U = [l_1] \Longrightarrow l_1 \rightarrow z^1 = P_1^U(l_1) = 0 + 0 + (v_1 - v_2)$

With these operations we have 'gone up' on the reduction:

$$\begin{array}{c} Cone(Ei)_* \\ {\scriptstyle g'}\Big\uparrow\Big\downarrow{\scriptstyle f'} \\ E(Cone(Ei))_* \end{array}$$

(c) We 'go up' on the reduction:

$$\begin{array}{c} Cone(i)_* \\ {\scriptstyle g_C}\Big\uparrow\Big\downarrow{\scriptstyle f_C} \\ Cone(Ei))_* \end{array}$$

to find the homology of the $Cone(i)$. If z is an homological generator of $Cone(Ei)$ then $y = g_C(z)$ is an homological generator of $Cone(i)$ where:

$$g_C = \begin{array}{c} \\ y_k^A \\ y_k^B \\ y_k^{A\cap B} \end{array} \begin{array}{c} \begin{array}{ccc} z_k^A & z_k^B & z_{k-1}^{A\cap B} \end{array} \\ \begin{pmatrix} P_k^A(r_k^A)^T & 0 & -h_k^A I_{k-1}^A \\ 0 & P_k^B(r_k^B)^T & -h_k^B I_{k-1}^B P_{k-1}^{(A\cap B)}(r_{k-1}^{(A\cap B)})^T \\ 0 & 0 & P_{k-1}^{(A\cap B)}(r_{k-1}^{(A\cap B)})^T \end{pmatrix} \end{array}$$

We are only interested in the values y_k^A and y_k^B. Compute:
$y1_k^A = P_k^A(r_k^A)^T z_k^A$
$y2_k^A = -h_k^A I_{k-1}^A P_{k-1}^{(A\cap B)}(r_{k-1}^{(A\cap B)})^T z^{(A\cap B)_{k-1}}$
$y1_k^B = P_k^B(r_k^B)^T z_k^B$
$y2_k^B = -h_k^B I_{k-1}^B P_{k-1}^{(A\cap B)}(r_{k-1}^{(A\cap B)})^T z^{(A\cap B)_{k-1}}$
Then $y_k^A = y1_k^A + y2_k^A$ and $y_k^B = y1_k^B + y2_k^B$.

- for $k = 0$
  $y1_0^A = 0$
  $y1_0^B = \gamma_3 \rightarrow P_0^B(r_0^B)^T(\gamma_3) = v_1$
  $I_{-1}^A P_{-1}^{(A\cap B)}(r_{-1}^{(A\cap B)})^T = null \Longrightarrow y2_0^A = 0$
  $y2_0^B = 0$

- for $k=1$
  $y1_1^A = 0$
  $y1_1^B = 0$
  $y2_1^A = -h_1^A I_0^A P_0^{(A\cap B)}(r_0^{(A\cap B)})^T(v1-v2) = h_1^A(v1-v2)$
  $y2_1^B = -h_1^B(v1-v2)$

  Lets compute $h_1^A : A_0 \to A_1$:

$$h_1^A = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} \alpha_1 & \alpha_2 & \alpha_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 1 & -1 & 1 \end{pmatrix} \end{array} \begin{array}{c} \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} x_0 & x_1 & x_2 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \end{array} \begin{array}{c} \\ x_0 \\ x_1 \\ x_2 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} v_0 & v_1 & v_2 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \end{array} = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} v_0 & v_1 & v_2 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & -1 & 0 \end{pmatrix} \end{array}$$

  Same with h_1^B:

$$h_1^B = \begin{array}{c} \\ e_4 \\ e_5 \end{array}\!\!\begin{array}{c} \begin{array}{ccc} v_0 & v_1 & v_2 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 0 & -1 & 0 \end{pmatrix} \end{array}$$

  Therefore,

$$y2_1^A = -h_1^A(v1-v2) = e_3$$
$$y2_1^B = -h_1^B(v1-v2) = e_4 - e_5$$

In conclusion, we have the homology $Cone(i)$, $H_0(Cone(i)) = [(y_0^A, y_0^B, y_0^{(A\cap B}{}^{)})] = [(0, v_1, y^{A\cap B)_0}]$ and $H_1(Cone(i)) = [(y_1^A, y_1^B, y_1^{A\cap B})] = [(e_3, e_4 - e_5, y^{A\cap B)_1}]$.

(d) Let us consider the reduction of Lemma 82:

$$\begin{array}{c} Cone(i)_* \\ \Uparrow lg \quad \Downarrow lf \\ (A\cup B)_* \end{array}$$

lf is defined as $lf(y^A, y^B, y^(A\cap B)) = y^A - y^B$. Compute:

$$H_0^{A\cup B} = [y_0^A - y_0^B] = [v_1]$$
$$H_1^{A\cup B} = [y_1^A - y_1^B] = [e_3 + e_5 - e_4]$$

Bibliography

[AGH+09] Dominique Attali, Marc Glisse, Samuel Hornus, Francis Lazarus, and Dmitriy Morozov. Persistence-sensitive simplification of functions on surfaces in linear time. In *SODA '09: Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1011–1020, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

[Ago76] Max K. Agoston. *Algebraic Topology. A First Course.* Pure and Applied Mathematics. A program of monographs, textbooks, and lecture notes, 1976.

[CdVL05] É. Colin de Verdière and Francis Lazarus. Optimal system of loops on an orientable surface. *Discrete & Computational Geometry*, 33:507–534, 2005.

[CSEH07] David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. *Discrete Comput. Geom.*, 37(1):103-120, 2007.

[CSEHM09] David Cohen-Steiner, Herbert Edelsbrunner, John Harer, and Dmitriy Morozov. Persistent homology for kernels, images, and cokernels. In *SODA '09: Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1011–1020, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

[DE93] Cecil Jose A. Delfinado and Herbert Edelsbrunner. An incremental algorithm for betti numbers of simplicial complexes. In *SCG '93: Proceedings of the ninth annual symposium on Computational geometry*, pages 232–239, New York, NY, USA, 1993. ACM.

[DFPH09] Leila De Floriani, Daniele Panozzo, and Annie Hui. A set of tools for representing, decomposing and visualizing non-manifold cellular complexes, 2009.

[DHSW03] J.-G. Dumas, F. Heckenbach, B. D. Saunders, and V. Welker. Computing simplicial homology based on efficient smith normal form algorithms. *Algebra, Geometry, and Software Systems*, pages 177-207, 2003.

[DLSCS08] Tamal K. Dey, Kuiyu Li, Jian Sun, and David Cohen-Steiner. Computing geometry-aware handle and tunnel loops in 3d models. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, pages 1–9, New York, NY, USA, 2008. ACM.

[EH10] Herbert Edelsbrunner and John Harer. *Computational Topology. An Introduction.* Amer. Math. Soc., Providence, Rhode Island, 2010.

[ELZ00] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. In *FOCS '00: Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, page 454, Washington, DC, USA, 2000. IEEE Computer Society.

[EW05] Jeff Erickson and Kim Whittlesey. Greedy optimal homotopy and homology generators. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1038–1046, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.

[FLM09] Barbara Fabio, Claudia Landi, and Filippo Medri. Recognition of occluded shapes using size functions. In *ICIAP '09: Proceedings of the 15th International Conference on Image Analysis and Processing*, pages 642–651, Berlin, Heidelberg, 2009. Springer-Verlag.

[Gie96] M. Giesbrecht. Probabilistic computation of the smith normal form of a sparse integer matrix. *Lecture Notes Computational Sciences*, 1122:173-186, 1996.

[GW01] Igor Guskov and Zoë J. Wood. Topological noise removal. In *GRIN'01: No description on Graphics interface 2001*, pages 19–26, Toronto, Ont., Canada, Canada, 2001. Canadian Information Processing Society.

[Hat02] Allen Hatcher. *Algebraic Topology.* Cambridge University Press, 2002. http://www.math.cornell.edu/ hatcher/AT/ATpage.html.

[HM91] James L. Hafner and Kevin S. McCurley. Asymptotically fast triangularization of matrices over rings. *SIAM J. Comput.*, 20(6):1068–1083, 1991.

[KA79] R. Kannan and Bachem A. Polinomial algorithms for computing the smith and hermite normal forms of an integer matrix. *SIAM Journal of Computing*, 8:499-507, 1979.

[KMS98] T Kaczynski, M Mrozek, and M Slusarek. Homology computation by reduction of chain complexes. 1998.

[MPZ08] Marian Mrozek, PawełPilarczyk, and Natalia Żelazna. Homology algorithm based on acyclic subspace. *Comput. Math. Appl.*, 55(11):2395–2412, 2008.

[Mun99] James Munkres. *Algebraic Topology.* Prentice Hall, 1999.

[Pel06] Samuel Peltier. *These: Calcul des groupes d'homologie sur des structures simpliciales, simploidales et cellulaires.* PhD thesis, 2006.

[PIK+09] Samuel Peltier, Adrian Ion, Walter G. Kropatsch, Guillaume Damiand, and Yll Haxhimusa. Directly computing the generators of image homology using graph pyramids. *Image Vision Comput.*, 27(7):846–853, 2009.

[Ser94] Francis Sergeraert. The computability problem in algebraic topology. 104:139–155, 1994.

[SR06] Francis Sergeraert and Julio Rubio. Constructive homological algebra and applications, 2006.

[Sto96] Arne Storjohann. Near optimal algorithms for computing smith normal forms of integer matrices. In *ISSAC '96: Proceedings of the 1996 international symposium on Symbolic and algebraic computation*, pages 267-274, New York, NY, USA, 1996. ACM.

[ZC05] A. Zomorodian and G Carlsson. Computing persistent homology. *Discrete & Computational Geometry*, 33:249-274, 2005.

[ZC08] Afra Zomorodian and Gunnar Carlsson. Localized homology. *Comput. Geom. Theory Appl.*, 41(3):126-148, 2008.

[Zom01] Afra Zomorodian. *Computing and Comprehending Topology: Persistence and Hierarchical Morse Complexes.* PhD thesis, Stanford University, 2001.



Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399