Jagiellonian University

Faculty of Mathematics and Computer Science

Institute of Computer Science and Computer Mathematics

A dissertation submitted for the degree
Doctor of Philosophy in Mathematical Sciences
Discipline Computer Science

Grzegorz Jabłoński

Persistent homology of a self-map

Supervisor:
prof. dr hab. Marian Mrozek

Kraków, 2015

Acknowledgments

First and foremost I would like to thank my supervisor, Professor Marian Mrozek, for all his invaluable advices, discussions and effort he put into leading me to my Ph.D. dissertation. He has been motivating, encouraging, enlightening and helpful.

I also thank Professor Herbert Edelsbrunner, who worked on a project that led to my dissertation. I appreciate all his contribution of ideas and time.

The members of KMO group have contributed immensely to my time at UJ. I would like to thank Piotr Brendel, Paweł Dłotko, Marc Ethier, Mateusz Juda, Hubert Wagner and Frank Weilandt. The group has been a source of friendships and good advice.

Hubert Wagner and Paweł Dłotko wrote parts of the source code used in a complex construction.

I am extremely grateful to my Family, especially my Mom and Dad for their constant support thorough all the way of my education.

And most of all I would like to thank my loving, supportive, encouraging, and patient wife Patrycja whose support during stages of this Ph.D. is so appreciated. Thank you.

This research was supported in part by National Science Centre Poland grant no. DEC-2013/09/N/ST6/02995, by the TOPOSYS project FP7-ICT-318493-STREP and by the SSDNM joint PhD programme co-financed by the European Social Fund through the Operational Programme Human Capital.

Podziękowania

Chciałbym serdecznie podziękować mojemu opiekunowi, Profesorowi Marianowi Mrozkowi, za wszystkie cenne rady, wskazówki i dyskusje oraz wysiłek włożony w poprowadzenie mnie w tej pracy. Zawsze zachęcał do wysiłku i pomagał, gdy było to potrzebne.

Chciałbym także podziękować Profesorowi Herbertowi Edelsbrunnerowi, z którym współpracowałem w projekcie będącym podstawą tej pracy. Chciałbym docenić jego pomysły i spędzony czas na dyskusjach.

Wszyscy członkowie KMO wspierali mnie podczas pracy w Uniwersytecie Jagiellońskim. Chciałbym podziękować Piotrowi Brendelowi, Pawłowi Dłotko, Marc Ethier, Mateuszowi Judzie, Hubertowi Wagnerowi oraz Frankowi Weilandt.

Hubert Wagner oraz Paweł Dłotko dostarczyli mi część kodu źródłowego używanego w konstrukcji kompleksów.

Chciałbym wyrazić niezmierną wdzięczność dla mojej rodziny, szczególnie moim rodzicom, za ich nieustające wsparcie w trakcie trwania całej mojej edukacji.

Pragnę także podziękować mojej żonie Patrycji za miłość, cierpliwość oraz za wsparcie we wszsytkich etapach prowadzących do powstania tej pracy. Dziękuję.

Praca została współfinansowana ze środków Narodowego Centrum Nauki przyznanych na podstawie decyzji numer DEC-2013/09/N/ST6/02995, grantu TOPOSYS FP7-ICT-318493-STREP oraz z programu Środowiskowych Studiów Doktoranckich z Nauk Matematycznych finansowanego przez Unię Europejską w ramach Programu Operacyjnego Kapitał Ludzki.

Contents

Chapter 1

Motivation

1.1 Introduction

In the last twenty years we have observed a tremendous progress in the computational and applied algebraic topology. The construction of efficient algorithms for homology groups triggered several applications in image and data analysis [1], robotics [11], sensor networks [10], and many other areas. One of the core new ideas is the concept of persistent homology proposed by Edelsbrunner, Letscher and Zomorodian [16]. It has been invented to topologically process data with noise. The method consists in analysing the changes in homology of a a nested sequence of simplicial complexes.

Another area of recent successful application of computational topology is the study of dynamical systems and continuous maps by means of cubical homology [23, 20, 24]. The main goal of this thesis is to use the ideas of persistent homology in the study of sampled maps in a topological setting.

Usually in applications only a finite sample of the space and map is available. The goal is to reconstruct at least some topological information about the space and map from the finite sample. In this thesis we focus on the reconstruction of the map by extending the methods of persistent homology used in the reconstruction of space. By applying the homology functor to a sequence of growing complexes we get a sequence of nested linear spaces. The analysis of a sequence of vectors spaces is a problem considered in quiver theory [12], topological data analysis [5] and category theory [17, 4]. We show how to use this technique with Vietoris–Rips [27] complexes. This can be easily adjusted to Čech complexes and with a bit more effort to α-complexes [15].

We restrict our attention to the homology over a field $\mathbb{F}$, therefore the homology map induced by a continuous map is a linear map between vector spaces.

A linear map $\varphi : U \to V$ is characterized up to a conjugacy by its rank. In the case of an algebraically closed field $\mathbb{F}$ an endomorphism $\gamma : V \to V$ is characterized up to a conjugacy by its Jordan form. More precisely, two endomorphisms are conjugate if their Jordan forms are equal up to some reordering. This implies that they have the same eigenvalues and dimensions of spaces spanned by eigenvectors for respective values. We exploit this fact in order to distinguish between the endomorphisms.

There is no obvious way of deriving the eigenvalues and eigenvectors from a sampled map. In our approach we reconstruct them from a sequence of growing complexes by inferring the most important candidates for eigenvalues and eigenvectors.

1.1.1 Toy example

We motivate our work with the following toy example. We consider $\mathbb{S}^1$ as the unit circle in the complex plane $\mathbb{C}$, that is, we take $\mathbb{S}^1 := \{x \in \mathbb{C} \ : \ |x| = 1\}$. Let $f : \mathbb{S}^1 \to \mathbb{S}^1$ be a map given by $f(x) = x^2$ for $x \in \mathbb{S}^1$. The map f doubles the angle between the X-axis and a line segment with end points in the origin and the point x. The only fixed point of f is the point $(1, 0)$, because it lies on the X-axis, thus the angle is zero.

As a finite sample of $\mathbb{S}^1$ we take six equidistant points on $\mathbb{S}^1$: $\{x_0, \ldots, x_5\}$ where $x_i = e^{\frac{2\pi i}{6}}$. We denote this set by S. Then, our finite sampling of the map f is $g(x_i) = x_{2i \mod 6}$.

To reconstruct $\mathbb{S}^1$ we first build a sequence of complexes

$$K_1 \subset K_2 \subset \cdots \subset K_7 = K$$

on points S visible in the Fig. 1.1. The level

$$K_1 := \{\{x_0\}, \{x_1\}, \{x_2\}, \{x_3\}, \{x_4\}, \{x_5\}\}$$

consists only vertices of K. The second level is

$$K_2 := K_1 \cup \{\{x_0, x_1\}, \{x_1, x_2\}, \{x_2, x_3\}, \{x_3, x_4\}, \{x_4, x_5\}\}$$

In $K_3 := K_2 \cup \{\{x_0, x_5\}\}$ we add edge so that homology generator of the circle is visible. The fourth level is

$$K_4 := K_3 \cup \{\{x_0, x_1, x_2\}, \{x_2, x_3, x_4\}\}$$

and has the same homology as K_3 but has 2 additional triangles. The fifth and sixth levels are $K_5 := K_4 \cup \{\{x_0, x_4, x_5\}\}$ and $K_6 := K_5 \cup \{\{x_1, x_2, x_3\}\}$. They still have the homology of the circle. Finally the hole is closed in

$$K_7 := K_6 \cup \{\{x_0, x_2, x_4\}\}.$$

Notice that in nearly all levels we can find simplices that are mapped by the map κ induced by g to simplices that appear only in the following levels. For instance, consider level K_2: the edge $\{x_0, x_1\}$ is mapped to a non-existent in K_2 edge $\{x_0, x_2\}$. The expansion present in f causes that κ does not induce a simplicial map in any level of the filtration. Our solution is to consider the collection of simplices of K_i mapped by κ into K_i. We denote it by $\bar{K}_i$. This way we get a simplicial map defined on every level: $\kappa_i : \bar{K}_i \to K_i$. In Fig. 1.2 we show $\bar{K}_i$ for all levels of the filtration. Notice that in $\bar{K}_5$ and $\bar{K}_6$ we have the homology of the circle, and the generator $c := [x_0, x_1] + [x_1, x_2] + [x_2, x_3] + [x_3, x_4] + [x_4, x_5] - [x_0, x_5]$ is mapped to $2([x_0, x_2] + [x_2, x_4] - [x_0, x_4])$ under κ_5 and κ_6. The chain $[x_0, x_2] + [x_2, x_4] - [x_0, x_4]$ is homologous to c in K_5 and K_6. By abusing the notation[1] we write $\kappa_5(c) = 2 \cdot c$, and the same for κ_6. This simple example indicates that it should be possible to extract eigenvalue 2 of the map f working only with sampled data.

1.2 Outline of the thesis

In Chapter 2 we recall definitions from algebraic topology needed in this thesis. In Chapter 3 we present foundations of the persistent homology of a self map from the point of view of category theory. Chapter 4 describes the algorithm and software for the persistence of maps. In Chapter 5 we present the convergence and stability results for the algorithm. In Chapter 6 we study several examples. In Chapter 7 we show how to use software delivered with the thesis.

1.3 Author's contribution

In the thesis we present the theory, algorithms and numerical experiments concerning the recovery of homological information about a map from a finite sample. The author of the thesis actively participated in the whole project. The entirely own, independent contribution of the author encompasses Thm. 3.15 on the existence of matching bases together with its proof in Section 4.8, the theory concerning generalized eigenvectors (Sections 3.6.3 and 3.6.4), the construction and analysis of algorithms for maps (Sections 4.1 to 4.3, 4.5, 4.6.1 and 4.7 to 4.9), implementation of the algorithms (delivered with the thesis) and the numerical experiments (Chapter 6). Parts of

[1] $\kappa_5(c)$ is a chain in K_5, whereas c is a chain in $\bar{K}_5$. We show that this is not a problem if we take $2 \cdot \iota(c)$ instead of $2 \cdot c$, where ι is a natural inclusion between $\bar{K}_5$ and K_5

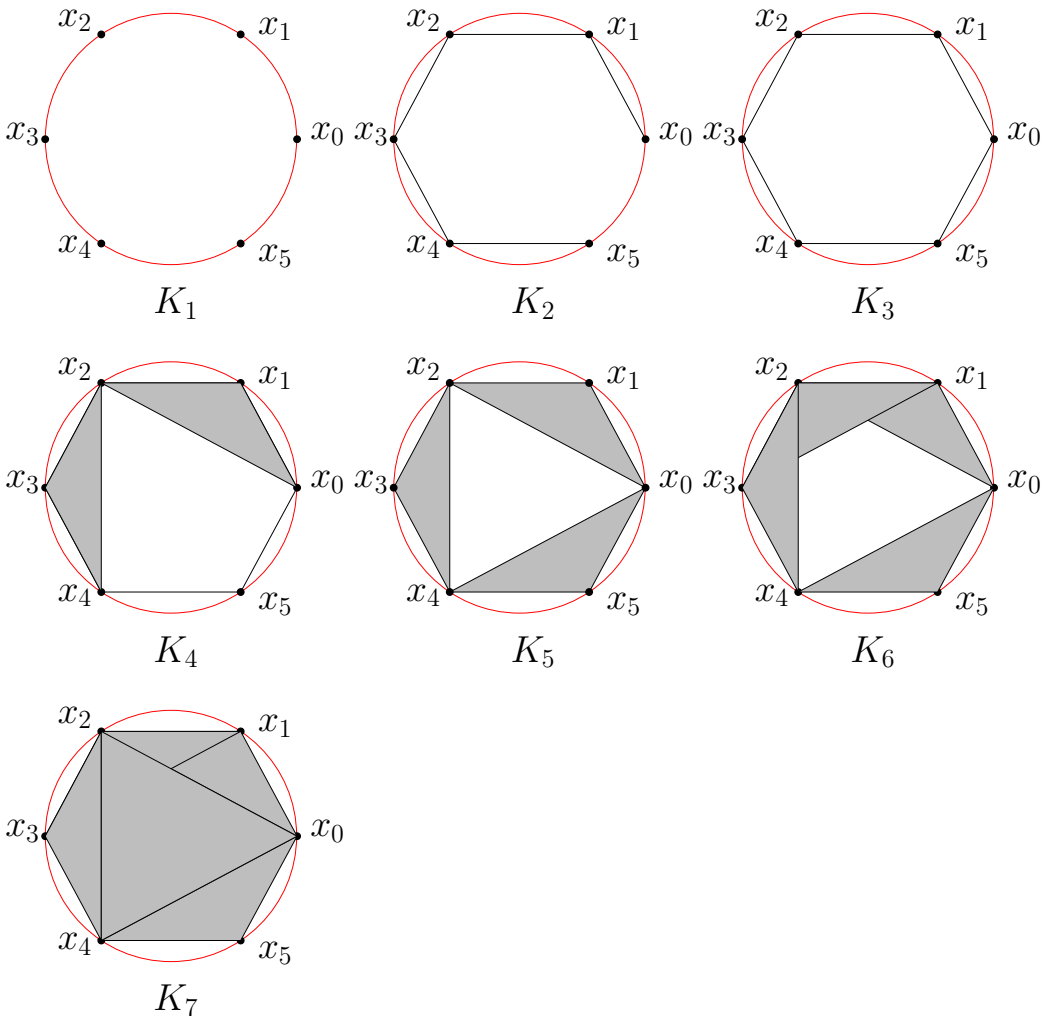


Figure 1.1: Seven levels of the filtration $\mathcal{K}$. The circle from which we sample points in S is shown in red.

this thesis were published in joint paper with M. Mrozek and H. Edelsbrunner [14].

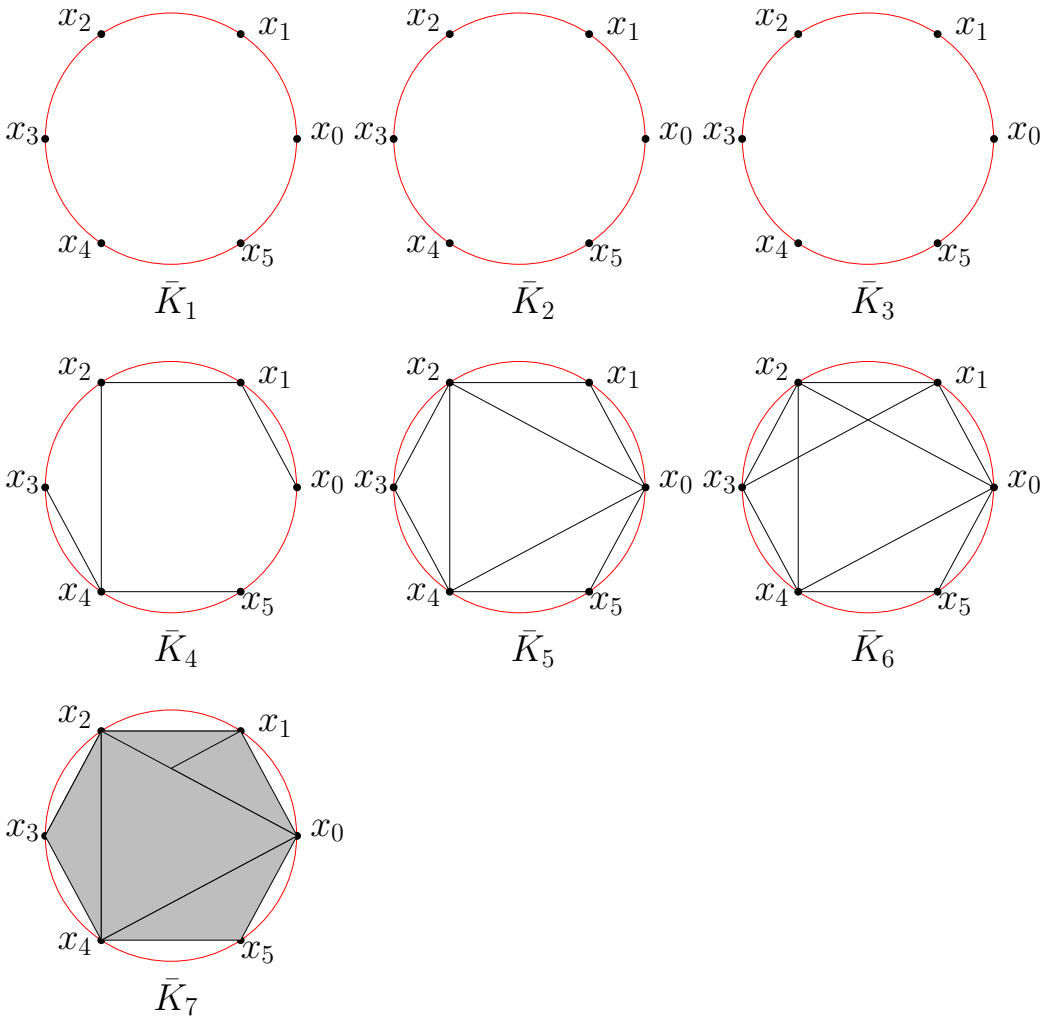


Figure 1.2: Maximal subcomplexes such that κ defines a simplicial map.

Chapter 2

Simplicial complexes and persistent homology

2.1 Introduction

Algebraic topology provides tools to translate topological problems into algebraic problems. This is useful, because there are many algorithmic tools to solve problems in algebra unlike the case of topology. For instance, there are algorithms to solve systems of equations, process matrices (for example diagonalization or multiplication methods) or solve eigenvalue problems.

In this chapter we recall basic concepts of algebraic topology relevant for this theses. We also describe the relatively new concept of persistent homology.

2.2 Simplicial complexes

Definition 2.1 (AS complex)**.** An *abstract simplicial complex* is a finite collection K of finite, non-empty sets, such that if $\sigma \in K$ and $\emptyset \neq \tau \subset \sigma$, then $\tau \in K$.

An element of K is called an *(abstract) simplex*. Every subset $\tau \subset \sigma$ is called a *face* of σ. The *dimension* of a simplex σ is the number of its elements minus 1

$$\dim(\sigma) := \operatorname{card} \sigma - 1.$$

The set $V(K) := \bigcup K$ is called the vertex set. Singletons constructed from elements of $V(K)$ are called 0-dimensional simplices. We will abuse notation and identify a 0-dimensional simplex $\{v\}$ with the vertex v. An n-dimensional simplex is briefly called n-simplex. A 1-simplex is called an edge, a 2-simplex

a triangle, a 3-simplex a tetrahedron. A *subcomplex* is a subset $L \subset K$ that is also an abstract simplicial complex. The *k-skeleton* of a complex K is the subcomplex S such that $\sigma \in S$ if and only if $\sigma \in K$ and $\dim(\sigma) \leq k$. A *proper face* τ of a simplex σ is a face of σ with $\dim(\tau) < \dim(\sigma)$. The *dimension of a complex* K is the maximal dimension of all simplices of K and is denoted by $\dim K$.

Example 2.2. An example of an abstract simplicial complex is the set

$$K = \{\{A, B, C\}, \{A, B\}, \{A, C\}, \{B, C\}, \{A\}, \{B\}, \{C\}\}$$

consisting of 3 vertices, 3 edges and one triangle. This is different from the complex

$$L = \{\{A, B\}, \{A, C\}, \{B, C\}, \{A\}, \{B\}, \{C\}\}$$

which does not contain the full triangle.

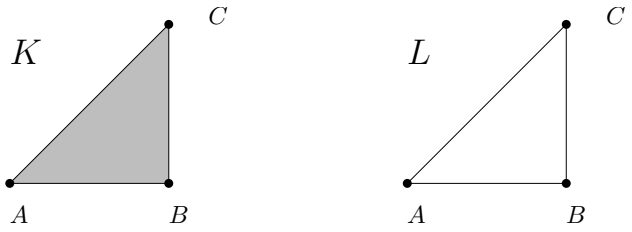


Figure 2.1: Graphical presentation of complexes K and L. Notice, the complex L is not filled, because it does not contain the 2-simplex $\{A, B, C\}$

The set $\{\{A, B\}, \{A\}\}$ is not a simplicial complex, because it lacks vertex B of the edge $\{A, B\}$.

2.2.1 Geometric simplicial complex

We intentionally work with abstract simplicial complexes, because this allows us to detach from the geometrical representation in the computer memory. However, to associate topology with an abstract simplicial complex we need the concept of a geometric simplicial complex. The *geometric n-simplex* on affinely independent set of points $\{v_0, \ldots, v_n\} \subset \mathbb{R}^m$ is the convex hull of $\{v_0, v_1, \ldots, v_n\}$, that is the smallest convex set in $\mathbb{R}^m$ containing the points $\{v_0, \ldots, v_n\}$. The elements of $\{v_0, \ldots, v_n\}$ are called the *vertices* of the simplex. A *face* of a geometric n-simplex σ is the convex hull of any non-empty subset of the vertices of σ. A *geometric simplicial complex* S is a finite set of geometric simplices such that:

i) every face of a simplex in S is in S,

ii) the intersection of two simplices in S is also a simplex in S.

Dimension of an n-simplex is n. A *subcomplex* of a simplicial complex S is any subset of S that is a simplicial complex.

Given a geometric simplicial complex S we define an abstract simplicial complex $\mathrm{cmplx}(S)$ by including in $\mathrm{cmplx}(S)$ the abstract simplex $\{v_1, \ldots, v_n\}$ for every n-simplex $\sigma \in S$ generated by points $\{v_1, \ldots, v_n\}$.

2.2.2 Vietoris–Rips complex

Definition 2.3. Let (M, δ) be a metric space with metric δ and let $r \in \mathbb{R}^+$. A *Vietoris–Rips complex* with threshold r and vertices in $S \subset M$ is an abstract simplicial complex whose simplices are finite subset of S with a diameter at most r. We denote this complex by $\mathrm{VR}_S(r)$.

Thus, a simplex $\sigma = \{v_0, v_1, \ldots, v_n\} \in \mathrm{VR}_S(r)$ if and only if the distance $\delta(v_i, v_j)$ between any two points v_i, v_j is not greater than r. For a given simplex σ we define its *weight* as the minimal r such that $\sigma \in \mathrm{VR}_S(r)$. The Vietoris–Rips complex containing simplices constructed from all subsets of S is called a *full Vietoris –Rips complex* and denoted $\mathrm{VR}_S(\infty)$. In the algorithms we only build the n-skeleton of the Vietoris–Rips complex, usually for $n = 2$, so we include only vertices, edges and triangles. We take as (M, δ) the Euclidean space $\mathbb{R}^l$ with the Euclidean metric in $\mathbb{R}^l$.

The *Vietoris–Rips filtration* is the filtration induced by an ordering of simplices in a full Vietoris–Rips complex according to their weight.

2.2.3 Čech complex

Definition 2.4. Let (M, δ) be a metric space with metric δ and let $r \in \mathbb{R}^+$. A *Čech complex* with threshold r and vertices in $S \subset M$ is an abstract simplicial complex whose simplices $\{v_1, \ldots, v_k\}$ are finite subsets of S such that the intersection of balls with diamater r centered at $\{v_1, \ldots, v_k\}$ is non-empty. We denote this complex by $\check{\mathrm{C}}_S(r)$.

An example of Čech complex is presented in Fig. 2.3. It is easy to show that $\check{\mathrm{C}}_S(r) \subset VR_S(r)$.

2.2.4 Simplicial map

A simplicial map f between abstract simplicial complexes K and L is a map $f : V(K) \to V(L)$ such that if $\sigma = \{v_1, \ldots, v_n\} \in K$ then $f(\sigma)$ is a simplex in L. We say that a simplicial map f is an *isomorphism* of K and L if f is a

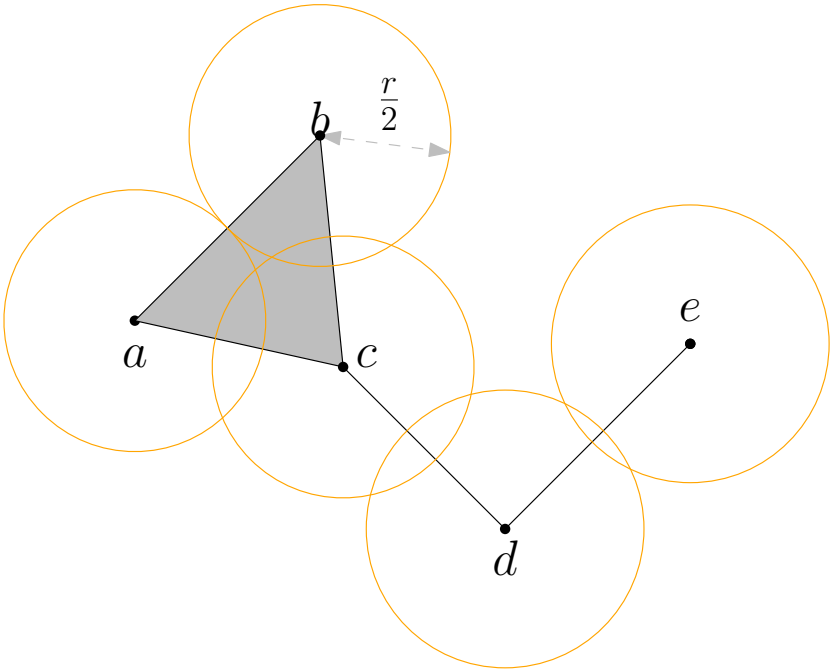


Figure 2.2: Let $A = \{a, b, c, d, e\}$ and r the parameter of Rips complex $\mathrm{VR}_X A(r)$. Then $\mathrm{VR}_A(r)$ consists of 5 vertices, 5 edges $\{a, b\}, \{b, c\}, \{c, d\}, \{d, e\}$ and the full triangle $\{a, b, c\}$.

bijection and both f and f^{-1} are simplicial maps. Complexes K and L are isomorphic if there exists an isomorphism between them.

We say that a geometric complex S is a *geometric realization* of an abstract simplicial complex K if $\mathrm{cmplx}\,(S)$ is isomorphic with K. There is no rule saying that an d-dimensional complex can be represented in the d-dimensional Euclidean space. Even worse, an d-dimensional abstract simplicial complex may have a geometric realization only in $\mathbb{R}^{2d+1}$[13][p. 53].

Given a simplicial map f we extend it to a continuous map $|f| : |K| \to |L|$ on geometrical representations $|K|$ and $|L|$ as follows. Let $x \in |K|$. Then

$$x = \sum_{v_i \in V(K)} t_i v_i,$$

for some $t_i \in \mathbb{R}$ such that

$$\sum_{i \in \{0,\ldots,n\}} t_i = 1$$

where $n = \mathrm{card}\, V(K)$, and we define

$$|f|(x) := \sum_{v_i \in V(K)} t_i f(v_i).$$

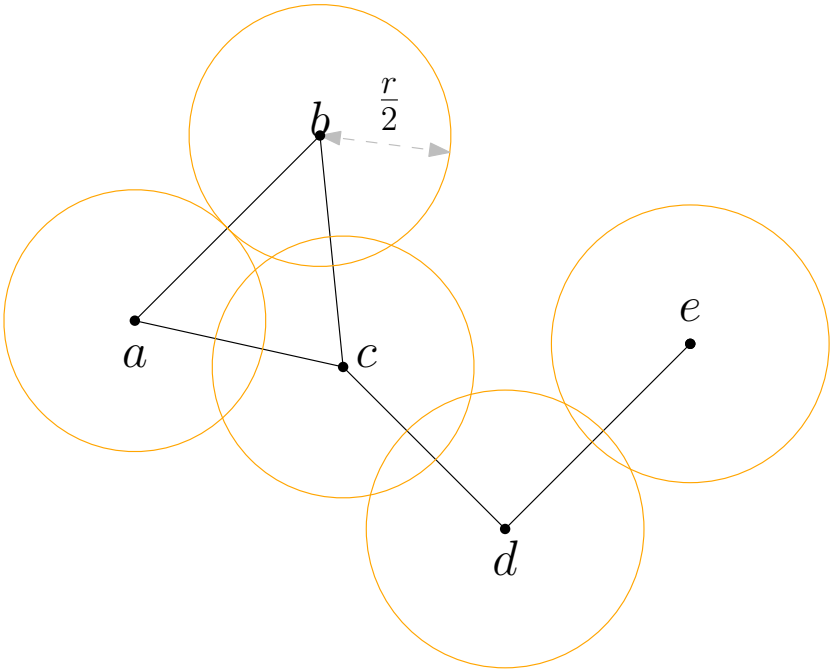


Figure 2.3: Let $A = \{a, b, c, d, e\}$ and let r be the parameter of Čech complex $\check{\mathrm{C}}_A(r)$ as in the figure. Observe that $\check{\mathrm{C}}_A(r)$ consists of 5 vertices and 5 edges $\{a, b\}, \{b, c\}, \{c, d\}, \{d, e\}$.

2.3 Boundary homomorphism

The main point of homology is to grasp the intuitive concept of holes in topological spaces, such as cycles, tunnels or voids. To do so we introduce basic concepts necessary to translate geometrical or combinatorial objects like abstract or simplicial complexes to algebraic structures.

Assume K is an abstract simplicial complex, and $s = \{v_0, \ldots, v_n\}$ is an n-dimensional simplex of K. An *orientation* of s is an ordering of its vertices up to an even permutation. An *oriented simplex*

$$\sigma := [v_0, v_1, \ldots, v_n]$$

is the simplex $s = \{v_0, v_1, \ldots, v_n\}$ with the orientation induced by the ordering $(v_0, v_1, \ldots, v_n)$. Hence, every simplex of dimension greater than zero has two possible orientations. We say that two oriented simplexes σ and σ' with the same vertices have reverse orientations if one belongs to the class of even permutations and the other to the class of odd permutations.

For every simplex $s \in K$ we arbitrarily fix one of its orientations and we treat s as an oriented simplex with this orientation. We denote σ with the

reversed orientation by $-\sigma$.

Let $\mathbb{F}$ be a field. By a *p-chain* of an abstract simplicial complex K with coefficients in $\mathbb{F}$ we mean a function $f : K_p \to \mathbb{F}$, such that for any $\sigma \in K$

$$f(-\sigma) = -f(\sigma).$$

The set of all p-chains together with argument-wise addition and multiplication by the elements of $\mathbb{F}$ is a vector space over $\mathbb{F}$, called the vector space of p-chains over K and denoted $C_p(K, \mathbb{F})$ or briefly $C_p(K)$. Elements of $C_p(K, \mathbb{F})$ may be written as finite sums $\sum_i \alpha_i \sigma_i$, where $\alpha_i \in \mathbb{F}$ and σ_i is identified with the map sending σ_i to 1 and any other simplex to 0. Thus, the set of p-dimensional simplices forms a basis of $C_p(K)$. We have 0-chains of vertices, 1-chains of edges and so on. Such a p-chain can be viewed as a collection of p-simplices with multiplicities.

Example 2.5. Let

$$L = \{\{A, B\}, \{A, C\}, \{B, C\}, \{A\}, \{B\}, \{C\}\}$$

be an abstract simplicial complex. Examples of 0-chains are $c_1 := [A] + [B]$ or $c_2 := [A] - [B]$. We can add them obtaining $c_1 + c_2 = 2[A]$.

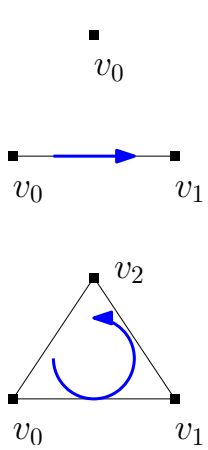
Let $\sigma_1, \sigma_2, \ldots, \sigma_n$ be the basis of $C_p(K)$ and let $c_1 = \sum_{i=1}^n \alpha_i \sigma_i$ and $c_2 = \sum_{i=1}^n \beta_i \sigma_i$ be chains in $C_p(K)$. The scalar product of the chains c_1, c_2 denoted as $\langle c_1, c_2 \rangle$ is:

$$\langle c_1, c_2 \rangle := \sum_{i=1}^{n} \alpha_i \beta_i.$$

Vector spaces of p-chains are connected by homorphisms $\partial_p : C_p \to C_{p-1}$, called *boundary operators* and defined on basis as follows: Let $\sigma = [v_0, v_1, ..., v_n]$ be an oriented n-simplex. We define its *boundary* as:

$$\partial\sigma = \sum_{i=0}^{n} (-1)^i [v_0, \ldots, \hat{v}_i, \ldots, v_n]$$

where $[v_0, \ldots, \hat{v}_i, \ldots, v_n]$ denotes the face of σ without vertex v_i. We skip the subscript p in ∂_p whenever p is deducible from the context. Intuitively the boundary operator on a p-simplex generates all faces of dimension $p-1$ with proper orientation. Sample boundaries are presented in Fig. 2.4.



$$\partial([v_0]) = 0$$

$$\partial([v_0, v_1]) = v_1 - v_0$$

$$\partial([v_0, v_1, v_2]) = [v_0, v_1] + [v_1, v_2] - [v_0, v_2]$$

Figure 2.4: Examples of simplices together with their boundaries.

The boundary operator satisfies the following equation:

$$\partial_n \circ \partial_{n+1} = 0 \tag{2.6}$$

which implies that

$$\mathrm{im}(\partial_{n+1}) \subseteq \ker(\partial_n). \tag{2.7}$$

The proof of (2.6) may be found for example in [18][Lemma 2.1;p. 105].

2.4 Chain complexes

A *chain complex* is a pair (C, ∂) consisting of vector spaces $C_0, C_1, \ldots$ and connecting homomorphisms $\partial_p : C_p \to C_{p-1}$ such that $\partial_p \partial_{p+1} = 0$ for any $p \geq 0$. Every abstract simplicial complex with the boundary operator is an example of a chain complex.

2.4.1 Chain maps

A *chain map* $\tau : C \to D$ between chain complexes (C, ∂^C) and (D, ∂^D) is a collection of homomorphisms: $\tau_p : C_p \to D_p$ s.t. $\tau_{p-1}\partial_p^C = \partial_p^D \tau_p$ for every p or equivalently the following diagram commutes:

$$\begin{array}{ccc} C_p & \xrightarrow{\partial_p^C} & C_{p-1} \\ \downarrow{\scriptstyle \tau_p} & & \downarrow{\scriptstyle \tau_{p-1}} \\ D_p & \xrightarrow{\partial_p^D} & D_{p-1} \end{array}.$$

2.4.2 Induced chain maps

Given a simplicial map $f : K \to L$ between abstract simplicial complexes K and L we define induced chain map between chain complexes $C_p(K)$ and $C_p(L)$ as follows:

$$f_p([v_0, v_1, \ldots, v_p]) := \begin{cases} [f(v_0), f(v_1), \ldots, f(v_p)], & \text{if } f(v_0), f(v_1), \ldots, f(v_p) \text{ are distinct,} \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

and extend f_p linearly to $C_p(K)$, thus obtaining $f_p : C_p(K) \to C_p(L)$.

Proposition 2.9. *The diagram*

$$\begin{array}{ccc} C_p(K) & \xrightarrow{\partial_p} & C_{p-1}(K) \\ \downarrow{\scriptstyle f_p} & & \downarrow{\scriptstyle f_{p-1}} \\ C_p(L) & \xrightarrow{\partial_p} & C_{p-1}(L) \end{array}$$

is commutative, hence series of homomorphisms f_p induces a chain map.

Proof of Prop. 2.9 may be found in [22][Proposition 3.33,p. 54].

2.4.3 Homology

To define homology we need two more definitions: $Z(C_p) := \ker(\partial_p)$ and $B(C_p) = \mathrm{im}(\partial_{p+1})$. Elements of $B(C_p)$ are called *boundaries*. Chains in $Z(C_p)$ are called *cycles*, because boundaries of its elements sum to zero. An example is the 1-cycle in Fig. 2.4 consisting of 3 elements $c := [v_0, v_1] + [v_1, v_2] - [v_0, v_2]$. Therefore, $\partial(c) = 0$, because each vertex appears twice in the boundary, with opposite coefficients.

Eqn. (2.7) allows us to define the *p-th homology group* as the quotient space:

$$H(C_p) := Z(C_p)/B(C_p). \tag{2.10}$$

The homology group $H(C_p)$ consists of classes of cycles in $Z(C_p)$, such that two cycles are in the same class if and only if their difference is the boundary of a $(p+1)$-chain.

Algebraically, we have a sequence of homomorphisms of vector spaces:

$$\ldots \xrightarrow{\partial_{n+2}} C_{n+1} \xrightarrow{\partial_{n+1}} C_n \xrightarrow{\partial_n} \ldots C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0. \tag{2.11}$$

Homology groups tell us how close we are to the case where $\operatorname{im}\partial_{n+1} = \ker\partial_n$. If we have $\operatorname{im}\partial_n = \ker\partial_{n+1}$ for every n then we say that the sequence (2.11) is *exact*.

The series of homology groups $H(C_0), H(C_1), \ldots$ is denoted by $H_*(C)$. We are mostly interested in homology groups generated by simplicial complexes. Let K be an n-dimensional abstract simplicial complex. To simplify notation we denote the homology group $H(C_p(K, \mathbb{F}))$ by $H_p(K)$. We write $H_*(K)$ for the homology groups $H_0(K), \ldots, H_n(K)$.

Example 2.12. Consider complexes K and L in Fig. 2.1. The homology groups $H_0(K)$ and $H_0(L)$ consist of one generator - the class of the connected component. The homology group $H_1(K)$ is trivial, whereas $H_1(L)$ is generated by the 1-cycle $[A, B] + [B, C] - [A, C]$, thus L has non-trivial homology in the first dimension.

Homology groups computed on chains with coefficients from a field are vector spaces, therefore the appropriate name should be "homology vector spaces". However we follow the common convention to use the name homology groups in all cases.

2.4.4 Maps induced in homology

If $\tau : C \to D$ is a chain map between chain complexes (C, ∂) and (D, ∂), then τ maps cycles and boundaries of C into cycles and boundaries of D respectively. Hence we can define the homomorphism $\tau_p^* : H(C_p) \to H(D_p)$ by setting:

$$\tau_p^*(c + B(C_p)) := \tau(c) + B(D_p)$$

where $c \in C_p$ is a p-chain. Combining the above statement and Prop. 2.9 we conclude:

Proposition 2.13. *Let K and L be abstract simplicial complexes, and let $f : K \to L$ be a simplicial map. Then for all $p \geq 0$, the map f induces homomorphisms $f_p^* : H_p(K) \to H_p(L)$.* □

We denote the homomorphism $f_p^* : H_p(K) \to H_p(L)$ with $H_p(f)$. For a more comprehensive introduction to homology we refer to [22].

2.5 Filtrations

A *filtration* $\mathcal{K}$ of an abstract simplicial complex K is a sequence of nested subcomplexes of K:

$$\emptyset = K_1 \subset K_2 \subset \cdots \subset K_n = K.$$

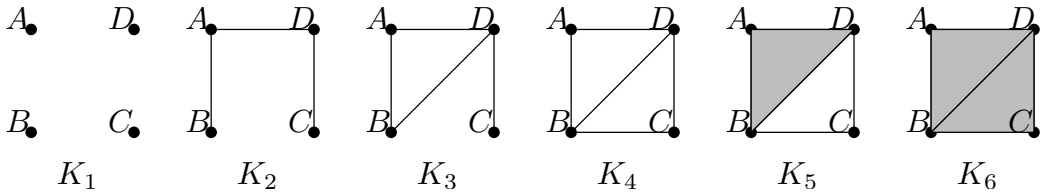


Figure 2.5: An example of a filtration $\mathcal{K}$ consisting of seven levels $K_1, \ldots, K_7$. For the sake of simplicity we omit the empty set at the beginning of the filtration.

In applications, a filtration often results from building the complex by adding simplexes step by step. A different way of constructing a filtration is to define the *weight* of a simplex $\sigma \in K$ as a function $w : K \to \mathbb{R}$ such that if τ is a face of σ then $w(\tau) \leq w(\sigma)$. A *sublevel set* of w for $a \in \mathbb{R}$ is the pre-image $w^{-1}((-\infty, a])$ of the interval $(-\infty, a]$.

Definition 2.14. Given a weight function $w : K \to \mathbb{R}$ on all simplices of K we define the subcomplex K_ε of K as the sublevel set $w^{-1}((-\infty, \varepsilon])$.

Proposition 2.15. *For $\varepsilon \in \mathbb{R}$ the complex K_ε is a subcomplex of K.*

Proof. If $\sigma \in K_\varepsilon$ then obviously $\sigma \in K$. To check that K_ε is an abstract simplicial complex, observe that for $\sigma \in K_\varepsilon$, and a face τ of σ we have $w(\tau) \leq w(\sigma) \leq \varepsilon$, hence $\tau \in K_\varepsilon$. □

Taking an increasing sequence of real numbers $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$ we obtain a filtration $K_{\varepsilon_1} \subset K_{\varepsilon_2} \subset \ldots K_{\varepsilon_n}$. A *critical value* of w is a value $\delta \in \mathbb{R}$ such that the homology of $K_{\delta-\epsilon}$ is not isomorphic with the homology of $K_{\delta+\epsilon}$ for arbitrarily small $\epsilon > 0$. A function w is *tame* if the number of critical values of w is finite.

Assume $w : K \to \mathbb{R}$ is a real valued tame function defined on an abstract simplicial complex K, and $a_1 < a_2 < \cdots < a_n$ are all critical values of w. Writing $K_i := w^{-1}((-\infty, a_i])$ we have a *filtration induced by the function w*:

$$\emptyset = K_1 \subset K_2 \subset \cdots \subset K_n = K \tag{2.16}$$

2.6 Persistent homology

Let $\mathcal{K}$ be a filtration $K_0 \subset K_1 \subset \cdots \subset K_n$ of an abstract simplicial complex K. We have a sequence of chain maps induced by inclusion maps:

$$\emptyset = C_p(K_1) \to C_p(K_i) \to \cdots \to C_p(K_n) \tag{2.17}$$

We are not interested in how K_i changes as i is increased, but how the homology groups of K_i differ with growing i. To accomplish the goal, we need one more notation: $f_p^{i,j} : H_p(K_i) \to H_p(K_j)$ is the homomorphism induced by the inclusion between two subcomplexes K_i and K_j such that $i < j$.

Definition 2.18. The (i, j) p-th persistent homology group is the image of the homomorphism $f_p^{i,j}$. We denote it by $H_p^{i,j}$.

There is also an equivalent description of the persistent homology groups. The persistent homology group $H_p^{i,j}$ consists of homology classes of K_i that are still visible in K_j. Let $Z_p(K_i) := Z(C_p(K_i))$ and $B_p(K_i) := B(C_p(K_i))$ we have:

$$H_p^{i,j} = Z_p(K_i)/(B_p(K_j) \cap Z_p(K_i)).$$

The *p-th persistent Betti number* is the rank of the p-th persistent homology group: $\beta_p^{i,j} := \operatorname{rank} H_p^{i,j}$. Informally, this is the number of classes that survive from K_i to K_j. We say that a homology class $[c]$ of a chain c is *born* at K_i if $[c] \in H_p^{i,j}$ and $[c] \notin H_p^{i-1,i}$. Hence, $H_p^{i,i}$ is the first time we observe $[c]$. The class $[c]$ *dies* entering K_j if $f_p^{i,j-1}([c]) \neq 0$ and $f_p^{i,j}([c]) = 0$. We then say that c has *persistence interval* equal to $[i, j]$.

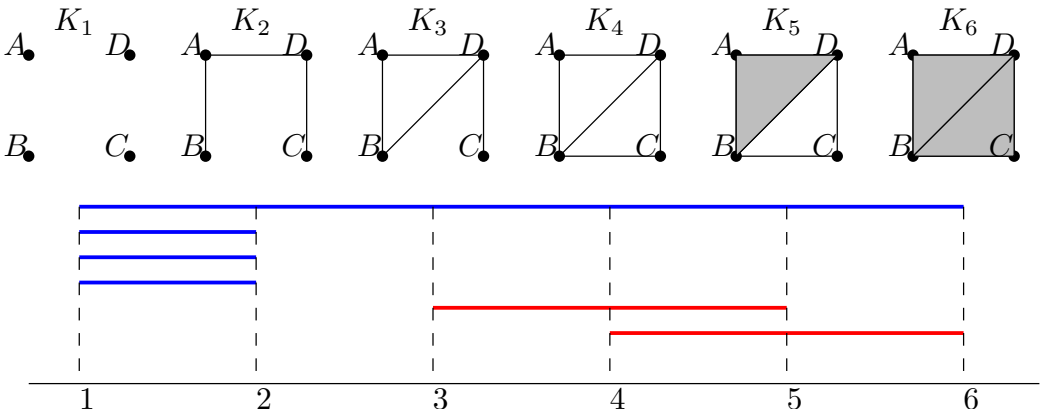


Figure 2.6: Filtration $\mathcal{K}$ with persistence intervals depicted as segments. The blue segments represent the persistence intervals in the 0th dimension i.e. connected components. One connected component "lives" for the whole filtration, three are killed in K_2, because it is connected. In the 1st dimension we have two persistence intervals, depicted in red: $[3, 5]$ generated by cycle $[A, B] + [B, D] - [A, D]$ and $[4, 6]$ generated by $[B, C] + [C, D] - [D, B]$. The cycle $[A, B] + [B, D] - [A, D]$ is killed by the simplex $\{A, B, D\}$.

Given persistent Betti numbers we can count linearly independent classes that are born at K_i and die entering K_j.

Definition 2.19 ([9])**.** The number of linearly independent classes that are born at K_i and die entering K_j is

$$\#_p^{i,j} := (\beta_p^{i,j-1} - \beta_p^{i,j}) - (\beta_p^{i-1,j-1} - \beta_p^{i-1,j}).$$

Note that $(\beta_p^{i,j-1} - \beta_p^{i,j})$ is the number of classes that are born not later than at level i and die entering j, whereas $(\beta_p^{i-1,j-1} - \beta_p^{i-1,j})$ counts classes born not later than at level $(i-1)$ and die entering j.

We say that the *multiplicity* of the interval $[i, j]$ is $\#_p^{i,j}$. Multiplicities are connected with multisets that are generalization of sets and allow multiple instances of its elements.

Let $\mathcal{J}$ be a subset of $\mathbb{N}$.

Definition 2.20. The persistence diagram of a filtration $\mathcal{K} = (K_i)_{i\in\mathcal{J}}$ is the multiset $\mathrm{Dgm}(\mathcal{K})$ of all its persistence intervals counted with multiplicities.

In the literature persistence diagrams are presented as plots of points in $\mathbb{R}^2$. An interval $[b, d]$ is depicted in the diagram as a point (b, d). An example of a persistence diagram is shown in Fig. 2.7. Points close to the diagonal represent homology classes existing for a short time in the filtration. Such points are likely to be the result of noise and are considered to be less important.

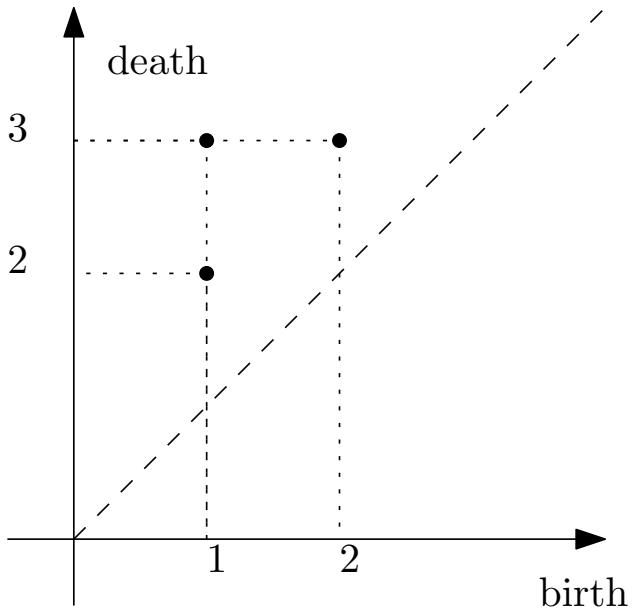


Figure 2.7: Plot of the persistence diagram containing intervals $[1, 2], [1, 3], [2, 3]$ all with multiplicity one.

If $\mathcal{K}$ is a filtration induced by a function $f : K \to \mathbb{R}$ then we write $\mathrm{Dgm}(f)$ for the respective persistence diagram. We will discuss the algorithm for computing the persistent homology groups in Chapter 4. For a more detailed description see [13][Chapter VII].

2.6.1 Stability

Definition 2.21. The bottleneck distance between two persistence diagrams X and Y is:

$$d_B(X, Y) = \inf_{\pi: X \to Y} \sup_{x \in X} \|x - \pi(x)\|_\infty$$

where π is a bijection between X and Y.

For this definition to make sense, we assume (see [9]) that points on the diagonal have infinite multiplicity. Therefore, the bottleneck distance is the maximal distance between points in two diagrams for an optimal matching π. A stability result, which we give below, asserts that persistence diagrams of two close functions are also close with respect to the bottleneck distance. We say that a topological space $\mathbb{X}$ is *triangulable* if there exists a simplicial complex that is homeomorphic to the space $\mathbb{X}$.

Theorem 2.22 (Stability for persistent homology, [9])**.** *Let $\mathbb{X}$ be a triangulable space and let $f, g : \mathbb{X} \to \mathbb{R}$ be continuous tame functions. Then the persistence diagrams satisfy:*

$$d_B(\mathrm{Dgm}(f), \mathrm{Dgm}(g)) \leq \|f - g\|_\infty.$$

Chapter 3

Persistent homology of self maps

In this chapter we present the problem of generalizing the persistent homology to maps from the point of view of category theory. We start with an elementary introduction to category theory, then we introduce the concept of towers – paths in categories – and finally we focus on the eigenvalue problem of a linear map.

3.1 Basic concepts of category theory

A *category* C is a pair consisting of a class of *objects* $\mathbb{O}(C)$ and *arrows* $\mathbb{A}(C)$ between the objects with the following two functions for arrows:

$$\text{src} \ : \mathbb{A}(C) \to \mathbb{O}(C)$$

taking an arrow into its *source* object, and

$$\text{trg} \ : \mathbb{A}(K) \to \mathbb{O}(K)$$

taking an arrow into its *target* object. The notation $a : ob_1 \to ob_2$ means that a is an arrow with ob_1 as its source and object ob_2 as its target. We also require that the following conditions hold:

i) for any two arrows $a, b \in \mathbb{A}(C)$ such that $\text{src}\, b = \text{trg}\, a$ an arrow $b \circ c$, called the composition of b and a, is given. It satisfies $\text{trg}\,(b \circ a) = \text{trg}\, b$ and $\text{src}\,(b \circ a) = \text{src}\, a$.

ii) for every object $ob \in \mathbb{O}(C)$ an identity arrow id_{ob} is given such that for any object ob' and arrows $a : ob \to ob'$, $b : ob' \to ob$ we have $a \circ id_{ob} = a$ and $\text{id}_{ob} \circ b = b$. It is called the identity arrow.

iii) for any three arrows $a : ob_1 \to ob_2$, $b : ob_2 \to ob_3$ and $c : ob_3 \to ob_4$ the associativity rule is fulfilled:

$$(c \circ b) \circ a = c \circ (b \circ a).$$

An arrow $a : ob_1 \to ob_2$ is *invertible* if there exists an arrow $a^{-1} : ob_2 \to ob_1$, called the inverse of a, such that compositions $a \circ a^{-1}$ and $a^{-1} \circ a$ are identities respectively for ob_2 and ob_1. An *initial object* $ob \in \mathbb{O}(C)$ is an object s.t. for $x \in \mathbb{O}(C)$ there exists only one arrow $ob \to x$. A *terminal object* $ob \in \mathbb{O}(C)$ is an objects s.t. for $x \in \mathbb{O}(C)$ there exist exactly one arrow $x \to ob$. A *zero object* is an object that is both initial and terminal.

Example 3.1. In the category of sets, with sets being object and maps between the sets being arrows the empty set is an initial object. All singletons are terminal objects and there are no zero objects.

Two objects are isomorphic if there exists an invertible arrow between them.

A diagram

$$\begin{array}{ccc} ob_1 & \xrightarrow{a} & ob_2 \\ \downarrow{\scriptstyle c} & & \downarrow{\scriptstyle d} \\ ob_3 & \xrightarrow{b} & ob_4 \end{array}$$

is said to be *commutative* if $d \circ a = b \circ c$.

Two arrows $a : ob_a \to ob'_a$ and $b : ob_b \to ob'_b$ are conjugate if there exist invertible arrows $c : ob_a \to ob_b$ and $c' : ob'_a \to ob'_b$ such that $c' \circ a = b \circ c$, that is the diagram

$$\begin{array}{ccc} ob_a & \xrightarrow{a} & ob'_a \\ \downarrow{\scriptstyle c} & & \downarrow{\scriptstyle c'} \\ ob_b & \xrightarrow{b} & ob'_b \end{array}$$

commutes. The most fundamental concept of category theory is a functor, which is an arrow between categories. The functor F maps objects and arrows from a category C_1 to objects and arrows, respectively, of another category C_2. We require the functor to map identity arrows of the first category to the identity arrows of the second category, and to preserve composition of arrows, that is: $F(b \circ a) = F(b) \circ F(a)$ for all arrows $a : ob_1 \to ob_2$, $b : ob_2 \to ob_3$ and $ob_1, ob_2, ob_3 \in \mathbb{O}(C_1)$.

To simplify our work we define the category $\mathbf{Iso}(C)$, in which we identify isomorphic objects and conjugate morphisms. This is useful if we are

interested in a characterization of objects that share the same structure, i.e. being isomorphic.

An example of a category is the category of topological spaces and continuous maps between them, which we denote **Top**.

3.2 Partial functions

Let X, Y be sets. A *partial function* $\xi : X \nrightarrow Y$ is a relation $\xi \subset X \times Y$ such that if $(x, y) \in \xi$ and $(x, z) \in \xi$ then $y = z$. The *domain* of ξ is

$$\operatorname{dom} \xi = \{x \in X \ : \ \exists_{y \in Y} \text{ such that } (x, y) \in \xi\}.$$

If $x \in \operatorname{dom} \xi$, then we denote by $\xi(x)$ the unique y s.t. $(x, y) \in \xi$. Note that there may exist $x \in X$ for which $\xi(x)$ is not defined. We define the *image* of ξ by

$$\operatorname{im} \xi := \{\xi(x) \ : \ x \in \operatorname{dom} \xi\}$$

and the *kernel* of ξ by

$$\ker \xi := X \setminus \operatorname{dom} \xi.$$

We compose two partial functions as relations: if $\alpha(a) = b$ and $\beta(b) = c$ then $(\beta \circ \alpha)(a) = c$. We say that a partial function $\xi : X \nrightarrow Y$ is *injective* if for all $x_1, x_2 \in X$ the equality $\xi(x_1) = \xi(x_2)$ implies $x_1 = x_2$. Note that sets and partial functions form a category. We denote it by **Part**. For the purpose of this work we limit objects in this category to finite sets.

3.3 Matchings

Of particular interest are special types of partial functions, namely matchings:

Definition 3.2. A *matching* $\alpha : A \nrightarrow B$ is an injective partial function.

Proposition 3.3. *An inverse α^{-1} of a matching α is also a matching.*

Proof. To show that the inverse is a matching it is enough to show that α^{-1} is a partial function which is injective. To see that α^{-1} is a partial function assume that $(b, a_1), (b, a_2) \in \alpha^{-1}$. Then $(a_1, b), (a_2, b) \in \alpha$. Hence $a_1 = a_2$ by the injectivity of α. Assume α^{-1} is not injective. Then there exist $b_1, b_2 \in B, b_1 \neq b_2$ such that $\alpha^{-1}(b_1) = \alpha^{-1}(b_2) = a$ for some $a \in A$. Hence, $\alpha(a) = b_1$ and $\alpha(a) = b_2$ which contradicts the assumption that α is a partial function. □

If A and B are finite we can enumerate their elements. Assume $A = \{a_1, a_2, \ldots, a_p\}$ and $B = \{b_1, b_2, \ldots, b_q\}$. Then, a q-by-p matrix M is the *matrix representation* of α if:

$$M[i,j] = \begin{cases} 1 & \text{if } (a_j, b_i) \in \alpha, \\ 0 & \text{otherwise.} \end{cases}$$

It is easy to show that the matrix of a matching has at most one non-zero entry in every row and every column. We call such a *matrix a matching* matrix. We define the *rank* of the matching matrix as the number of non-zero columns of M. Note that the rank of the matching matrix does not change under permutations of rows or columns. Thus, when elements in A or B are reordered the rank of the matrix of α does not change. We call it the rank of α and denote $\operatorname{rank}\alpha$. It is the same as the number of non-zero rows, as there is one-to-one correspondence between non-zero rows and columns.

Remark 3.4. *For any matching α*

$$\operatorname{rank}\alpha = \operatorname{card}\operatorname{dom}\alpha = \operatorname{card}\operatorname{im}\alpha.$$

The following property of matchings is convenient in the computation of the rank of compositions:

Proposition 3.5. *Let $\alpha : A \twoheadrightarrow B$, $\beta : B \twoheadrightarrow C$ be matchings, then:*

$$\operatorname{card}(\operatorname{dom}\beta \setminus \operatorname{im}\alpha) = \operatorname{rank}\beta - \operatorname{rank}\beta\alpha.$$

Proof. First observe that

$$\beta(\operatorname{dom}\beta \cap \operatorname{im}\alpha) = \operatorname{im}\beta \cap \beta(\operatorname{im}\alpha) = \beta(\operatorname{im}\alpha) = \operatorname{im}\beta\alpha.$$

Thus

$$\operatorname{card}(\operatorname{dom}\beta \cap \operatorname{im}\alpha) = \operatorname{card}\beta(\operatorname{dom}\beta \cap \operatorname{im}\alpha) = \operatorname{card}\operatorname{im}\beta\alpha = \operatorname{rank}\beta\alpha.$$

We have the decomposition into disjoint sets

$$\operatorname{dom}\beta = (\operatorname{dom}\beta \cap \operatorname{im}\alpha) \cup (\operatorname{dom}\beta \setminus \operatorname{im}\alpha).$$

Therefore,

$$\begin{aligned} \operatorname{rank}\beta &= \operatorname{card}\operatorname{dom}\beta = \operatorname{card}(\operatorname{dom}\beta \cap \operatorname{im}\alpha) + \operatorname{card}(\operatorname{dom}\beta \setminus \operatorname{im}\alpha) \\ &= \operatorname{rank}\beta\alpha + \operatorname{card}(\operatorname{dom}\beta \setminus \operatorname{im}\alpha), \end{aligned}$$

which proves the required identity. □

The following proposition lets us compute the rank of the composition of three matchings:

Proposition 3.6. *Let $\alpha : A \nrightarrow B$, $\beta : B \nrightarrow C$ and $\gamma : C \nrightarrow D$ be matchings, then:*

$$\operatorname{card}(\operatorname{dom}\beta \setminus \operatorname{im}\alpha \cap \ker\gamma\beta) = (\operatorname{rank}\beta - \operatorname{rank}\beta\alpha) - (\operatorname{rank}\gamma\beta - \operatorname{rank}\gamma\beta\alpha).$$

Proof. Since obviously $\operatorname{dom}\gamma\beta \subset \operatorname{dom}\beta$, we have

$$\operatorname{dom}\beta \setminus \ker\gamma\beta = \operatorname{dom}\beta \cap B \setminus \ker\gamma\beta = \operatorname{dom}\beta \cap \operatorname{dom}\gamma\beta = \operatorname{dom}\gamma\beta$$

Therefore, we have the following decomposition into disjoint sets

$$\begin{aligned} \operatorname{dom}\beta \setminus \operatorname{im}\alpha &= (\operatorname{dom}\beta \setminus \operatorname{im}\alpha \cap \ker\gamma\beta) \cup (\operatorname{dom}\beta \setminus \operatorname{im}\alpha \setminus \ker\gamma\beta) \\ &= (\operatorname{dom}\beta \setminus \operatorname{im}\alpha \cap \ker\gamma\beta) \cup (\operatorname{dom}\gamma\beta \setminus \operatorname{im}\alpha). \end{aligned}$$

It follows that

$$\operatorname{card}(\operatorname{dom}\beta \setminus \operatorname{im}\alpha) = \operatorname{card}(\operatorname{dom}\beta \setminus \operatorname{im}\alpha \cap \ker\gamma\beta) + \operatorname{card}(\operatorname{dom}\gamma\beta \setminus \operatorname{im}\alpha).$$

Thus, we get the conclusion from Prop. 3.5. □

We define the *category of matchings* **Mch** as the category whose objects are finite sets and morphisms between them are matchings.

3.4 Vector spaces

For the needs of this thesis we restrict our attention to finitely dimensional vector spaces. Let $\mathbb{F}$ be a field. We denote the category of finitely dimensional vector spaces over $\mathbb{F}$ by **Vect**. The objects in the category **Vect** are vector spaces and the arrows are the linear maps between them. For a linear map $f : U \to V$ we use the standard notation for the kernel

$$\ker f := \{x \in U : f(x) = 0\},$$

the image

$$\operatorname{im} f := \{f(x) : x \in U\}$$

and the rank

$$\operatorname{rank} f := \dim \operatorname{im} f.$$

Proposition 3.7. *Let $A = \{u_1, \ldots, u_n\}$ be a basis of U, $B = \{v_1, \ldots, v_m\}$ a basis of V and M the matrix representation of a map f in bases A and B. Assume M is in the matching form. Then, we have a well defined partial function $\alpha : A \nrightarrow B$ such that $\alpha(a) = b$ whenever $v(a) = b$. Moreover,*

i) $\ker \alpha = \ker f \cap A$,

ii) $\operatorname{im} \alpha = \operatorname{im} f \cap B$,

iii) $\operatorname{rank} \alpha = \operatorname{rank} f$.

Lemma 3.8. *For every linear map $f : U \to V$ there exist bases A in U and B in V such that the matrix of f in these bases is a matching matrix.*

Proof. It is a simple consequence of a Smith normalization algorithm. For details see [20][Chapter 3.3]. □

It is useful to notice that the matrix of f in a matching form is not necessarily unique.

Proposition 3.9. *Let A, A' be bases of U, B, B' bases of V such that*

$$f : U \to V$$

has a matching matrix in bases A, B and A', B'. Then there exist bijections $\mu : A \to A', \nu : B \to B'$ s.t. the following diagram is commutative

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ \downarrow{\scriptstyle \mu} & & \downarrow{\scriptstyle \nu} \\ A' & \xrightarrow{f} & B' \end{array}$$

□

3.4.1 Endomorphisms

An endomorphism is an arrow that has the same target and source. Given category C we construct a new category of endomorphisms called **Endo**(C), with objects being arrows in K. An arrow between objects $f : U \to U$ and $g : V \to V$ in **Endo**(C) is an arrow $\alpha : U \to V$ such that the diagram

$$\begin{array}{ccc} U & \xrightarrow{f} & U \\ \downarrow{\scriptstyle \alpha} & & \downarrow{\scriptstyle \alpha} \\ V & \xrightarrow{g} & V \end{array}$$

commutes. If C is known from the context we will omit it in the notation and write **Endo**.

3.4.2 Pairs

Similarly to the construction of **Endo**(C) we introduce the category of pairs **Pairs**(C) of arrows with the same source and target for a category C, that is objects are pairs of arrows (f, g) such that $f, g : U \to V$. Equivalently, one can write them in the form of a diagram

$$V \xleftarrow{f} U \xrightarrow{g} V \quad .$$

Arrows in **Pairs**(C) are pairs of arrows (α, β) such that $\alpha : V \to V$,$\beta : U \to U$ and the diagram

$$\begin{array}{ccccc} V & \xleftarrow{f_1} & U & \xrightarrow{g_1} & V \\ \downarrow{\scriptstyle \beta} & & \downarrow{\scriptstyle \alpha} & & \downarrow{\scriptstyle \beta} \\ V & \xleftarrow{f_2} & U & \xrightarrow{g_2} & V \end{array}$$

commutes.

3.5 Towers

A *tower* in a category C is a sequence of objects $(X_i)_{i\in\mathbb{Z}}$ and arrows $(\xi_i)_{i\in\mathbb{Z}}$ such that X_i is non-zero for finitely many i and $\xi_i : X_i \to X_{i+1}$. We denote this tower by (X_i, ξ_i) and use the notation ξ_i^j for the composition of arrows $\xi_{j-1} \circ \cdots \circ \xi_{i+1} \circ \xi_i$. Note that $\xi_i^j : X_i \to X_j$. Given two towers $T_1 = (X_i, \xi_i)$ and $T_2 = (Y_i, \eta_i)$, we define an arrow $\Phi : T_1 \to T_2$ as a collection $\Phi := (\varphi_i)_{i\in\mathbb{Z}}$ such that $\varphi_i : X_i \to Y_i$ is an arrow satisfying $\eta_i \circ \varphi_i = \varphi_{i+1} \circ \xi_i$ for every $i \in \mathbb{Z}$. To distinguish arrows between objects in C from arrows between towers in C we call the latter *morphisms*. We say that Φ is invertible if every φ_i is invertible.

3.5.1 Towers of matchings

Assume we have a tower $\mathcal{A} = (A_i, \alpha_i)$ in the category of matchings where each A_i is a finite set, and $\alpha_i : A_i \to A_{i+1}$ is a matching for every $i \in \mathbb{Z}$. Recall that α_i^j is a composition of maps $\alpha_i, \ldots, \alpha_{j-1}$. Let $[p \ldots q]$ denote an interval consisting of elements $i \in \mathbb{Z}$: such that $p \le i \le q$. Observe that an element of A_i is mapped through α_i^j either to an element of A_j or at some point it is mapped outside $\operatorname{dom} \alpha_k$ for $k \in [i \ldots (j-1)]$, and then it does not belong to $\operatorname{dom} \alpha_i^j$. The second important observation is that the pre-image

of an element of A_j under $(\alpha_i^j)^{-1}$ is either a singleton of an element of A_i or it is empty when the element is not in $\operatorname{im} \alpha_k$ for some $k \in [i \dots (j-1)]$. Therefore, we can think of the possible sequences of elements of $(A_i)_{i\in\mathbb{Z}}$ such that their image and preimage exist and they are mapped to each other. In this spirit we define an *interval* a that is a partial function $a : \mathbb{N} \nrightarrow A$ with a domain $[k \dots l]$, where A is the disjoint union of the A_i, $a(i) := a_i \in A_i$ and $a_{i+1} = \alpha_i(a_i)$ for $k \leq i < l$. We say a is a *maximal interval* if its domain is maximal, that is there is no interval a' with $\operatorname{dom} a' \supset \operatorname{dom} a$ such that a and a' agree on $\operatorname{dom} a$.

3.5.2 Persistence in towers of matchings

Similarly to the definitions in Section 2.6 about persistent homology, we define a *persistence interval* as the domain of a maximal interval. Recall that $\operatorname{rank} \alpha_i^j$ is the number of pairs in α_i^j, therefore:

Proposition 3.10. $\operatorname{rank} \alpha_i^j$ *is the number of intervals with domain* $[i, j]$.

We have a counterpart of this claim for maximal intervals. Let $\#_{[i,j]} = \#_{[i,j]}(\mathcal{A})$ be the number of maximal intervals with domain $[i, j]$, then:

Proposition 3.11. $\#_{[i,j]} = \operatorname{rank} \alpha_i^j - \operatorname{rank} \alpha_{i-1}^j - \operatorname{rank} \alpha_i^{j+1} + \operatorname{rank} \alpha_{i-1}^{j+1}$

Proof. An element of a maximal interval with domain $[i..j]$ must be in the domain of α_i^j. Moreover, it does not lie the image of α_{i-1} and the domain of α_i^{j+1}. Hence, Prop. 3.11 follows from Prop. 3.6 for $\alpha := \alpha_{i-1}, \beta := \alpha_i^j$ and $\gamma := \alpha_j$. □

We generalize Def. 2.20 to towers of matchings:

Definition 3.12. The *persistence diagram* $\operatorname{Dgm}(\mathcal{A})$ of a tower $\mathcal{A}$ of matchings is the multiset of domains of maximal intervals with their multiplicity.

We conclude this section with an important theorem characterizing the tower up to a conjugacy:

Theorem 3.13. *Let $\mathcal{A}$ and $\mathcal{B}$ be towers in* ***Mch****, then the following conditions are equivalent:*

i) $\mathcal{A}$ *and* $\mathcal{B}$ *are isomorphic*

ii) $\operatorname{rank} \alpha_i^j = \operatorname{rank} \beta_i^j$ *for all* $i \leq j$

iii) $\operatorname{Dgm}(\mathcal{A}) = \operatorname{Dgm}(\mathcal{B})$

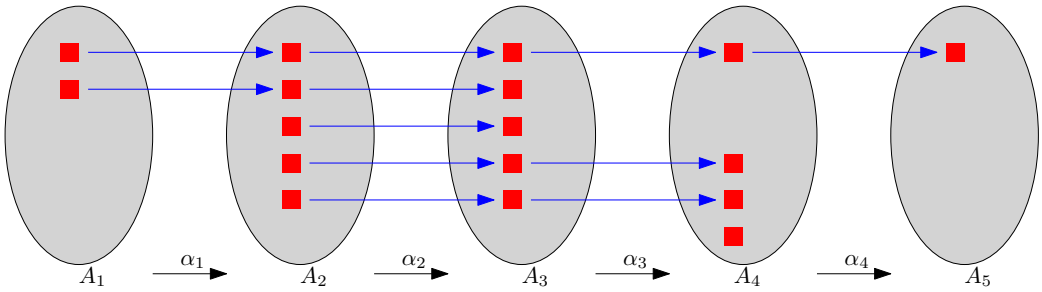


Figure 3.1: Diagram of a tower with finite sets A_i for $i \in \{1, \ldots, 5\}$ and partial functions $\alpha_i : A_i \nrightarrow A_{i+1}$ with red squares depicting elements of A_i, blue arrows illustrating pairs in matchings. The persistence diagram consists of the intervals $[1, 5]; [1, 3]; [2, 3]; [2, 4]; [2, 4]; [4, 4]$. Note that $[2, 4]$ is counted with multiplicity 2 and there are degenerate intervals such as $[4, 4]$.

Proof. i) $\Rightarrow$ ii) Fix $i \leq j$. Since $\mathcal{A}$ is isomorphic to $\mathcal{B}$, we have isomorphisms $\theta_i : A_i \to B_i$ and $\theta_j : A_j \to B_j$ such that the following diagram is commutative:

$$\begin{array}{ccccccc} \ldots & \longrightarrow & A_i & \longrightarrow \ldots \longrightarrow & A_j & \longrightarrow & \ldots \\ & & \downarrow{\scriptstyle \theta_i} & & \downarrow{\scriptstyle \theta_j} & & \\ \ldots & \longrightarrow & B_i & \longrightarrow \ldots \longrightarrow & B_j & \longrightarrow & \ldots \end{array}$$

Therefore, we can go both ways first using top arrows and then down or first down and then bottom arrows. Notice that

$$\begin{aligned} &\operatorname{rank} \alpha_i^j = \operatorname{card} \operatorname{im} \alpha_i^j = \operatorname{card} \alpha_i^j(A_i) = \operatorname{card} \theta_j(\alpha_i^j(A_i)) \\ &= \operatorname{card} \beta_i^j(\theta_i(A_i)) = \operatorname{card} \beta_i^j(B_i) = \operatorname{card} \operatorname{im} \beta_i^j = \operatorname{rank} \beta_i^j \end{aligned}$$

so isomorphisms θ_i and θ_j do not change ranks of the functions.

ii) $\Rightarrow$ iii) intervals in the persistence diagrams are determined by the values $\operatorname{rank} \alpha_i^j$ by Prop. 3.11.

iii) $\Rightarrow$ i) if $\operatorname{Dgm}(\mathcal{A})$ and $\operatorname{Dgm}(\mathcal{B})$ are the same then we can match intervals in $\mathcal{A}$ and $\mathcal{B}$. This matching enables us to construct isomorphism between $\mathcal{A}$ and $\mathcal{B}$.

□

3.5.3 Towers of linear maps

Let $\mathcal{U} = (U_i, v_i)$ be a tower of vector spaces.

Definition 3.14. We call a tower $\mathcal{A} = (A_i, \alpha_i)$ in **Mch** a *matching basis* of the tower $\mathcal{U}$ if A_i is linearly independent and spans U_i and the matrix representation of v_i in bases A_i is in matching form and agrees with α_i.

3.5.4 Persistence in towers of vector spaces

In Lem. 3.8 we state that a linear map admits bases in which its matrix is a matching. An analogous theorem is true for the tower of linear maps.

Theorem 3.15. *Any tower of vector spaces admits a matching basis.*

We postpone the proof to Section 4.8.

Proposition 3.16. *If $\mathcal{A} = (A_i, \alpha_i)$ is a basis of a tower $\mathcal{U} = (U_i, v_i)$, then:*

$$\operatorname{rank} \alpha_i^j = \operatorname{rank} v_i^j$$

for all $i \leq j$.

The above proposition is a consequence of the algorithm used to construct α_i from v_i in which we use elementary row and column additions, which do not alter rank of matrices. One can use a basis $\mathcal{A}$ of the tower $\mathcal{U}$ to generalize the notion of a persistence diagram:

Definition 3.17. A persistence diagram $\operatorname{Dgm}(\mathcal{U})$ of a tower $\mathcal{U}$ in **Vect** is the persistence diagram $\operatorname{Dgm}(\mathcal{A})$ of the basis tower $\mathcal{A}$ of $\mathcal{U}$.

The persistence diagram of a tower is well defined as a direct consequence of Prop. 3.16 and Prop. 3.11. Therefore, the persistence diagram does not depend on the choice of a basis. Similarly, we define $\#_{[i,j]}(\mathcal{U}) := \#_{[i,j]}(\mathcal{A})$. Prop. 3.11 generalizes to towers of vector spaces. We also generalize Thm. 3.13:

Theorem 3.18. *If $\mathcal{U} = (U_i, v_i)$ and $\mathcal{V} = (V_i, \phi_i)$ are towers in **Vect**, then the following conditions are equivalent:*

1. *$\mathcal{U}$ and $\mathcal{V}$ are isomorphic*
2. *$\operatorname{rank} v_i^j = \operatorname{rank} \phi_i^j$ for all $i \leq j$*
3. *$\operatorname{Dgm}(\mathcal{U}) = \operatorname{Dgm}(\mathcal{V})$*

Proof. Proofs of i)⇒ ii) and ii)⇒ iii) are analogous to the proofs in Thm. 3.13. We will show that iii)⇒ i). We use Thm. 3.15, which we will prove in Chapter 4. It allows us to select bases $\mathcal{A}$ for $\mathcal{U}$ and $\mathcal{B}$ for $\mathcal{V}$. Both $\mathcal{A}$ and $\mathcal{B}$ are matchings, hence by Thm. 3.13, the equality $\operatorname{Dgm}(\mathcal{A}) = \operatorname{Dgm}(\mathcal{B})$ implies isomorphisms between $\mathcal{A}$ and $\mathcal{B}$, which induces the isomorphism between $\mathcal{U}$ and $\mathcal{V}$. □

One can present the theory of persistent homology in the spirit of towers. Recall, that we start from a filtration of abstract simplicial complexes

$$K_1 \subset K_2 \subset \cdots \subset K_n,$$

or, equivalently, a tower in the category of abstract simplicial complexes and simplicial maps to which we apply a homology functor, i.e. compute homology groups:

$$H_p(K_1) \to H_p(K_2) \to \cdots \to H_p(K_n) \tag{3.19}$$

and induced linear maps $f_i : H_p(K_i) \to H_p(K_{i+1})$. We can generalize this definition by constructing tower of chain complexes $\mathcal{C} = (C_i, \gamma_i)$, where C_i are chains with basis K_i and γ_i is a chain map induced by inclusion of K_i into K_{i+1}. Writing $H'_i := H_p(C_i)$ we can equivalently present diagram (3.19) as a tower (H'_i, f_i) in **Vect**.

Proposition 3.20. *H_p is a homology functor from the category of abstract simplicial complexes to the category **Vect** of vector spaces.*

The proof is presented in [18].

A generalization of persistent homology known as zig-zag persistence is introduced in [5], where maps between chain complexes may be in both directions not only from left to right. Thus, the sequence of complexes may look like this:

$$K_1 \subset K_2 \supset K_3 \subset \cdots \supset K_n$$

Note that the persistent homology of towers can be easily modified to include this case if we allow inverses of matchings between levels of towers. Then, all definitions and proofs have to also include the inverses of matchings. But, by Prop. 3.3 the inverse of a matching is a matching. Therefore, the generalization of persistence in towers to zig-zag persistence is straightforward.

3.6 Eigenspaces

Definition 3.21. A Jordan block J is a square matrix of the following form:

$$\begin{bmatrix} \lambda & 1 & 0 & \cdots & 0 \\ 0 & \lambda & 1 & \cdots & 0 \\ 0 & 0 & \lambda & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & 0 & 0 & \cdots & \lambda \end{bmatrix}$$

where $\lambda \in \mathbb{F}$.

Definition 3.22. A square matrix is in Jordan Canonical Form, if it is a matrix of the form:

$$\begin{bmatrix} J_1 & 0 & \cdots & 0 \\ 0 & J_2 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & J_k \end{bmatrix}$$

where each J_i is a Jordan block.

3.6.1 Eigenvectors

For a fixed value $\lambda \in \mathbb{F}$ and an endomorphism $\gamma : V \to V$ we define the set:

$$E_\lambda(\gamma) := \{v \in V \mid \gamma(v) = \lambda v\} \tag{3.23}$$

We call $E_\lambda(\gamma)$ the *eigenspace* of λ. If $E_\lambda(\gamma) \neq 0$ then λ is called an *eigenvalue* of γ and non-zero vectors in $E_\lambda(\gamma)$ are called *eigenvectors*. Note that $E_\lambda(\gamma)$ is a subspace of V.

The *graph* of γ is $G\gamma = \{(v, \gamma(v)) | v \in V\}$ and we have two projections $\varphi, \psi : G\gamma \to V$ respectively to the first and the second coordinate. Moreover φ is invertible, hence $\gamma = \psi\varphi^{-1}$. It follows that the following diagram is commutative:

$$\begin{array}{ccccc} & & G\gamma & & \\ & \swarrow^{\varphi} & & \searrow^{\psi} & \\ V & & \xrightarrow{\quad\gamma\quad} & & V \end{array} \tag{3.24}$$

Eigenvectors for an eigenvalue t are given as non-zero solutions to the equation $\gamma(v) = \lambda v$. We can rewrite the equation as $\psi\varphi^{-1}(v) = \lambda v$. Let $u := \varphi^{-1}(v)$, then $\varphi(u) = \varphi\varphi^{-1}(v) = v$. This lead to the equation:

$$\psi(u) = \lambda\varphi(u) \tag{3.25}$$

which is the condition for u to be in $\bar{E}_\lambda(\varphi, \psi)$.

If we know only Γ an approximation of $G\gamma$, the projection $\varphi : \Gamma \to V$ may not be invertible. But, the equation (3.25) still makes sense. This motivates the study of eigenvalues and eigenvectors of pairs of maps.

3.6.2 Eigenvectors of pairs

We extend the definition of an eigenspace to a pair of linear maps:

Definition 3.26. For a pair $\varphi, \psi : U \to V$ and $\lambda \in \mathbb{F}$ we define

$$\bar{E}_\lambda(\varphi, \psi) := \{u \in U \mid \lambda\varphi(u) = \psi(u)\}$$

and

$$E_\lambda(\varphi, \psi) := \bar{E}_\lambda(\varphi, \psi)/(\bar{E}_\lambda(\varphi, \psi) \cap \ker \varphi)$$

If $E_\lambda(\varphi, \psi) \neq 0$ then we call λ an *eigenvalue of the pair* (φ, ψ) and $E_\lambda(\varphi, \psi)$ an *eigenspace of the pair*.

The following proposition will be used in the next chapters:

Proposition 3.27. $\bar{E}_\lambda(\varphi, \psi) \cap \ker \varphi = \ker \varphi \cap \ker \psi$

Proof. If $u \in \bar{E}_\lambda(\varphi, \psi) \cap \ker \varphi$ then $\varphi(u) = 0$ and $\lambda\varphi(u) = \psi(u)$, therefore $0 = \psi(u)$ and $u \in \ker \psi$. If $u \in \ker \varphi \cap \ker \psi$ then $\varphi(u) = 0$ and $\psi(u) = 0$, so $\lambda\varphi(u) = \psi(u)$ for any λ and $u \in \bar{E}_\lambda(\varphi, \psi)$. □

Computing $\bar{E}_\lambda(\varphi, \psi)$ is sometimes called the generalized eigenvalue problem [2].

Eigenspace functor

Assume an eigenvalue λ is fixed. We extend E_λ to a functor from the category **Endo** to the category **Vect**.

Proposition 3.28. *Assume $\gamma : V \to V$, $\gamma' : V' \to V'$ and an arrow $w : (V, \gamma) \to (V', \gamma')$ between objects in **Endo** are given. Then*

$$w(E_\lambda(\gamma)) \subset E_\lambda(\gamma').$$

Proof. Let $v \in E_\lambda(\gamma)$. Then $\gamma(v) = \lambda v$ and $w(\gamma(v)) = w(\lambda v)$. From the commutativity of the diagram

$$\begin{array}{ccc} V & \xrightarrow{\gamma} & V \\ \downarrow{\scriptstyle w} & & \downarrow{\scriptstyle w} \\ V' & \xrightarrow{\gamma'} & V' \end{array} \tag{3.29}$$

and the linearity of w we have $\gamma'(w(v)) = \lambda w(v)$. Thus $w(v) \in E_\lambda(\gamma')$, which proves the proposition. □

Prop. 3.28 allows us to define the restriction $E_\lambda(w) : E_\lambda(\gamma) \to E_\lambda(\gamma')$. Similarly, we extend E_λ to a functor from **Pairs(Vect)** to **Vect**.

Proposition 3.30. *Let $\varphi, \psi : U \to V$ and $\varphi', \psi' : U' \to V'$ be objects in* ***Pairs(Vect)*** *and $(v, w) : \varphi, \psi \to \varphi', \psi'$ be an arrow in* ***Pairs(Vect)****. Then $v(\bar{E}_\lambda(\varphi, \psi)) \subset \bar{E}_\lambda(\varphi', \psi')$.*

Proof. Let $u \in \bar{E}_\lambda(\varphi, \psi)$. Then $\varphi(u) = \lambda\psi(u)$ and $w(\varphi(u)) = w(\lambda\psi(u))$. From the commutativity of the diagram

$$\begin{array}{ccccc} V & \xleftarrow{\varphi} & U & \xrightarrow{\psi} & V \\ \downarrow{\scriptstyle w} & & \downarrow{\scriptstyle v} & & \downarrow{\scriptstyle w} \\ V' & \xleftarrow{\varphi'} & U' & \xrightarrow{\psi'} & V' \end{array} \tag{3.31}$$

and the linearity of w we have $\varphi'(v(u)) = \lambda\psi'(v(u))$. Thus $v(u) \in \bar{E}_\lambda(\varphi', \psi')$. □

We define $\bar{E}_\lambda(v, w)$ as $v|_{E_\lambda(\varphi,\psi)} : \bar{E}_\lambda(\varphi, \psi) \to \bar{E}_\lambda(\varphi', \psi')$, which is the restriction of v to subspaces $\bar{E}_\lambda(\varphi, \psi)$ and $\bar{E}_\lambda(\varphi', \psi')$. This defines the functor

$$\bar{E}_\lambda : \mathbf{Pairs(Vect)} \to \mathbf{Vect}.$$

The following proposition is also a consequence of commutativity of (3.31).

Proposition 3.32. *Let $\varphi, \psi : U \to V$ and $\varphi', \psi' : U' \to V'$ be objects in* ***Pairs(Vect)*** *and $(v, w) : \varphi, \psi \to \varphi', \psi'$ be an arrow in* ***Pairs(Vect)****. Then $v(\bar{E}_\lambda(\varphi, \psi) \cap \ker\varphi) \subset \bar{E}_\lambda(\varphi', \psi') \cap \ker\varphi'$.* □

We define the eigenfunctor $E_\lambda : \mathbf{Pairs(Vect)} \to \mathbf{Vect}$ on a pair (v, w) as

$$E_\lambda(v, w)[u] := [\bar{E}_\lambda(v, w)(u)] \tag{3.33}$$

It follows from Prop. 3.32 that the definition (3.33) is well set up.

3.6.3 Generalized eigenvectors

Given $\lambda \in \mathbb{F}$ and an endomorphism $\gamma : U \to U$ one can generalize equation (3.23) to:

$$E_{\lambda,k}(\gamma) := \{v \in V \mid (\gamma - \lambda\mathrm{id})^k u = 0\}, \tag{3.34}$$

where $(\gamma - \lambda\mathrm{id})^k$ is the k-fold composition of $(\gamma - \lambda\mathrm{id})$ with itself, and $k \in \mathbb{Z}^+$. The space $E_{\lambda,k}(\gamma)$ is called the *generalized eigenspace* of λ. For $k = 1$ we get equation (3.23). For $k \geq 2$ we call the elements of $E_{\lambda,k}(\gamma)$ *generalized eigenvectors*. The *index* of a generalized eigenvector v is the smallest k such that $(\gamma - \lambda\mathrm{id})^k v = 0$.

Definition 3.35. Let $\{v_1, v_2, \ldots, v_k\} \subset E_{\lambda,k}$. Moreover, assume that v_k is a generalized eigenvector of index k and:

$$\begin{aligned} v_{k-1} &= (\gamma - \lambda \mathrm{id})(v_k), \\ v_{k-2} &= (\gamma - \lambda \mathrm{id})(v_{k-1}), \\ &\ldots, \\ v_2 &= (\gamma - \lambda \mathrm{id})(v_3), \\ v_1 &= (\gamma - \lambda \mathrm{id})(v_2). \end{aligned} \tag{3.36}$$

Then $(v_1, v_2, \ldots, v_k)$ is the *chain of generalized eigenvectors* of length k.

Theorem 3.37. *[21][Thm. 9.1.] Let V be a vector space over an algebraically closed field and let $\gamma : V \to V$ be an endomorphism. Then, there exists a basis B of V such that the matrix representation of γ in basis B is in Jordan Canonical Form, and vectors of B are generalized eigenvectors of γ.*

3.6.4 Generalized eigenvectors of pairs

Analogously to Def. 3.26 we introduce generalized eigenvectors of a pair.

Definition 3.38. For a pair $(\varphi, \psi) : U \to V$, $\lambda \in \mathbb{F}$ and $k \in \mathbb{Z}^+$ we define recursively a sequence of spaces:

$$\begin{aligned} \bar{E}_{\lambda,1}(\varphi, \psi) &:= \bar{E}_{\lambda}(\varphi, \psi), \\ \bar{E}_{\lambda,k}(\varphi, \psi) &:= \{u \in U | (\psi - \lambda\varphi)(u) \in \varphi(\bar{E}_{\lambda,k-1})\}. \end{aligned}$$

We define also:

$$E_{\lambda,k}(\varphi, \psi) = \bar{E}_{\lambda,k}(\varphi, \psi)/(\bar{E}_{\lambda,k}(\varphi, \psi) \cap \ker \varphi)$$

If $E_{\lambda,k}(\varphi, \psi) \neq 0$ then $E_{\lambda,k}(\varphi, \psi)$ is the *k-th level generalized eigenspace* of the pair φ, ψ.

Generalized eigenvectors of γ for an eigenvalue $\lambda \in \mathbb{F}$ are solutions of the equation:

$$(\gamma - \lambda \cdot \mathrm{id})^k(v) = 0$$

for some integer $k > 0$, where id is the identity map. If $k = 1$, then we get the ordinary eigenvectors. Finding generalized eigenvectors is equivalent to finding a collection of chains of generalized eigenvectors, such that vectors

from chains are linearly independent. Recall that $v_1, v_2, ..., v_k \in V$ is a chain if:

$$\begin{aligned} \gamma(v_1) - \lambda v_1 &= 0, \\ \gamma(v_2) - \lambda v_2 &= v_1, \\ &\dots, \\ \gamma(v_k) - \lambda v_k &= v_{k-1}. \end{aligned} \tag{3.39}$$

Proposition 3.40. *The chain* $(v_1, \dots, v_k)$ *of vectors from* (3.39) *fulfils the condition:*

$$(\gamma - \lambda \cdot \mathrm{id})^k(v_k) = 0 \tag{3.41}$$

Proof. Let $(v_1, \dots, v_k)$ be a chain s.t. (3.39) is true. Then for $k = 1$ the thesis is straightforward. Assume $(\gamma - \lambda \cdot \mathrm{id})^{k-1}(v_{k-1}) = 0$ holds. From (3.39) we have $\gamma(v_k) - \lambda v_k = v_{k-1}$, hence by the induction assumption: $(\gamma - \lambda \cdot \mathrm{id})^{k-1}(\gamma(v_k) - \lambda v_k) = 0$, therefore $(\gamma - \lambda \cdot \mathrm{id})^k(v_k) = 0$ and by the induction argument (3.41) holds for any k. □

If φ is invertible, from the commutativity of diagram (3.24) and from (3.39) we get:

$$\begin{aligned} \psi\varphi^{-1}(v_1) - \lambda v_1 &= 0, \\ \psi\varphi^{-1}(v_2) - \lambda v_2 &= v_1, \\ &\dots, \\ \psi\varphi^{-1}(v_k) - \lambda v_k &= v_{k-1}. \end{aligned}$$

Setting $\varphi^{-1}(v_i) = u_i \in G\gamma$ we obtain a system of equations

$$\begin{aligned} \psi(u_1) - \lambda\varphi(u_1) &= 0, \\ \psi(u_2) - \lambda\varphi(u_2) &= \varphi(u_1), \\ &\dots, \\ \psi(u_k) - \lambda\varphi(u_k) &= \varphi(u_{k-1}). \end{aligned} \tag{3.42}$$

which makes sense even if φ is not invertible. We can solve iteratively this system for subspaces of $G\gamma \subset V \times V$ starting from the first equation. The space spanned by the vectors $u_1, \dots, u_k$ after quotienting out the kernel of φ provides $E_{\lambda,k}(\varphi, \psi)$.

Functorial description

Similarly to the definition of the eigenspace functor E_λ we introduce the k-th level *generalized eigenspace functor* $E_{\lambda,k}$ from **Pairs(Vect)** to **Vect**. Let $\varphi, \psi : U \to V$ and $\varphi', \psi' : U' \to V'$ be objects in **Pairs(Vect)**. We have an arrow from φ, ψ to φ', ψ' that is a pair v, w for which diagram (3.31)

commutes. It suffices to check that $v(\bar{E}_{\lambda,k}(\varphi,\psi)) \subset \bar{E}_{\lambda,k}(\varphi',\psi')$ to define $E_{\lambda,k}(v,w)$.

Proposition 3.43. *Let $\varphi,\psi : U \to V$ and $\varphi',\psi' : U' \to V'$ be objects in **Pairs(Vect)** and $(v,w) : \varphi,\psi \to \varphi',\psi'$ be an arrow in **Pairs(Vect)**. Then $v(\bar{E}_{\lambda,k}(\varphi,\psi)) \subset \bar{E}_{\lambda,k}(\varphi',\psi')$ for any k.*

Proof. For $k = 1$ the proof is identical as in Prop. 3.30. Assume the thesis holds for $k-1$. Let $u_k \in \bar{E}_{\lambda,k}(\varphi,\psi)$. Then $\psi(u_k) - \lambda\varphi(u_k) = \varphi(u_{k-1})$ for some $u_{k-1} \in \bar{E}_{\lambda,k-1}(\varphi,\psi)$. Thus $w(\psi(u_k) - \lambda\varphi(u_k)) = w(\varphi(u_{k-1}))$. From the commutativity of the diagram

$$\begin{array}{ccccc} V & \xleftarrow{\varphi} & U & \xrightarrow{\psi} & V \\ \downarrow{\scriptstyle w} & & \downarrow{\scriptstyle v} & & \downarrow{\scriptstyle w} \\ V' & \xleftarrow{\varphi'} & U' & \xrightarrow{\psi'} & V' \end{array} \tag{3.44}$$

and the linearity of w we have $\psi'(v(u_k)) - \lambda\varphi'(v(u_k)) = \varphi'(v(u_{k-1}))$. From the inductive assumption $v(u_{k-1}) \in \bar{E}_{\lambda,k-1}(\varphi',\psi')$. Thus $v(u_k) \in \bar{E}_{\lambda,k}(\varphi',\psi')$. □

We define $\bar{E}_{\lambda,k}(v,w)$ to be $v|_{E_{\lambda,k}(\varphi,\psi)} : \bar{E}_{\lambda,k}(\varphi,\psi) \to \bar{E}_{\lambda,k}(\varphi',\psi')$. This defines the functor

$$\bar{E}_{\lambda,k} : \mathbf{Pairs(Vect)} \to \mathbf{Vect}.$$

The following proposition can be proved analogously to Prop. 3.43:

Proposition 3.45. *Let $\varphi,\psi : U \to V$ and $\varphi',\psi' : U' \to V'$ be objects in **Pairs(Vect)** and $(v,w) : \varphi,\psi \to \varphi',\psi'$ be an arrow in **Pairs(Vect)**. Then $v(\bar{E}_{\lambda,k}(\varphi,\psi) \cap \ker\varphi) \subset \bar{E}_{\lambda,k}(\varphi',\psi') \cap \ker\varphi'$.*

The *generalized eigenfunctor* $E_\lambda : \mathbf{Pairs(Vect)} \to \mathbf{Vect}$ on a pair (v,w) is

$$E_{\lambda,k}(v,w)[u] := [\bar{E}_{\lambda,k}(v,w)(u)] \tag{3.46}$$

From Prop. 3.45 it follows that definition (3.46) is well set up.

3.7 Persistence of eigenspaces

Assume $\mathcal{M}$ is a tower of endomorphisms (γ_i, v_i), such that $\gamma_i : V_i \to V_i$ are objects, $v_i : V_i \to V_{i+1}$ are arrows and $v_i \circ \gamma_i = \gamma_{i+1} \circ v_i$, or equivalently the

following diagram is commutative:

$$\begin{array}{ccccccc} \dots & \xrightarrow{v_{i-1}} & V_i & \xrightarrow{v_i} & V_{i+1} & \xrightarrow{v_{i+1}} & \dots \\ & & \downarrow{\scriptstyle \gamma_i} & & \downarrow{\scriptstyle \gamma_{i+1}} & & \\ \dots & \xrightarrow{v_{i-1}} & V_i & \xrightarrow{v_i} & V_{i+1} & \xrightarrow{v_{i+1}} & \dots \end{array}$$

After applying the eigenspace functor we get the tower $\mathcal{E}_\lambda := E_\lambda(\mathcal{M})$:

$$\dots \xrightarrow{\delta_{\lambda,i-1}} E_\lambda(\gamma_i) \xrightarrow{\delta_{\lambda,i}} E_\lambda(\gamma_{i+1}) \xrightarrow{\delta_{\lambda,i+1}} \dots$$

Thus, we have a functor E_λ from towers of **Endo** to towers of **Vect**, where $\delta_{\lambda,i} := E_\lambda(v_i)$ is the restriction of v_i to $E_\lambda(\gamma_i)$ and $E_\lambda(\gamma_{i+1})$.

We also generalize the eigenspace functor to towers in **Pairs(Vect)**. Assume we have a tower $\mathcal{M}$ in **Pairs(Vect)**:

$$\begin{array}{ccccccc} \dots & \xrightarrow{w_{i-1}} & V_i & \xrightarrow{w_i} & V_{i+1} & \xrightarrow{w_{i+1}} & \dots \\ & & \uparrow{\scriptstyle \varphi_i} & & \uparrow{\scriptstyle \varphi_{i+1}} & & \\ \dots & \xrightarrow{v_{i-1}} & U_i & \xrightarrow{v_i} & U_{i+1} & \xrightarrow{v_{i+1}} & \dots \\ & & \downarrow{\scriptstyle \psi_i} & & \downarrow{\scriptstyle \psi_{i+1}} & & \\ \dots & \xrightarrow{w_{i-1}} & V_i & \xrightarrow{w_i} & V_{i+1} & \xrightarrow{w_{i+1}} & \dots \end{array} \tag{3.47}$$

After applying the eigenspace functor E_t for pairs we get the tower:

$$\dots \xrightarrow{\epsilon_{\lambda,i-1}} E_\lambda(\varphi_i, \psi_i) \xrightarrow{\epsilon_{\lambda,i}} E_\lambda(\varphi_{i+1}, \psi_{i+1}) \xrightarrow{\epsilon_{\lambda,i+1}} \dots \tag{3.48}$$

which we denote by $\mathcal{E}$. The above diagram is an extended and rotated version of (3.31), such that φ and ψ maps are now vertical. Given a pair $\varphi_i, \psi_i : U_i \to V_i$ we send it to the vector space $E_\lambda(\varphi_i, \psi_i) \subset U_i$, and $v_i : U_i \to U_{i+1}$ is sent to its restriction $\epsilon_{\lambda,i} := v_i|_{E_\lambda(\varphi_i,\psi_i)}$. However, $E_\lambda(\varphi_i, \psi_i)$ is a quotient module, thus we need to keep in mind that $\epsilon_{\lambda,i}([u])$ where $[u]$ is an equivalence class in $E_\lambda(\varphi_i, \psi_i)$ is mapped into the class $[v_i(u)]$.

The persistence diagram $\mathrm{Dgm}(\mathcal{E})$ of the tower (3.48) is called the *λ-persistence diagram* of the tower $\mathcal{M}$. Similarly, we define the *generalized λ-persistence diagram* of $\mathcal{M}$ to be the persistence diagram of the tower obtained by applying the generalized eigenspace functor $E_{\lambda,k}$ to the tower $\mathcal{M}$.

3.8 Persistence of self maps

Let $\mathbb{X} \subset \mathbb{R}^l$ be a topological space with topology induced by the Euclidean metric and $f : \mathbb{X} \to \mathbb{X}$ be a continuous map which we call a *self-map*.

We use the Euclidean metric and denote the distance by $\|x - y\|$ for $x, y \in \mathbb{R}^l$. For graphs, which are subsets of $\mathbb{R}^l \times \mathbb{R}^l$ we use the product metric distance, i.e.

$$\|(x, y) - (x', y')\| := \max\{\|x - x'\|, \|y - y'\|\}$$

for $(x, y), (x', y') \in \mathbb{R}^l \times \mathbb{R}^l$. Let p and q be the projections from the graph Gf of the map f to the first and the second coordinate. The following diagram is commutative:

$$\begin{array}{ccccc} & & Gf & & \\ & \swarrow^{p} & & \searrow^{q} & \\ \mathbb{X} & & \xrightarrow{\quad f \quad} & & \mathbb{X} \end{array} \tag{3.49}$$

The following proposition is a direct consequence of the fact that p is continuous and invertible:

Proposition 3.50.

$$H_*(f) = H_*(q) \circ H_*(p)^{-1}$$

Thus, we can compute the homology of a map by computing the homology of projections. The advantage is that p and q lead directly to the chain maps, whereas f may not.

3.8.1 Persistence via projections

We assume that we only have a finite approximation $S \subset \mathbb{X}$ of the space we are interested in. The map f is known by an approximation $g : S \to S$ such that $g(s)$ is some approximation of $f(s)$. We construct the Vietoris–Rips complex on points S for the threshold r, and denote it by $K_r := VR_S(r)$. Similarly we define $L_r := VR_{Gg}(r)$. Let $p_r : L_r \to K_r$ be the simplicial map induced by the projection $Gg \ni (x, y) \mapsto x \in S$ to the first coordinate x, and $q_r : L_r \to K_r$ be the simplicial map induced by the projection $Gg \ni (x, y) \mapsto y \in S$ to the second coordinate. If $\sigma \in L_r$, then the the pairwise distance of points in σ is less than r. Since the distance in Gg is the maximum of distances of the first and the second coordinate, the pairwise distances between points in $p_r(\sigma)$ are less than r. Hence $p_r(\sigma) \in K_r$. The same

argument holds for q_r, and as a consequence both p_r and q_r are simplicial maps.

Let $r_1, r_2, \ldots, r_n$ be a sequence s.t. $r_i \leq r_{i+1}$ and $r_i \in \mathbb{R}$. Denote $L_{r_i} = VR_{Gg}(r_i)$ by L_i and $K_{r_i} = VR_S(r_i)$ by K_i. We get a filtration build on points S:

$$K_0 \subset K_1 \subset \cdots \subset K_n \tag{3.51}$$

and another filtration built on points Gg:

$$L_0 \subset L_1 \subset \cdots \subset L_n \tag{3.52}$$

Let $p_i := p_{r_i}$ and $q_i := q_{r_i}$. We have a collection of pairs of simplicial maps: $p_i, q_i : L_i \to K_i$:

$$\begin{array}{ccccccc} \ldots & \subseteq & K_i & \subseteq & K_{i+1} & \subseteq & \ldots \\ & & {\scriptstyle p_i}\uparrow & & {\scriptstyle p_{i+1}}\uparrow & & \\ \ldots & \subseteq & L_i & \subseteq & L_{i+1} & \subseteq & \ldots \\ & & {\scriptstyle q_i}\downarrow & & {\scriptstyle q_{i+1}}\downarrow & & \\ \ldots & \subseteq & K_i & \subseteq & K_{i+1} & \subseteq & \ldots \end{array} \tag{3.53}$$

Applying the homology functor to sequences in Eqn. (3.51) and (3.52) and to the simplicial maps p_i, q_i we get the tower in **Pairs(Vect)** that is shown in Eqn. (3.47). Denote that tower by $\mathcal{M}$. The persistent diagram $\mathrm{Dgm}(\mathcal{E})$ of the tower $\mathcal{E}$ obtained by applying eigenfuctor E_λ to the tower $\mathcal{M}$ is called the λ-*persistence diagram* of g. The persistent diagram $\mathrm{Dgm}(\mathcal{E})$ of the tower $\mathcal{F}$ obtained by applying the generalied eigenfunctor $E_{\lambda,k}$ is called the k-th *level* λ-*generalized persistence diagram* of g.

3.8.2 Persistence via inclusions

The approach presented in the previous section has one disadvantage: we have to compute the graph, hence the dimension we are working with is doubled with respect to the dimension of domain of a map. This causes additional overhead, which is eliminated in the approach of this section.

Let K be a full Vietoris–Rips simplicial complex $VR_S(\infty)$. Let $\kappa : K \to K$ be a simplicial map induced by the map on points $g : S \to S$. The map κ is simplicial on K in this setting, however it is not true for arbitrarily chosen parameters of r of the complex $VR_S(r)$. In the example in Section 1.1.1 for a filtration that does not originate from Vietoris–Rips simplicial complex we

face the same problem. We denote the Vietoris–Rips complex on points S and for the threshold r by $K_r := VR_S(r)$.

To solve the problem of missing images of the map κ we construct a filtration of subcomplexes such that the restriction of the map κ is simplicial.

Definition 3.54. Let $\bar{K}_r$ be the maximal subcomplex $\bar{K}_r \subset K_r$ such that $\kappa|_{\bar{K}_r} : \bar{K}_r \to K_r$ is a simplicial map. We denote $\kappa|_{\bar{K}_r}$ by κ'_r.

Notice that the above definition is equal to restricting κ to simplices that are mapped to objects in K_r. Let κ_r be the partial map obtained via described procedure.

For the simplicity we again assume that the Vietoris–Rips filtration is given as a sequence $\mathcal{K}$ of sublevel sets

$$\emptyset = K_0 \subset K_1 \subset K_2 \subset \cdots \subset K_n = K, \tag{3.55}$$

where K_i is K_{r_i}.We generalize Definition 3.54 to K_i by setting $\bar{K}_i$ to be the maximal subset of K_i s.t. κ is simplicial.

Notice that $\bar{K}_i \subset K_i$, and denote the inclusion map by ι_i. Combining sequences of K_i, $\bar{K}_i$ and simplicial maps ι_i, κ_i we get the diagram:

$$\begin{array}{ccccccc}
\dots & \subseteq & K_i & \subseteq & K_{i+1} & \subseteq & \dots \\
 & & {\scriptstyle \iota_i}\uparrow & & {\scriptstyle \iota_{i+1}}\uparrow & & \\
\dots & \subseteq & \bar{K}_i & \subseteq & \bar{K}_{i+1} & \subseteq & \dots \\
 & & {\scriptstyle \kappa'_i}\downarrow & & {\scriptstyle \kappa'_{i+1}}\downarrow & & \\
\dots & \subseteq & K_i & \subseteq & K_{i+1} & \subseteq & \dots
\end{array} \tag{3.56}$$

The following proposition is a consequence of the fact that $\iota_i : \bar{K}_i \to K_i$ is generated by an inclusion $\bar{K}_i \to K_i$ therefore every vertex is mapped to itself:

Proposition 3.57. *The inclusion map ι_i is equal to the map from $\bar{K}_i$ to K_i induced by the identity map on vertices.*

We apply the homology functor to (3.56) and get the tower $\mathcal{N}$ in **Pairs(Vect)** similar to the one presented in (3.47). The following lemma let us switch between the pair (p_r, q_r) and (κ'_r, ι_r).

Lemma 3.58 (Projection Lemma, [14])**.** *Let $\kappa_r : K_r \twoheadrightarrow K_r$ be the partial simiplicial map obtained by restricting κ to K_r, and let $p_r, q_r : L_r \to K_r$ be the simplicial maps induced by projections to the first and the second coordinate. Then $\kappa_r = q_r \circ p_r^{-1}$, for every $r \geq 0$.*

The following proposition is consequence of Lem. 3.58:

Proposition 3.59. *The persistence diagram* $\mathrm{Dgm}(\mathcal{F})$ *of the tower* $\mathcal{F}$ *obtained by applying the (generalized) eigenfunctor* E_λ *to the tower* $\mathcal{N}$ *agrees with the* λ*-(generalized) persistence diagram of* g*.*

We use Prop. 3.59 to compute λ-persistence diagrams via inclusion approach.

Example 3.60. Consider the filtration $\mathcal{K}$ introduced in Fig. 1.1. The filtration induced by κ is visible in Fig. 1.2. In the diagram $\mathrm{Dgm}(E_\lambda(H(\mathcal{K})))$ for $\lambda = 2$, where H is homology functor we have only one interval $[5, 6]$ with multiplicity 1.

Notice that for the presented approach we do not use any specific properties of Vietoris–Rips complexes. In particular, the assumption that (3.55) is constructed from Vietoris–Rips filtration may be lifted and the method may be applied to any filtration.

Chapter 4

Algorithm

Let $\mathbb{X} \subset \mathbb{R}^l$ be a topological space and let $f : \mathbb{X} \to \mathbb{X}$ be a continuous map. Assume we have a finite set S of points and a map $g : S \to S$ sampled from $\mathbb{X}$ and f. The sampling process may introduce errors, hence we do not assume that $g = f|_S$ or even that $S \subset \mathbb{X}$. In Chapter 5 we discuss how to deal with any type of noise including sampling errors. Our goal is to compute the persistent homology diagram of the tower of eigenspaces of g for a fixed eigenvalue $\lambda \in \mathbb{F}$.

4.1 Notation and representation

Let A be an m-by-n matrix. Then $A[i, .]$ denotes the i-th row, $A[., j]$ the j-th column, $A[i_1..i_2, .]$ sub-matrix composed of the rows $i_1 \dots i_2$, $A[., j_1..j_2]$ sub-matrix composed of the columns $j_1 \dots j_2$ and $A[i_1..i_2, j_1..j_2]$ sub-matrix composed of entries that lie in the intersection of rows $i_1..i_2$ and columns $j_1..j_2$.

Let G, H be subspaces of $\mathbb{F}^p$ and $H \subset G$. The quotient space G/H is represented as a basis $\{u_1, u_2, \dots, u_n\}$ of H and the set of vectors $\{u_{n+1}, \dots, u_m\}$ s.t. $\{u_1, \dots, u_m\}$ is a basis of G. In other words we store basis of H and the complement to a basis of G. Let columns of a matrix U be vectors $\{u_{n+1}, u_{n+2}, \dots, u_m\}$ and columns of a matrix U' be vectors $\{u_1, u_2, \dots, u_n\}$. We say that U, U' represents the quotient space G/H.

4.2 Linear algebra algorithms

We use three algorithms described in [20]:

- KERNEL(`matrix` A) Given an $m \times n$ matrix A on input the function returns

$n \times (n-k)$ matrix W such that the columns of W constitute a basis of $\ker A$ [20][Alg. 3.42].

- SOLVE(`matrix` A, `vector` b) If the problem $Ax = b$ has a solution then the function returns the vector x satisfying $Ax = b$, otherwise raises an error. [20][Alg. 3.54].

- QUOTIENTGROUP(`matrix` W, V). The algorithm returns a matrix U such that the vectors of U complemented by the vectors of V span the same subspace as the vectors of W. In other words it computes a representation of the quotient space G/H, where G is spanned by W, H is spanned by V and the representation is composed of vectors of U and V s.t. the columns of U and V constitute a basis of G.

In the next paragraphs we describe linear algebra procedures that are necessary in our approach and not described elsewhere.

4.2.1 Intersection

Assume a vector space $U \subset \mathbb{F}^k$ is spanned by a basis $\{u_1, \ldots, u_n\}$, forming k-by-n matrix, denoted A, with columns equal to $u_1, u_2, \ldots, u_n$. Similarly assume that a vector space $V \subset \mathbb{F}^k$ is represented by a k-by-m matrix B with columns forming a basis of V. The goal is to find a matrix C whose columns form a basis of the intersection $W := U \cap V$.

Algorithm 4.1 Intersection of two vector spaces

```
1: function spaceIntersection(matrix A, B)
2:     let n be the number of columns of A
3:     D = [A|B]
4:     K = kernel(D)
5:     matrix C = A · K[1..n, .]
6:     return C
7: end function
```

Proposition 4.1. *Algorithm 4.1 applied to vector spaces U and V with bases represented as columns of matrices A and B respectively computes a basis of the intersection $U \cap V$ represented by columns of the matrix C.*

Proof. Assume A is $k \times n$-matrix and B is $k \times m$ matrix. Observe that a vector $x \in U \cap V$ fulfils the condition

$$x = \sum_{i=1,..,n} \alpha_i A[., i]$$

for some sequence $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$ and

$$x = \sum_{i=1,..,m} \beta_i B[., i]$$

for some sequence $\beta_1, \ldots, \beta_m \in \mathbb{F}$. Thus,

$$\sum_{i=1,..,n} \alpha_i A[., i] = \sum_{i=1,..,m} \beta_i B[., i]. \tag{4.2}$$

Let D be a $k \times (n+m)$-matrix spanned by columns of the matrix A followed by columns of the matrix B. Applying the KERNEL procedure we get matrix K. Columns of K span $\ker D$, therefore if c is a column of K then $D \cdot c = 0$. Observe that

$$0 = D \cdot c = \sum_{i=1..(n+m)} c[i] \cdot D[i]. \tag{4.3}$$

The columns 1 to n of D agree with the columns of A and the columns $n+1, \ldots, n+m$ agree with the columns of B. If follows from (4.3) that:

$$\sum_{i=1,..,n} c[i] A[., i] = \sum_{i=1,..,m} -c[n+i] B[., i] \tag{4.4}$$

Hence, $A \cdot c[1..n] \in U \cap V$. This is true for every column c of the matrix K. The function KERNEL computes a basis of the kernel, hence $A \cdot K[1..n, .]$ is a basis of $U \cap V$. □

The dominant factor in the running time of Algorithm 4.1 is the computation of the kernel of the matrix $[A|B]$. For the matrices with field coefficients the complexity is equivalent to Gaussian elimination. Let A and B be $k \times n$ and $k \times m$ matrices. Without loss of generality assume $k < n$ and $k < m$.

Proposition 4.5. *Algorithm 4.1 has complexity $O((n+m)^3)$.* □

4.2.2 Restriction

Assume $U' \subset U \subset M \subset \mathbb{F}^m$. Let Q_1, Q_2 be a representation of U/U'. Similarly, assume $V' \subset V \subset N \subset \mathbb{F}^n$ and R_1, R_2 be a representation of V/V'. Let $f : M \to N$ be a linear map with the matrix representation A in the canonical basis. The following procedure computes the matrix of the map $f|_{U/U'} : U/U' \to V/V'$ in new bases:

Proposition 4.6. *Algorithm 4.2 applied to matrices A, Q_1, Q_2, R_1, R_2 computes the matrix representation B of the restriction $f|_{U/U'} : U/U' \to V/V'$ in the bases consisting of columns of Q_1, R_1.*

Algorithm 4.2 Restriction of a map.

```
1: function RESTRICTION(matrix A, Q_1, Q_2, R_1, R_2)
2:     C = A · Q_1
3:     let n be the number of columns of R_1
4:     R = [R_1|R_2]
5:     for i = 1.. number of columns of C do
6:         x = SOLVE(R, C[., i])
7:         B[., i] := x[1..n]
8:     end for
9:     return B
10: end function
```

Proof. We first compute the images of the representatives of the equivalence classes of U/U', that are columns of Q_1 under the map f. The map f is represented by the matrix A, so the images are columns of the matrix $C = A \cdot Q_1$. For every column $C[., i]$ we find its linear combination with respect to the columns of R_1 via solving the system of linear equations

$$R \cdot x = C[., i], \tag{4.7}$$

where $R := [R_1|R_2]$ is the matrix formed by R_1 followed by R_2. Notice that the equation (4.7) is equivalent to:

$$\sum_{j=1}^{n+m} R[., j] \cdot x[j] = \sum_{j=1}^{n} R_1[., j] \cdot x[j] + \sum_{j=1}^{m} R_2[., j] \cdot x[n + j] = C[., i]$$

where m is the number of column of R_2. For the map restriction it is enough to store coefficients related to the columns of R_1 in a new matrix B, so we take n coefficients of x. The coefficients of x related to columns of R_2 are not necessary because the solution lies in the quotient space. □

4.3 Computational topology algorithms

In this section we show basic algorithms and data structures in processing algebraic topology objects.

4.3.1 Vector representation of a chain.

Assume we have an abstract simplicial complex K and we want to store in the computer memory a chain $c \in C_p(K)$. First, we select some ordering of

simplices in K: $(\sigma_1, \sigma_2, \ldots, \sigma_n)$. The chosen ordering is not important, but it has to be fixed and cannot be changed while the program is running. Then, we store the chain c as a vector v of elements from $\mathbb{F}$, and the ith entry of v corresponds to the coefficient α_i of σ_i in the chain c.

4.3.2 Representation of a filtration

Let $\mathcal{K}$ be a filtration $K_1 \subset K_2 \subset \cdots \subset K_n = K$ of an abstract simplicial complex K. To save memory we do not store every subset K_i separately, but for a simplex $\sigma \in K$ we store only the index of a subset K_i such that $\sigma \in K_i$ and $\sigma \notin K_{i-1}$. We denote such index of σ by $w_{\mathcal{K}}(\sigma)$. Then, $K_i := \{\sigma | w_{\mathcal{K}}(\sigma) \leq i\}$. Notice that $j > w_{\mathcal{K}}(\sigma)$ implies $\sigma \in K_j$.

4.3.3 Induced chain maps

Let $g : K \to K$ be a simplicial map on a simplicial complex K. Algorithm 4.3 constructs the induced chain map. Let $c = \sum_{i=1}^{n} a_i \sigma_i$. The induced chain map is given by: $g(c) := \sum_{i=1}^{n} a_i \cdot g(\sigma_i)$, where $g(\sigma) = [g(v_1), \ldots, g(v_n)]$ for $\sigma_i = [v_1, \ldots, v_n]$. However, in Algorithm 4.3 we store simplices with vertices in a predefined order, hence we have to multiply a_i by the sign of the permutation $[g(v_1), \ldots, g(v_n)]$.

Example 4.8. Let $K = \{\{1, 2\}, \{2, 3\}\}$ and $g : K \to K$ be a map s.t. $g(1) = 2, g(2) = 1, g(3) = 3$. Assume $c = [1, 2] + [2, 3]$, hence $g(c) = [2, 1] + [1, 3] = -[1, 2] + [1, 3]$. In the algorithm we store $[1, 2]$ as the simplex $\{1, 2\}$ with the orientation 1 and $[2, 1]$ as the simplex $\{1, 2\}$ with the orientation -1.

Algorithm 4.3 Induced chain map

```
1: function chainMap(map g, chain c)
2:     c' := ∅
3:     for each a_i · σ_i ∈ c do
4:         [v_1, v_2, ..., v_n] := vertices of σ_i
5:         let σ' be a simplex with vertices {g(v_1), g(v_2), ..., g(v_n)}
6:         c' := c' + a_i·sign([g(v_1), g(v_2), ..., g(v_n)]) · σ'
7:     end for
8:     return c'
9: end function
```

Algorithm 4.3 has complexity of $O(np \log p)$ for a p-chain, where n is the number of elements in c. The term $p \log p$ comes from the computation of the

sign of a permutation. The sign of the permutation is calculated by counting the number of transpositions using for example merge sort algorithm.

4.4 Filtration

The first step we explain is the construction of a filtration of a simplicial complex with vertices in S. There are many ways to do it. In this thesis we choose the Vietoris–Rips complex for the sake of its simplicity.

We use a two-phase algorithm [27] to construct the 2-skeleton of the Vietoris–Rips on the set of points S. The algorithm proceeds as follows: in the first phase, we build a neighbourhood graph, i.e. the 1-skeleton of Vietoris–Rips complex with weights defined on edges. In the second phase we expand the neighbourhood graph up to a desired dimension. We use the incremental method to add simplices to the neighbourhood graph (for details see [27]).

We denote the procedure to construct the 2-skeleton of the full Vietoris–Rips filtration by VIETORISRIPSFILTRATION(S), where S is the set of points.

4.5 Filtration of domains

The Vietoris–Rips complex provides a method to deal with the reconstruction of the space. Unfortunately, as we already explained, there is no guarantee that a simplex in $\mathrm{VR}_S(r)$ will be mapped by a map κ induced by g to a simplex in $\mathrm{VR}_S(r)$, particularly when the map g is expanding. Let the Vietoris–Rips filtration be $\emptyset = K_0 \subset K_1 \subset K_2 \subset \cdots \subset K_n = K$. Recall that $\bar{K}_i$ is the maximal subset of K_i such that $\kappa|_{\bar{K}_i} : \bar{K}_i \to K_i$ is a simplicial map. The following procedure computes the sequence $\bar{\mathcal{K}}$ of domains: $\bar{K}_1 \subset \bar{K}_2 \subset \cdots \subset \bar{K}_n$.

Algorithm 4.4 Domains computation

```
1: function DOMFILTRATION(filtration K, map g)
2:     let K̄ be an empty filtration
3:     for each σ ∈ K do
4:         τ := CHAINMAP(g, σ)
5:         w_K̄(σ) := w_K(τ)
6:         add τ to K̄
7:     end for
8:     return (K̄, w_K̄)
9: end function
```

Proposition 4.9. *Algorithm 4.4 applied to a filtration $\mathcal{K}$ with the function $w_{\mathcal{K}}$ on its simplices representing a filtration $K_1 \subset K_2 \subset \cdots \subset K_n$ returns the filtration $\bar{\mathcal{K}}$ with the function $w_{\bar{\mathcal{K}}}$ representing the maximal filtration $\bar{K}_1 \subset \bar{K}_2 \subset \cdots \subset \bar{K}_n$, such that $\kappa|_{\bar{K}_i} : \bar{K}_i \to K_i$ is a simplicial map.* □

Time complexity of Algorithm 4.4 is linear with respect to the number of simplices in $\mathcal{K}$.

4.6 Persistent homology

For a filtration $\mathcal{K} = (K_1 \subset K_2 \subset K_n = K)$, we construct the persistent homology groups and get a tower in **Vect**. We define the *boundary matrix* D of $\mathcal{K}$:

$$D[k,l] := \langle \sigma_k, \partial(\sigma_l) \rangle \tag{4.10}$$

The ordering of simplices $\sigma_1, \sigma_2, \ldots, \sigma_m$ in the columns and the rows of D is consistent with the ordering in which they appear in the filtration: if $i < j$, $\sigma_k \in K_i$ and $\sigma_l \in K_j$ then $k < l$. Note that $D[k,l]$ is 1 (or -1) if σ_k is a face of σ_l with coefficient 1 (respectively -1) in $\partial(\sigma_l)$ or 0 otherwise.

Example 4.11. Let $\mathcal{K}$ be a filtration presented in Fig. 4.1. Let X denote the simplex $\{X\}$, XY the simplex $\{X, Y\}$ and so on. The ordering of the simplices in which they appear in the complex is:

$$A, B, C, D, AB, AD, CD, BD, BC, ABD, BCD.$$

The boundary matrix of $\mathcal{K}$ is shown in Table 4.1.

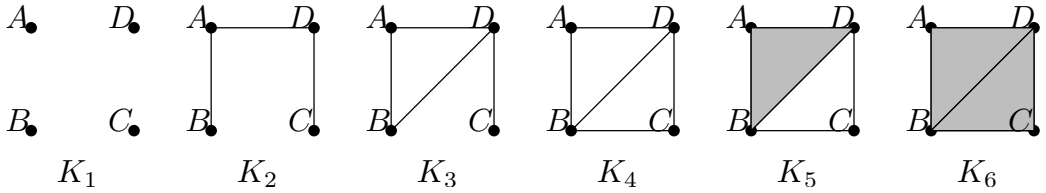


Figure 4.1: Filtration $\mathcal{K}$ consisting of seven levels $K_1, \ldots, K_7$.

The simplices of K are arranged into blocks: entries corresponding to simplices appearing for the first time in K_i belong to the i-th block of the matrix D and appear before blocks corresponding to K_{i+1},K_{i+2} and so on (see Fig. 4.2).

	A	B	C	D	AB	AD	CD	BD	BC	ABD	BCD
A					-1	-1					
B					1			-1	-1		
C							-1		1		
D						1	1	1			
AB										1	
AD										-1	
CD											1
BD										1	-1
BC											1
ABD											
BCD											

Table 4.1: Boundary matrix R of the filtration $\mathcal{K}$.

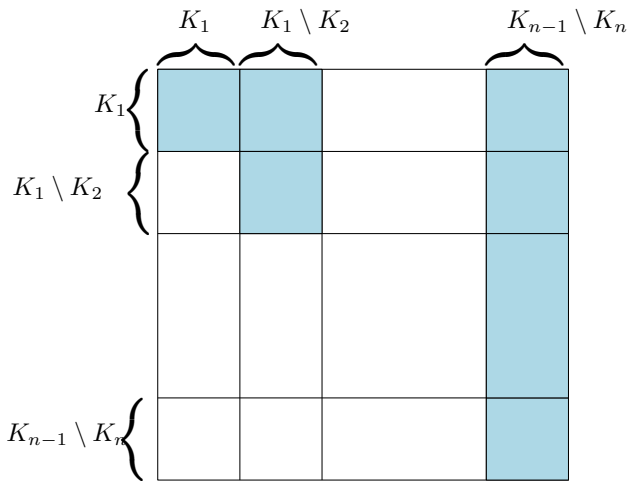


Figure 4.2: The structure of the matrix D: simplices are arranged into blocks, s.t. i-th block corresponds to new simplices in K_i.

The classical persistent homology algorithm uses left-to-right column additions to bring D into the so-called reduced form, from which the persistence diagram may be extracted. To explain this in detail we recall some notations and definitions. Given a matrix we let $low(j)$ denote the row index of the lowest non-zero entry in column j and we leave $low(j)$ undefined if column j is zero. We say that a non-zero column j of a matrix R is *reduced*, if there is no $j_0 \neq j$ such that $low(j) = low(j_0)$ in R. The matrix R is reduced when all its columns are reduced or zero.

The classical persistent homology algorithm [13][p. 153] consist in bring-

ing the boundary matrix to the reduced form R. Its generalization to the case of matrices over a general field $\mathbb{F}$ is presented as Algorithm 4.5. Note that by performing the same sequence of column additions as in Algorithm 4.5 but on the identity matrix instead of the boundary matrix we obtain a matrix B whose columns constitute the basis in which the boundary matrix takes the reduced form. The chains in B such that the corresponding column in R is zero are cycles.

Algorithm 4.5 Generalized classical persistent homology algorithm

```
1: function REDUCE(filtration 𝒦)
2:     let R be the boundary matrix of the filtration 𝒦
3:     n := number of columns of R
4:     for j = 1 to n do
5:         while there exists j0 < j with low(j0) = low(j) do
6:             α := R[low(j), j]/R[low(j), j0]
7:             R[., j] := R[., j] − α · R[., j0]
8:         end while
9:     end for
10:    extract persistence diagram dgm from R
11:    return dgm, R
12: end function
```

Theorem 4.12. *[25] The worst-case running time of Algorithm 4.5 is $\Omega(n^3)$ where n is the number of simplices in the filtration.*

Note that the practically observed running time of Algorithm 4.5 is significantly better than the worst case [3, 25]. In order to explain how the reduced form of the boundary matrix is used to obtain the persistence diagram we need a few more definitions. The simplex σ_j is said to be *positive* if the j-th column of the reduced boundary matrix R is zero. The simplex σ_j is called *negative* when the j-th column of the matrix R is non-zero. The following propositions are useful in the reconstruction of the persistence diagram from the reduced boundary matrix R.

Proposition 4.13. *[13][Ch. VII.1] Addition of a positive simplex σ_j into a filtration gives birth to a new homology class.*

Proposition 4.14. *[13][Ch. VII.1] Addition of a negative simplex σ_j gives death to a homology class. Moreover, the cycle representing the killed homology class is stored in the j-th column of the reduced boundary matrix.*

Proposition 4.15. *[13][Ch. VII.1] If $\sigma_i \in K_a$, $\sigma_j \in K_b$ are such that $low(j) = i$, then the interval $[a, b]$ belongs to the persistence diagram* $\mathrm{Dgm}(\mathcal{K})$ *where $[a, b]$ is an interval. The multiplicity of the interval $[a, b]$ is the number of pairs of simplices (σ_i, σ_j) with the requested properties.*

From Propositions 4.13 to 4.15 we can extract the persistence diagram $\mathrm{Dgm}(\mathcal{K})$. Indeed, it suffices to reduce the boundary matrix of a filtration $\mathcal{K}$ and output for every non-zero column j the interval $[a, b]$ s.t. $low(j) = i$, $\sigma_i \in K_a$ and $\sigma_j \in K_b$. We say that interval $[a, b]$ corresponds to the column i. The i-th element of the basis B is a cycle. We say that $[a, b]$ is generated by this cycle.

	A	B	C	D	AB	AD	-AD +CD	AB -AD +BD	CD -AD +AB +BC	ABD	BCD
A					-1	-1	1				
B					1						
C							-1				
D						1					
AB										1	
AD										-1	
CD											1
BD										1	1
BC											-1
ABD											
BCD											

Table 4.2: The reduced form of the boundary matrix of Ex. 4.11. Notice, that AB, AD, CD, ABD and BDC are negative simplices. Consider ABD: observe that $low(10) = 8$, column 10 corresponds to ABD appearing in K_5 and column 8 corresponds to BD appearing in K_3. Therefore, we add persistence interval $[3, 5]$ to $\mathrm{Dgm}(\mathcal{K})$, see Fig. 2.6 for comparison. The simplex ABD kills the cycle created when BD was added.

Our implementation of the peristent homology algorithm uses a modified version known as the persistence algorithm with a twist [7]. This version in practice is much faster in most cases [3]. Also, we store the boundary matrix D separately as sparse matrices for every dimension to reduce the necessary memory.

In the study of the persistence of self-maps it is not sufficient to bring the boundary matrix to a reduced form. We also need the bases B_i of $V_i = H(K_i)$

in which the boundary matrix is in the reduced form, as well as the matrices of the maps $w_i : H(K_i) \to H(K_{i+1})$ induced in homology by the inclusions $K_i \subset K_{i+1}$. The following proposition reduces the problem of finding bases for $V_1, \ldots, V_n$ to computing persistence diagrams.

Proposition 4.16. *The homology classes of the set of cycles in the basis B such that their corresponding intervals contain the value i constitute a basis B_i of the space $V_i = H(K_i)$.* □

Since B_i is the basis of the homology group $H(K_i)$ every homology class $[c] \in H(K_i)$ may be represented as a linear combination of the elements of B_i. The coefficient of this linear combination may be obtained by solving the respective linear system.

However, when the reduced matrix of the boundary operator R is already computed together with the matrix B, a cheaper way to obtain the coefficients is presented in Section 4.6.

Algorithm 4.6 Finding base coefficients of a cycle

```
1: function baseCoeff(cycle c, matrix R)
2:     let n be the number of columns of R
3:     let v be a vector with n entries
4:     let B be matrix obtained by modifying the identity matrix during
       reduction of the matrix R
5:     while c is a non zero cycle do
6:         i :=index of the last non-zero element in c
7:         a := −c[i]/B[i, i]
8:         c := c − a · B[., i]
9:         v[i] := a
10:    end while
11:    return v
12: end function
```

Proposition 4.17. *Section 4.6 applied to a cycle $c \in C(K)$ and the matrix B returns a vector v s.t.*

$$c = \sum_{i=1}^{n} v[i]B[., i]$$

Let G_i be a matrix representation of $w_i : V_i \to V_{i+1}$ in the bases B_i, B_{i+1}. The following algorithm computes G_i:

Proposition 4.18. *Algorithm 4.7 applied to a filtration $\mathcal{K}$ and the persistence diagram dgm of $\mathcal{K}$ returns matrices $G_1, \ldots, G_n$ of $w_1, \ldots, w_n$ in bases $B_1, \ldots, B_n$.* □

Algorithm 4.7 Computation of matrix representations of $w_1 \ldots, w_n$

```
1: function INCLUSIONSEQUENCE(diagram dgm)
2:     let n be the number of subsets in the filtration used to compute dgm
3:     for k := 1 to n do
4:         for each interval [b, d] ∈ dgm do
5:             if k ∈ [b, d] and (k + 1) ∈ [b, d] then
6:                 let c be the chain that generates interval [b, d]
7:                 let j be the index of the cycle c in B_k
8:                 let i be the index of the cycle c in B_{k+1}
9:                 G_k[i, j] := 1
10:            end if
11:        end for
12:    end for
13:    return G_1, ..., G_n
14: end function
```

4.6.1 Tower construction

Let $\mathcal{K}$ be a filtration $K_1 \subset K_2 \subset \cdots \subset K_n = K$ of a simplicial complex K with vertices in a finite set S. Let $g : S \to S$ be a map. As in Section 3.8.2 the map g may not induce a simplicial map $\kappa : K_i \to K_i$. Let $\bar{K}_i$ be the maximal subcomplex of K_i such that $\kappa|_{\bar{K}_i} : \bar{K}_i \to K_i$ is simplicial. In the last section we constructed a representation of the tower in **Vect**:

$$V_1 \xrightarrow{w_1} V_2 \xrightarrow{w_2} \ldots \xrightarrow{w_{n-1}} V_n,$$

where V_i is the homology group of K_i and w_i is the linear map between two consecutive homology groups induced by the inclusion map.

Let $\bar{\mathcal{K}}$ be a filtration of domains $\bar{K}_1 \subset \bar{K}_2 \subset \cdots \subset \bar{K}_n$ inducing the following tower in **Vect**:

$$U_1 \xrightarrow{v_1} U_2 \xrightarrow{v_2} \ldots \xrightarrow{v_{n-1}} U_n$$

where $U_i = H(\bar{K}_i)$ and v_i is the map induced in homology by the inclusion between $\bar{K}_i$ and $\bar{K}_{i+1}$. Finally, let $\varphi_i : U_i \to V_i$ be the map induced in homology by $\kappa|_{\bar{K}_i} : \bar{K}_i \to K_i$.

We show how to get matrix representation of the map φ_i from the underlying map g on vertices, persistence diagrams of $\mathcal{K}, \bar{\mathcal{K}}$ and reduced matrices $R_{\mathcal{K}}$ and $R_{\bar{\mathcal{K}}}$ of $\mathcal{K}$ and $\bar{\mathcal{K}}$.

Proposition 4.19. *Algorithm 4.8 applied to a map $g : S \to S$, persistence diagrams $dgm_{\mathcal{K}}, dgm_{\bar{\mathcal{K}}}$ together with the reduced matrices of filtrations $\mathcal{K}$ and*

Algorithm 4.8 Construction of a matrix representation of φ

```
1: function SEQUENCE(diagram dgm_K, dgm_K̄, matrix R_K, R_K̄, map g)
2:     let n be the number of subsets in filtration K
3:     for k := 1 to n do
4:         for each interval [b, d] ∈ dgm_K̄ do
5:             if k ∈ [b, d] then
6:                 let c be the cycle constructed from R_K̄ that is paired with
   interval [b, d]
7:                 c̄ = CHAINMAP(g, c)
8:                 w = BASECOEFF(c̄, R_K)
9:                 for each element w[i] do
10:                    if persistence interval of i-th cycle in R_K does not con-
   tain k then
11:                        w[i] := 0
12:                    end if
13:                end for
14:                let j be the index of the cycle c in basis of U_k
15:                Φ_k[., j] := w
16:            end if
17:        end for
18:    end for
19:    return Φ_1, ..., Φ_n
20: end function
```

$\bar{K}$ outputs matrices $\Phi_1, \ldots, \Phi_n$ of the maps $\varphi_i : U_i \to U_{i+1}$ induced by g and in the bases given by Prop. 4.16.

Proof. Construction of the matrix Φ_k is carried out column-by-column. In line 5 we check if the cycle corresponding to the interval $[b, d]$ exists in U_k. If so, we compute its image using CHAINMAP function. Next we find linear combination of the image $\bar{c}$ and zero-out cycles that do not exists in the homology group of $V_k := H(K_k)$. Finally we set the j column of Φ_k to be the vector w. □

Remark 4.20. *Notice that the cycle visible in levels k and $(k+1)$ is send to the same image, therefore one can speed up the computation storing that information. In our implementation we use auxiliary procedure to pre-compute all redundant data.*

The pseudocode to compute representation of a tower in **Pairs(Vect)** is presented in Algorithm 4.9. In lines 2-3 we call subroutine SEQUENCE

to calculate matrix representation $\Phi_1, \ldots, \Phi_n$ of $\varphi_1, \ldots, \varphi_n$ and $\Psi_1, \ldots, \Psi_n$ of $\psi_1, \ldots, \psi_n$. In line 4 we use procedure INCLUSIONSEQUENCE described in the last section to compute matrices of maps $v_1, \ldots, v_n$. To do so we need diagrams K_{dgm} of the filtration $\mathcal{K}$, $\bar{\mathcal{K}}_{dgm}$ of the filtration $\bar{\mathcal{K}}$ and reduced boundary matrices R_K and $R_{\bar{K}}$ respectively for $\mathcal{K}$ and $\bar{\mathcal{K}}$.

Algorithm 4.9 Construction of a tower in **Pairs(Vect)**

1: **function** TOWER(`diagram` $dgm_{\mathcal{K}}, dgm_{\bar{\mathcal{K}}}$, `matrix` $R_{\mathcal{K}}, R_{\bar{\mathcal{K}}}$, `map` h, g)
2: `matrix` $\Phi_1, \ldots, \Phi_n :=$ SEQUENCE$(dgm_{\mathcal{K}}, dgm_{\bar{\mathcal{K}}}, R_{\mathcal{K}}, R_{\bar{\mathcal{K}}}, h)$
3: `matrix` $\Psi_1, \ldots, \Psi_n :=$ SEQUENCE$(dgm_{\mathcal{K}}, dgm_{\bar{\mathcal{K}}}, R_{\mathcal{K}}, R_{\bar{\mathcal{K}}}, g)$
4: `matrix` $G_1 \ldots, G_n :=$ INCLUSIONSEQUENCE$(dgm_{\bar{\mathcal{K}}})$
5: **return** $\Psi_1, \ldots, \Psi_n, \Phi_1, \ldots, \Phi_n, G_1 \ldots, G_n$
6: **end function**

4.7 Eigenfunctors

Assume we have a tower $\mathcal{M}$ in **Pairs(Vect)**:

$$\begin{array}{ccccccc}
\ldots & \xrightarrow{w_{i-1}} & V_i & \xrightarrow{w_i} & V_{i+1} & \xrightarrow{w_{i+1}} & \ldots \\
 & & \varphi_i \uparrow & & \varphi_{i+1} \uparrow & & \\
\ldots & \xrightarrow{v_{i-1}} & U_i & \xrightarrow{v_i} & U_{i+1} & \xrightarrow{v_{i+1}} & \ldots \\
 & & \psi_i \downarrow & & \psi_{i+1} \downarrow & & \\
\ldots & \xrightarrow{w_{i-1}} & V_i & \xrightarrow{w_i} & V_{i+1} & \xrightarrow{w_{i+1}} & \ldots
\end{array} \tag{4.21}$$

The following algorithm computes the tower $E_\lambda(\mathcal{M})$ in **Vect** obtained by applying the eigenfunctor E_λ.

Proposition 4.22. *Algorithm 4.10 applied to sequences $\Psi_1, \ldots, \Psi_n$ of matrix representations of $\psi_1, \ldots \psi_n$, $\Phi_1, \ldots, \Phi_n$ of $\varphi_1, \ldots, \varphi_n$ and $G_1 \ldots, G_{n-1}$ of maps $v_1, \ldots, v_{n-1}$ returns the tower of eigenspaces*

$$(E_\lambda(\varphi_i, \psi_i), \delta_{\lambda,i})$$

represented by matrices $E_1, \ldots, E_n$ whose columns are bases of $E_\lambda(\varphi_i, \psi_i)$ and matrices $H_1, \ldots, H_{n-1}$ storing the maps $\delta_{\lambda,1}, \ldots, \delta_{\lambda,n-1}$ in computed bases.

Algorithm 4.10 Construction of an eigenspace tower in **Vect** from a tower in **Pairs(Vect)**

1: **function** EIGENFUNCTOR(`matrix` $\Psi_1, \ldots, \Psi_n, \Phi_1, \ldots, \Phi_n, G_1 \ldots, G_{n-1}$; `scalar` λ)
2: **for** $i = 1$ to n **do**
3: `matrix` $K_\Psi :=$ KERNEL(Ψ_i)
4: `matrix` $K :=$ KERNEL$(\Psi_i - \lambda \cdot \Phi_i)$
5: `matrix` $E'_i :=$ SPACEINTERSECTION(K_Ψ, K)
6: `matrix` $E_i :=$ QUOTIENTBASEMATRIX(K, E'_i)
7: `matrix` $H_{i-1} :=$ MATRIXRESTRICTION$(G_{i-1}, E_{i-1}, E'_{i-1}, E_i, E'_i)$
8: **end for**
9: **return** $E_1, \ldots, E_n, H_1, \ldots, H_{n-1}$
10: **end function**

Proof. Line 4 is an application of Def. 3.26 to compute the space $\bar{E}_\lambda(\varphi_i, \psi_i)$. In line 6 we compute the quotient space $E_\lambda(\varphi_i, \psi_i)$. Finally in line 7 we restrict v_{i-1} to the quotient space $E_\lambda(\varphi_{i-1}, \psi_{i-1})$ and get the matrix representation H_{i-1} of the map $\delta_{\lambda,i-1} := v_{i-1}|_{E_\lambda(\varphi_{i-1},\psi_{i-1})} : E_\lambda(\varphi_{i-1}, \psi_{i-1}) \to E_\lambda(\varphi_i, \psi_i)$. □

4.7.1 Generalized eigenfunctor

Algorithm 4.10 presented in the last section may be extended to compute the tower of generalized eigenspaces. Based on Def. 3.38 we formulate Algorithm 4.11 to compute the second level generalized eigenfunctor $E_{\lambda,2}$. The main difference with respect to Algorithm 4.10 is in solving the equation:

$$\psi_i(u_2) - \lambda\varphi_i(u_2) - \varphi_i(u_1) = 0 \tag{4.23}$$

for u_2, where u_1 is one of the solutions of

$$\psi_i(u_1) - \lambda\varphi_i(u_1) = 0. \tag{4.24}$$

Equations (4.23) and (4.24) are based on (3.42). Solution of (4.23) is given by projecting the kernel of the matrix $[\Psi_i - \lambda \cdot \Phi_i | - \Phi_i \cdot K]$ to the coordinates equivalent to u_2, where $K = \ker \Psi_i - \lambda \cdot \Phi_i$.

4.8 Matching algorithm

The final phase in the computation of the persistence of eigenspaces is an algorithm that constructs a basis for a tower in **Vect**. To present the algorithm we will need the following definitions:

Algorithm 4.11 Computation of the 2nd level generalized eigenfuctor a tower in **Pairs(Vect)**

1: **function** GENERALIZEDEIGENFUNCTOR(`matrix` $\Psi_1, \ldots, \Psi_n$, $\Phi_1, \ldots, \Phi_n$, $G_1 \ldots, G_{n-1}$, `scalar` λ)
2: **for** $i = 1$ to n **do**
3: `matrix` $K_\Psi :=$ KERNEL(Ψ_i)
4: `matrix` $K :=$ KERNEL$(\Psi_i - \lambda \cdot \Phi_i)$
5: `matrix` $K_2 :=$ KERNEL$([\Psi_i - \lambda \cdot \Phi_i | \Phi_i \cdot K])$
6: let m be the number of rows of Ψ_i
7: `matrix` $E'_i :=$ SPACEINTERSECTION$(K_\Psi, K_2[1..m, .])$
8: `matrix` $E_i :=$ QUOTIENTBASEMATRIX$(K_2[1..m, .], E'_i)$
9: `matrix` $H_{i-1} :=$ MATRIXRESTRICTION$(G_{i-1}, E_{i-1}, E'_{i-1}, E_i, E'_i)$
10: **end for**
11: **return** $E_1, \ldots, E_n, H_1, \ldots, H_{n-1}$
12: **end function**

Definition 4.25. A *pivot position* of a non-zero vector is the position of the first non-zero element of this vector. It is undefined for the zero vector.

Definition 4.26. A matrix M is in the *column echelon* form if for any two consecutive columns c_i and c_{i+1} of M if $c_{i+1} \neq 0$ then $c_i \neq 0$ and the pivot position of c_{i+1} is greater than the pivot position of c_i.

Example 4.27. The following matrices are in the column echelon form:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 2 & 1 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}.$$

The following matrices are not in the column echelon form:

$$\begin{bmatrix} 1 & 0 & 0 \\ 2 & 0 & 3 \\ 1 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}.$$

The next proposition is the consequence of the column echelon form definition.

Proposition 4.28. *Right-to-left column addition in a matrix M that is in column echelon form does not violate the column echelon form of M.*

We recall the theorem stated in the last chapter:

Theorem. 3.15. Any tower of vector spaces admits a matching basis.

Proof. Let $\mathcal{X} = (X_i, \xi_i)$ be a tower in **Vect**. Without loss of generality we assume that for $i \leq 0$ and $i > n$ $X_i = 0$. Let $m_i = \dim X_i$ and let B_i be a matrix whose columns constitute a basis of X_i. Let A_i be the matrix of ξ_i in the bases B_i and B_{i+1}.In order to prove Thm. 3.15 it is sufficient to show that a suitable change of bases B_i brings matrices A_i to matching form. This is achieved by Algorithm 4.12 which transforms matrices $A_1, \ldots, A_{n-1}$ to matching form while keeping track of changes in the bases $B_1, \ldots, B_n$.

Algorithm 4.12 Matching algorithm

1: **function** MATCHING(`matrix` $B_1, \ldots, B_n; A_1, \ldots, A_{n-1}$)
2: **for** $i := (n-1)$ to 1 **do**
3: Bring A_i to the column echelon form using elementary column operations on A_i. Update B_i using elementary column operations and A_{i-1} using elementary row operations.
4: **end for**
5: **for** $i := 1$ to $(n-1)$ **do**
6: Bring A_i to the matching form using elementary row operations on A_i. Update B_{i+1} using elementary column operations and A_{i+1} using elementary column operations.
7: **end for**
8: **return** $B_1, \ldots, B_n; A_1, \ldots, A_{n-1}$
9: **end function**

The algorithm is divided into two phases:
In Phase 1 we transform A_i to the column echelon form using elementary column operations with decreasing i. To obtain the column echelon form we use a Gaussian elimination method. We first clear the first row using the first column, then the second row using second column and so on. We call the collection of the pivot positions in columns the *staircase*. In the last step we multiply columns, so that all entries in the staircase are equal to 1. Column operations on A_i are mirrored in B_i and performed on row space of A_{i-1}. We do not modify matrices A_k with $k > i$, therefore the column-echelon form of matrices A_k with $k > i$ is kept.
In Phase 2 we modify A_i to a matching form with increasing i. We use elementary row operations to zero-out the elements below the staircase. Observe that when we add the kth row to the lth row in A_i and $k < l$ then the the corresponding operation in the base B_{i+1} is to subtract lth column from the kth one. The same column operation is applied to A_{i+1} and from Proposition 4.28 we have that column echelon form of A_{i+1} is not violated.

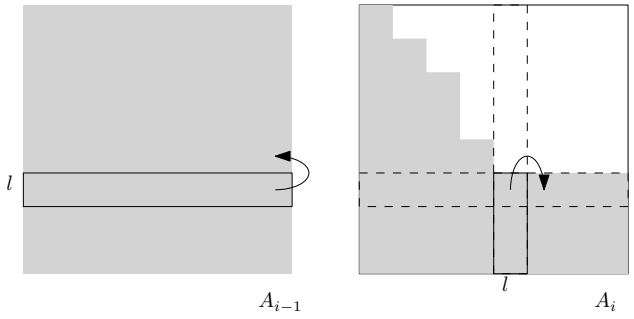


Figure 4.3: Whenever we do a left-to-right column operations on a matrix A_i, we do the same operation on columns of B_i. However in A_{i-1} we apply a corresponding top-to-bottom row operation.

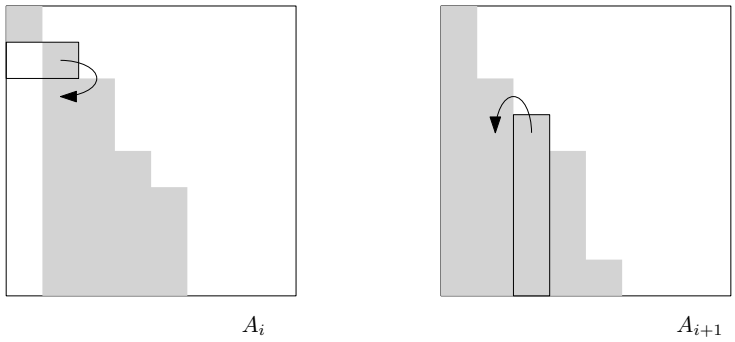


Figure 4.4: In Phase 2 when we perform a bottom-to-down row operation in A_i for example we add the second row to the third one, we have to modify the matrix A_{i+1} by subtracting the third column from the second column. In consequence, the column echelon form is not violated for A_{i+1}.

The overview of Phase 1 and Phase 2 is presented in Fig. 4.5. □

The form of the matrices $A_1, \ldots, A_n$ allows to extract the persistence intervals. Observe that in the matching form basis element is mapped either to 0 or to an element of another basis. We extract the persistence intervals by "following" basis element until it is mapped to 0.

Lemma 4.29. *Algorithm 4.13 applied to matrices $A_1, \ldots, A_n$ in the matching form returns the persistence diagram of a tower $\mathcal{X} := (X_i, \xi_i)$ in **Vect** s.t. A_i is the matrix of ξ_i.*

Proof. Assume B_t is the basis of X_t. Let b be the jth element of the basis B_t. We mark the column $A_t[., j]$ if the pre-image $(\xi_{t-1})^{-1}(b)$ is non-empty. For $k < l$ let $\xi_k^l := \xi_l \circ \cdots \circ \xi_k$. We start with an observation that a persistence

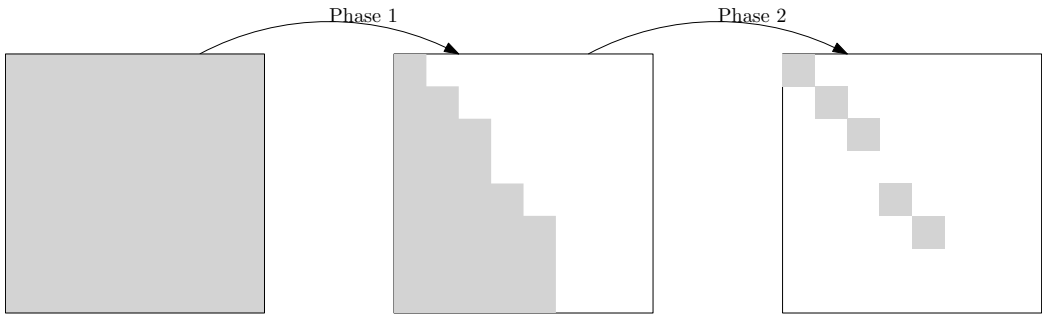


Figure 4.5: Two phases of the matching algorithm shown on a matrix A_i. First we bring the matrix A_i into the column echelon form, and then to the matching form.

Algorithm 4.13 Extracting persistence diagram

```
1: function DIAGRAM(matrix A_1, ..., A_n)
2:     let dgm be an empty persistence diagram
3:     let all columns of A_1, ..., A_n be unmarked
4:     for k := 1 to n do
5:         for each unmarked column A_k[., j] of A_k do
6:             l := k
7:             while A_l[., j] is non zero do
8:                 if l ≠ k then
9:                     mark column A_l[., j]
10:                end if
11:                let i be such that A_l[i, j] = 1
12:                j := i
13:                l := l + 1
14:            end while
15:            add [k, l] to dgm
16:        end for
17:    end for
18:    return dgm
19: end function
```

interval $[k, l]$ in the persistence diagram of $\mathcal{X}$ implies the existence of $b \in B_k$ s.t. $(\xi_{k-1})^{-1}(b) = 0$, $\xi_k^l(b) = 0$ and $\xi_k^s(b) \neq 0$ for $k < s < l$. In Line 5 we iterate over all columns of A_k to find elements of B_k with an empty pre-image. Let b be the element fulfilling this condition. It induces a persistence interval with an end in k. Then in Lines 6-14 we compute the minimal l s.t

$\xi_k^l(b)$ is zero. We mark the column $A_l[.,j]$, since the jth element of the base B_l has a non-empty pre-image. □

Example 4.30. Assume:

$$A_1 := \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, A_2 := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, A_3 := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Let A_1, A_2, A_3 be matrices of maps ξ_1, ξ_2, ξ_3 of a tower $\mathcal{X} = (X_i, \xi_i)$ The persistence diagram of $\mathcal{X}$ consists of intervals $[1,3], [1,2], [2,3]$.

Observe that we can stop computation after Phase 1 and extract the persistence intervals from modified matrices A_i. Phase 2 is necessary to compute the bases of the tower.

Proposition 4.31. *The running time of Algorithm 4.12 is $O(nm^3)$ where m is the maximum of dimensions of matrices A_i.*

Proof. Phase 1 can be divided into n subproblems of bringing A_i into the column echelon form. Assume A_i is a $m \times m$ matrix with full rank. Computing the column echelon form requires $O(m^3)$ steps: ith column has to be used $m - i$ times to zero-out all elements lying on the right side of the pivot element. Every column addition requires pessimistically m operations. Phase 2 also requires $O(nm^3)$ operations. We can divide it into n subproblems and every subproblem needs pessimistically the same number of steps as Phase 1. □

4.9 Persistent homology of a self map

The computation of λ-persistence diagram for a set of points S, approximation $g : S \to S$ and a value of λ is presented in Algorithm 4.14.

Theorem 4.32. *Algorithm 4.14 applied to a set of points $S \subset \mathbb{R}^k$ and a map $g : S \to S$ computes the λ-eigenvalue persistence diagram of a map g.*

Proof. Algorithm 4.14 is an implementation of the approach described in Section 3.8.2. First we compute filtration $\mathcal{K}$ on points S, then the filtration of domains $\bar{\mathcal{K}}$. Next, we apply the reduction algorithm to compute the persistence diagrams. In the third step, we use Prop. 3.57 to compute the tower in **Pairs(Vect)** of homology of the pairs ι_i, κ_i', where $\iota_i : \bar{K}_i \to K_i$ is induced by the inclusion and $\kappa_i' : \bar{K}_i \to K_i$ is induced by g. Then, the eigenfunctor is applied and finally the basis using Algorithm 4.12 is computed. Finally the persistence diagram is extracted from matrices $A_1, \ldots, A_n$. □

Algorithm 4.14 Main algorithm

1: **function** MAIN(`points` S, `map` g, `scalar` λ)
2: let $id : S \to S$ be an identity map on points S
3: `filtration` $\mathcal{K}$:= VIETORISRIPSFILTRATION(S)
4: `filtration` $\bar{\mathcal{K}}$:= DOMFILTRATION($\mathcal{K}, g$)
5: `diagram` $dgm_{\mathcal{K}}$,`matrix` $R_{\mathcal{K}}$:= REDUCE($\mathcal{K}$)
6: `diagram` $dgm_{\bar{\mathcal{K}}}$,`matrix` $R_{\bar{\mathcal{K}}}$:= REDUCE($\bar{\mathcal{K}}$)
7: `matrix` $\Psi_1, \ldots, \Psi_n, \Phi_1, \ldots, \Phi_n, G_1 \ldots, G_n$:= TOWER($dgm_{\mathcal{K}}, dgm_{\bar{\mathcal{K}}}, R_{\mathcal{K}}, R_{\bar{\mathcal{K}}}, id, g$)
8: `matrix` $E_1, \ldots, E_n, A_1, \ldots, A_{n-1}$:= EIGENFUNCTOR($\Psi_1, \ldots, \Psi_n, \Phi_1, \ldots, \Phi_n, G_1 \ldots, G_{n-1}, \lambda$)
9: `matrix` $E_1, \ldots, E_n, A_1, \ldots, A_{n-1}$:= MATCHING($E_1, \ldots, E_n, A_1, \ldots, A_{n-1}$)
10: `diagram` dgm := DIAGRAM($A_1, \ldots, A_{n-1}$)
11: **return** dgm
12: **end function**

Computation of generalized λ-persistence diagram is similar to Algorithm 4.14, but with the call to the function GENERALIZEDEIGENFUNCTOR in line 8.

Chapter 5

Stability and convergence results

In this chapter we give conditions under which we can infer the correct dimension of the eigenspace of a map given only an approximation of the map. The material of this chapter is presented in detail in [14] therefore we skip the proofs.

Given two subsets X, Y of $\mathbb{R}^d$ the *Hausdorff distance* $d_H(X, Y)$ is defined by:

$$d_H(X, Y) := \max\{\sup_{x \in X} \inf_{y \in Y} \|x - y\|, \sup_{y \in Y} \inf_{x \in X} \|x - y\|\}$$

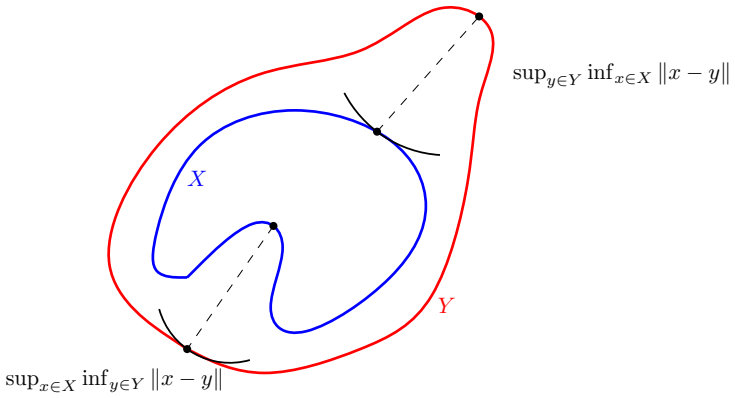


Figure 5.1: Dashed lines represent distances from the definition of the Hausdorff distance. The Hausdorff distance is the maximal one.

Given a subset $A \subset \mathbb{R}^l$, the distance function $d_A : \mathbb{R}^l \to \mathbb{R}$ maps each point in $\mathbb{R}^l$ to its distance to the set A, i.e.

$$d_A(x) := \inf_{y \in A} \|x - y\|.$$

We define distance function for the graph $Gf \subset \mathbb{R}^l \times \mathbb{R}^l$ of $f : \mathbb{R}^l \to \mathbb{R}^l$ in the same way and denote it d_{Gf}.

For a distance function $d_A : \mathbb{R}^l \to \mathbb{R}$ we define *sublevel sets*:

$$A_r := d_R^{-1}([0, r]).$$

The *homological critical value* of the distance function is every value at which the homology of the sublevel sets change. We say a distance function is *tame* if every sublevel set has finite-dimensional homology groups and there is only finitely many homological critical values. In the sequel we assume that all the considered distance functions are tame. In particular, this assumption allows us to define the homological feature size. The *homological feature size* $\mathrm{hfs}(A)$ of A is is the smallest homological critical value of the distance function d_A.

Let $\mathbb{X} \subset \mathbb{R}^l$ and $f : \mathbb{X} \to \mathbb{X}$ be a continuous self function. Moreover, let $S \subset \mathbb{X}$ be a finite and $g : S \to S$ be an approximate sampling of f. By considering the maximum distance in $\mathbb{R}^l \times \mathbb{R}^l$ we ensure that:

$$d_H(\mathbb{X}, S) \leq d_H(Gf, Gg)$$

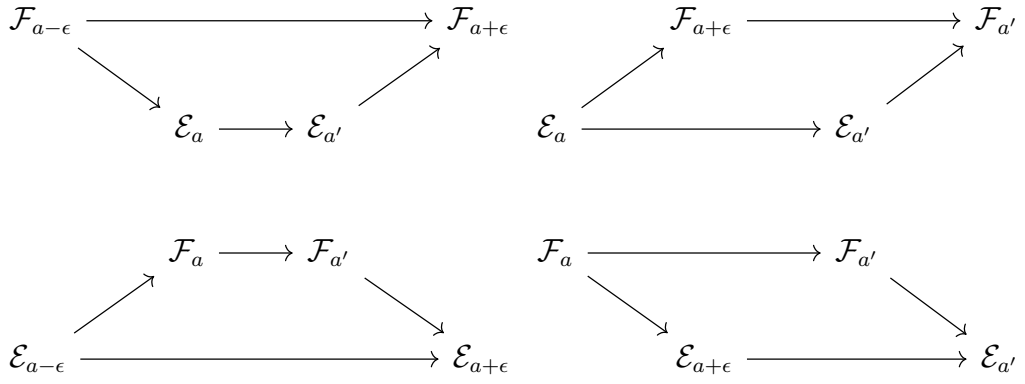
If $\epsilon \geq d_H(Gg, Gf) \geq d_H(\mathbb{X}, S)$ then:

$$\mathbb{X}_r \subset S_{r+\epsilon} \subset X_{r+2\epsilon} \tag{5.1}$$

$$Gf_r \subset Gg_{r+\epsilon} \subset Gf_{r+2\epsilon} \tag{5.2}$$

5.1 Interleaving

A concept similar to towers in categories was studied in [6]. The *persistence module* $\mathcal{F}$ is the family $\{F_\alpha\}_{\alpha \in A}$ of vector spaces indexed by $A \subset \mathbb{R}$ together with a family of homomorphisms $\{f_\alpha^\beta : F_\alpha \to F_\beta\}$ such that $\forall \alpha < \beta < \gamma$, $f_\alpha^\gamma = f_\beta^\gamma \circ f_\alpha^\beta$ and $f_\alpha^\alpha = id_{F_\alpha}$. We say that two persistence modules $\mathcal{F}$ and $\mathcal{E}$ are *strongly ϵ-interleaved* if the following diagrams commute:



Note that, for a tame continuous function f we have a finite number of critical values. Therefore, there is a finite number of distinct homology groups of the sublevel sets of the distance function d_{Gf}. Hence, the tower in **Vect** constructed from those groups is the persistence module indexed by a subset of $\mathbb{N}$.

Necessary condition for the stability and convergence theorems are provided in the next lemma. Assume $\mathbb{U}, \mathbb{V} \subset \mathbb{R}^l$, $h : \mathbb{U} \to \mathbb{U}$ and $k : \mathbb{V} \to \mathbb{V}$ are continuous self maps, $\epsilon := d_H(Gh, Gk)$. Choose $r' \in \mathbb{R}$ such that $r + \epsilon \leq r'$, then $Gh_r \subset Gk_{r'}$. Let $p_r : Gh_r \to \mathbb{U}_r, p_{r'} : Gk_{r'} \to \mathbb{V}_r$ and $q_r : Gh_r \to \mathbb{U}_r, q_{r'} : Gh_{r'} \to \mathbb{V}_{r'}$ be respectively the projections on the first and the second coordinate. The following diagram is an arrow in **Pairs(Top)**:

$$\begin{array}{ccccc} \mathbb{U}_r & \xleftarrow{p_r} & Gh_r & \xrightarrow{q_r} & \mathbb{U}_r \\ \downarrow & & \downarrow & & \downarrow \\ \mathbb{V}_{r'} & \xleftarrow{p_{r'}} & Gk_{r'} & \xrightarrow{q_{r'}} & \mathbb{V}_{r'} \end{array} \tag{5.3}$$

After application of the homology functor we get an arrow in **Pairs(Vect)**:

$$\begin{array}{ccccc} H(\mathbb{U}_r) & \xleftarrow{\varphi_r} & H(Gh_r) & \xrightarrow{\psi_r} & H(\mathbb{U}_r) \\ \downarrow & & \downarrow & & \downarrow \\ H(\mathbb{V}_{r'}) & \xleftarrow{\upsilon_{r'}} & H(Gk_{r'}) & \xrightarrow{\nu_{r'}} & H(\mathbb{V}_{r'}) \end{array} \tag{5.4}$$

And finally, when applying eigenfunctor we get the arrow in **Vect**:

$$E_\lambda(\varphi_r, \psi_r) \to E_\lambda(\upsilon_{r'}, \nu_{r'}).$$

The following lemma is useful in proving that towers induced by functions h and k are strongly ϵ-interleaved:

Lemma 5.5 (Interleaving lemma, [14])**.** *Let $\mathbb{U}, \mathbb{V} \subset \mathbb{R}^l$, and $h : \mathbb{U} \to \mathbb{V}, k : \mathbb{V} \to \mathbb{V}$ be such that the associated distance functions are tame. Set $\epsilon := d_H(Gh, Gk)$. If $a + \epsilon \leq b \leq c \leq d - \epsilon$, then*

$$\begin{array}{ccc} E_\lambda(\varphi_a, \psi_a) & \longrightarrow & E_\lambda(\varphi_d, \psi_d) \\ \downarrow & & \uparrow \\ E_\lambda(\upsilon_b, \nu_b) & \longrightarrow & E_\lambda \upsilon_c, \nu_c) \end{array} \tag{5.6}$$

commutes. If $a + \epsilon \leq b \leq c$ and $a \leq d \leq c - \epsilon$, then the following diagram

commutes:

$$\begin{array}{ccc} E_\lambda(\varphi_a, \psi_a) & \longrightarrow & E_\lambda(\varphi_d, \psi_d) \\ \downarrow & & \downarrow \\ E_\lambda(\upsilon_b, \nu_b) & \longrightarrow & E_\lambda(\upsilon_c, \nu_c) \end{array} .$$

5.2 Convergence

The first result is about the quality of an output from the algorithm given some information about quality of an approximation of a map. Let $f : \mathbb{X} \to \mathbb{X}$ be a continuous self map we are interested in, $S \subset \mathbb{X}$ be a finite sample with a map $g : S \to S$, which is some approximation of f. Let $X := H(\mathbb{X})$ be the homology group of $\mathbb{X}$, and $\phi : X \to X$ be the endomorphism induced by the map f in homology. As previously we define projections: $\varphi_r, \psi_r : H(Gg_r) \to H(S_r)$. We are interested in the invariants of the map induced between sublevel sets of Gg at two different thresholds r and r':

$$\epsilon_{\lambda,r}^{r'} : E_\lambda(\varphi_r, \psi_r) \to E_\lambda(\varphi_{r'}, \psi_{r'})$$

Theorem 5.7 (Inference theorem, [14])**.** *Let $\mathbb{X} \subset \mathbb{R}^l$ be a compact absolute neighbourhood retract*[1]*, $S \subset \mathbb{X}$, and let $f : \mathbb{X} \to \mathbb{X}$, be a map such that the distance functions for $\mathbb{X}$ and Gf are tame. Then any map $g : S \to S$ satisfies*

$$\dim E_\lambda(\phi) = \operatorname{rank} \epsilon_{\lambda,\varepsilon}^{3\varepsilon}, \tag{5.8}$$

for all $d_H(Gf, Gg) < \varepsilon < \frac{1}{4} \min\{\mathrm{hfs}(\mathbb{X}), \mathrm{hfs}(Gf)\}$.

Note that, computing the $\operatorname{rank} \epsilon_{\lambda,\varepsilon}^{3\varepsilon}$ is equivalent to finding the persistence intervals lasting from ε to 3ε. Hence, by computing the λ-persistence interval we can recover the dimension of eigenspace of the map induced in homology by the map f.

5.3 Stability

The following theorem is a generalization of Thm. 2.22:

Theorem 5.9 (Strong Stability Theorem, [9])**.** *Let $\mathcal{E}$ and $\mathcal{F}$ be two towers in* ***Vect****. If $\mathcal{F}$ and $\mathcal{E}$ are strongly ϵ-interleaved, then $d_B(\mathrm{Dgm}(\mathcal{F}), \mathrm{Dgm}(\mathcal{E})) \leq \epsilon$.*

[1] the definition of absolute neighbourhood can be found in [26, p. 56]

The second theorem is a counterpart of Thm. 2.22 in the setting of a self map, i.e. we state that if maps are similar, then the corresponding λ-persistence diagrams are also similar with respect to the bottleneck distance.

Let $h : \mathbb{U} \to \mathbb{U}$ and $k : \mathbb{V} \to \mathbb{V}$ be two self-maps on $\mathbb{U}, \mathbb{V} \subset \mathbb{R}^l$. Let Gh and Gk be their graphs. Let $\mathcal{E}_\lambda$ and $\mathcal{F}_\lambda$ be the towers in **Vect** of eigenspaces for an eigenvalue $\lambda \in \mathbb{F}$ respectively for h and k, then following theorem is true:

Theorem 5.10 (Stability theorem, [14])**.** *Let $\mathbb{U}, \mathbb{V} \subset \mathbb{R}^l$ and $h : \mathbb{U} \to \mathbb{U}, k : \mathbb{V} \to \mathbb{V}$ such that their associated distance functions are tame. Then*

$$d_B(\mathrm{Dgm}(\mathcal{E}_\lambda), \mathrm{Dgm}(\mathcal{F}_\lambda)) \leq d_H(Gh, Gk).$$

Chapter 6

Numerical experiments

In this chapter we present four numerical experiments. The first two examples are self-maps of the unit circle $\mathbb{S}^1$ in the complex plane $\mathbb{C}$. The third example is a collection of three maps acting on a 2-dimensional torus. The last example concerns a map on a wedge of circles. The aim is to test whether studying the persistence diagrams of eigenspaces and generalized eigenspaces obtained from a finite sample suffices to recover the eigenspaces and generalized eigenspaces of the original map.

In all examples we first choose a finite sample from the domain of the map. Depending on the type of experiment we add some noise. We denote the set of sampled points by S. Then, we compute the value of the map on elements of S. Every value not in S is replaced by the point in S that is closest to the actual value. The respective algorithm is presented as Algorithm 6.1.

We work over finite field $\mathbb{Z}_{1009}$. It is big enough to capture the behaviour of our examples and and the computations over $\mathbb{Z}_{1009}$ are not time consuming.

All experiments are based on author's implementation of the algorithms described in Chapter 4.

6.1 Expansion map

As our first example we study the map $f : \mathbb{S}^1 \ni z \mapsto z^2 \in \mathbb{S}^1$. We represent points in $\mathbb{S}^1$ as points in $\mathbb{R}^2$. The circle has only one homology generator in the first dimension and since the map doubles the angle of points, the homology generator is doubled. Therefore, the only eigenvalue in dimension one is 2.

In the first stage we choose $m = 100$ equidistant points on the unit circle and we add Gausian noise with 2D kernel of standard deviation equal to $\sigma \in [0, 0.30]$. Let's denote this set by S. We vary the level σ of noise, in

Algorithm 6.1 Approximation algorithm for a function f on points S

```
1: function PREPAREDATA(f, S)
2:     Let g : S → S be a map
3:     for each p ∈ S do
4:         q := f(p)
5:         let q̄ be a randomly chosen point in S
6:         for each r ∈ S do                ▷ Find nearest point to q in S
7:             if dist(r, q) < dist(q̄, q) then
8:                 q̄ := r
9:             end if
10:        end for
11:        g(p) := q̄
12:    end for
13:    return g
14: end function
```

order to be able to inspect several cases. We then use Algorithm 6.1 to approximate values of f on the points S. Denote the map that approximates f by $g : S \to S$. The triple (g, S, λ) constitutes the input to Algorithm 4.14.

The running times of our algorithm averaged from 10 runs on a laptop with 32GB RAM and Intel processor with 4 dual 2.4GHz cores are presented in Table 6.1. The bottleneck of our algorithm is the size of the Vietoris–Rips complex and consequently the size of the boundary matrices.

# of points	Phase 1	Phase 2	Phase 3	Phase 4
60	0.5	5.1	0.01	0.01
80	1.4	24	0.01	0.01
100	3.2	71.5	0.02	0.01

Table 6.1: Running time in seconds of Algorithm 4.14 for $\lambda = 2$ and S consisting of 60, 80 and 100 points. Phase 1 is construction of the Vietoris–Rips complex. Phase 2 is reduction of the boundary matrices. Phase 3 consists of computing towers in **Vect** and the application of the eigenspace functor. Phase 4 is the matching algorithm.

Results. We compute the λ-persistence diagram of g for every value in the field $\mathbb{Z}_{1009}$. The results are presented in Fig. 6.1. As expected, we see the dominating interval for eigenvalue 2 for all σ's, except for $\sigma = 0.24$ where it vanishes due to the noise. When σ is 0.15 or more, which means most of the points lie in the strip that is equal roughly to one third of the radius, we see

intervals in diagrams for any λ. This phenomenon is known for the pencils of matrices[8][8.3]. It is caused by the high noise.

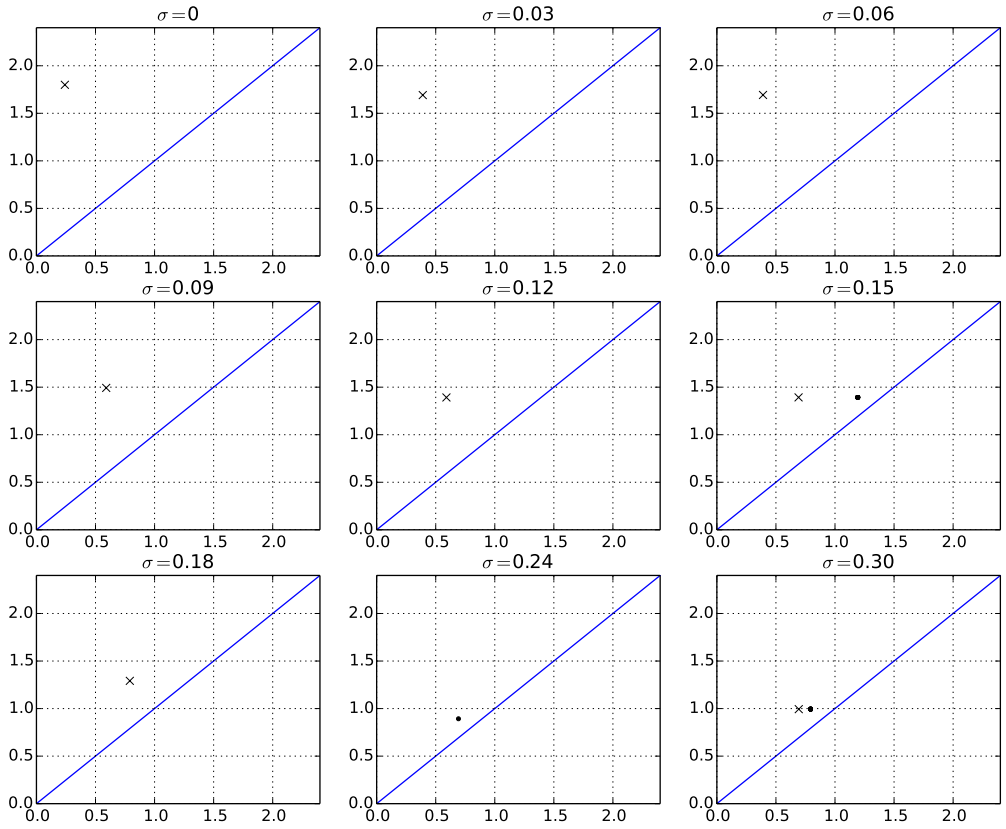


Figure 6.1: The persistence diagrams of $f(z) : \mathbb{S}^1 \ni z \mapsto z^2 \in \mathbb{S}^1$ in the first homology level, with varying noise σ. Crosses show the persistence intervals for eigenvalue $\lambda = 2$ and dots represent intervals visible for all values of $\mathbb{Z}_{1009}$. The x-axis is the birth of a homology generator and y-axis is the death.

6.2 Reflection map

In the second experiment we keep the space $\mathbb{S}^1$ and we replace the map by

$$f : \mathbb{S}^1 \ni z \mapsto \bar{z} \in \mathbb{S}^1,$$

which is the reflection across the x-axis. In this map the unique first homology generator is send to itself but with the opposite orientation. In consequence, the only eigenvalue is $\lambda = -1$. We compute similarly λ-persistence diagrams for every $\lambda \in \mathbb{Z}_{1019}$ on a finite sample and Gaussian noise with standard deviation $\sigma \in [0, 0.24]$. The only significant interval is the one for eigenvalue $\lambda = -1$ in all persistence diagrams.

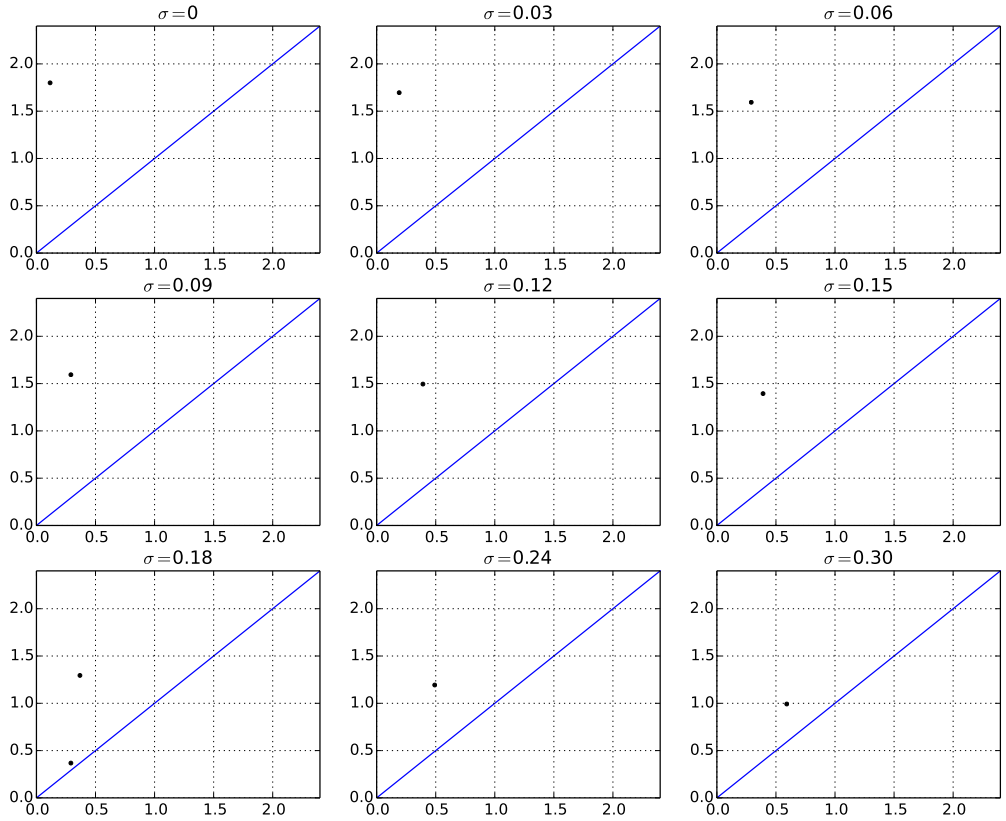


Figure 6.2: λ-persistence diagrams of the map $f(z) := \bar{z}$ for different noise levels. Dots represent intervals of the $\lambda = -1$ eigenavlue.

6.3 Torus maps

In the third experiment we study three maps on a torus. We represent the torus $\mathbb{T}$ as the square $[0,1]^2$ with identified opposite edges as in Fig. 6.3. Then, every point on $\mathbb{T}$ can be described as a pair of two numbers: $(x,y) \in [0,1]^2$. The distance function induced after gluing edges is:

$$dist((x_1,y_1),(x_2,y_2))) := \sqrt{\mu(x_1-x_2)^2+\mu(y_1-y_2)^2},$$

where, $\mu(t) := \min(|t|,|1-t|)$. $\mathbb{T}$ may also be viewed as $\mathbb{R}/_{\sim}$, where $(x_1,y_1) \sim (x_2,y_2)$ if and only if $x_1-x_2 \in \mathbb{Z}$ and $y_1-y_2 \in \mathbb{Z}$.

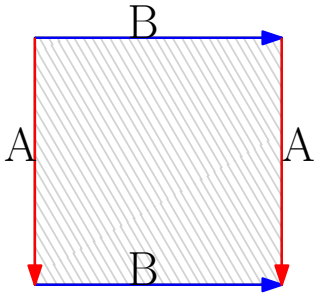


Figure 6.3: Torus representation as a square with opposite edges A and B identified. Arrows show the direction of glueing.

A 2-by-2 integer matrix A induces a map $\bar{A} : \mathbb{T} \to \mathbb{T}$ definedby $\bar{A}([x]_{\sim}) := [Ax]_{\sim}$. We study the eigenspace towers for maps induced by the following three matrices:

$$A_1 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, A_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, A_3 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}. \tag{6.1}$$

It is easy to show that the maps induced in the first homology have the same matrix if we choose the basis properly. For example A_1 doubles both generators. In consequence, A_1 has only one distinct eigenvalue $\lambda = 2$ with two different eigenvectors. The map induced by A_2 has two eigenvalues $\lambda = 1$ and $\lambda = -1$ and two eigenvectors: one respectively for every eigenvalue. Finally, A_3 has only one eigenvector for $\lambda = 1$ and one generalized eigenvector for $\lambda = 1$, which is visible in the Jordan form of A_3.

For every matrix A_i with $i \in \{1,2,3\}$ we generate g_i the approximation of $\bar{A}_i$ on the set S of uniformly drown 100 point in $\mathbb{T}$. The outcome of Algorithm 4.14 applied to S and g_i is presented in Fig. 6.4, Fig. 6.5 and Fig. 6.6. For g_1 we have two dominating intervals for eigenvalue $\lambda = 2$ corresponding to two expected eigenvectors. In comparison to the previous example we get

higher number of short intervals. But they are easily distinguishable from the expected ones. In the λ-persistence diagram of g_2 we have again two important intervals, this time for $\lambda = 1$ and $\lambda = -1$. This agrees with the expectations. For g_3 we have only one significant interval for an eigenvalue $\lambda = 1$ in a λ-persistence diagrams. However, there are two persistence intervals in 1-generalized persistence diagram. The one not visible in 1-persistence diagram is generated by the generalized eigenvector for eigenvalue 1.

6.4 Figure eight map

In our last example we study a map on the wedge of two circles. Let S_p be a unit circle centered at the point $p \in \mathbb{R}^2$. Let $a := (-1, 0) \in \mathbb{R}^2$ and $b := (1, 0) \in \mathbb{R}^2$. We denote the wedge of circles $S_a \cup S_b$ with the letter $\mathbb{E}$. Let $S_p(\alpha)$ denote the point on S_p with polar angle α in polar coordinate system in which p is the pole, and the pole axis is parallel to the OX-axis and facing the same direction. Two examples of points in $\mathbb{E}$ are shown in Fig. 6.7. Let the map $f : \mathbb{E} \to \mathbb{E}$ be defined as follows:

$$f(x) := \begin{cases} S_a(2\alpha) \text{ if } x = S_a(\alpha) \text{ and } \alpha \in [0, \pi) \\ S_b(3\pi - 2\alpha) \text{ if } x = S_a(\alpha) \text{ and } \alpha \in [\pi, 2\pi) \\ x \text{ if } x \in S_b \end{cases} \tag{6.2}$$

Intuitively the map f:

- on the upper left half it is defined as function $z \mapsto z^2$ in the complex numbers with origin in the middle of the left circle(Fig. 6.8)
- stretches the lower left half to the right circle (Fig. 6.8)
- leaves the right circle untouched.

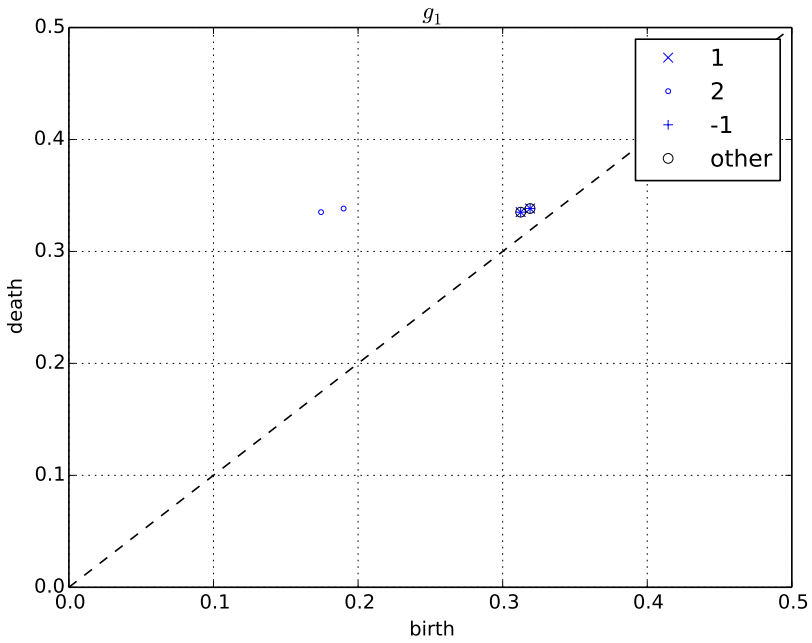
The space has two homology generators in the first dimension: A and B representing respectively the left circle and the right one. Then A is sent to $A+B$ and B is sent to itself by map induced in homology. Therefore, the matrix of the map induced in homology in properly chosen bases is:

$$M = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

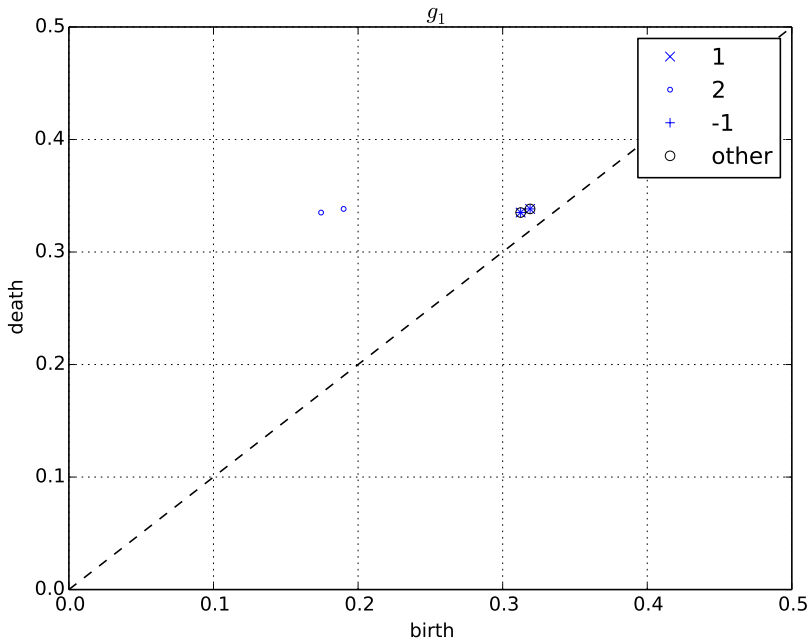
The matrix M is in a Jordan normal form, so we can deduce there is only one eigenvalue $\lambda = 1$. Moreover, there is one ordinary eigenvector and one generalized eigenvector for the eigenvalue 1.

By selecting 50 equidistant points on the left and right circle we discretize $\mathbb{E}$. Next we add a Gaussian noise to the points. An approximation of the map is defined as a point closest to the original image. We repeat the above procedure for the Gaussian noise with zero mean and variance equal to $0, 0.03,\ 0.06,\ 0.09$. Finally we build Vietoris–Rips filtration and compute λ-persistence and generalized λ-persistence diagrams for $k = 2$. In figures 6.9(b), 6.10(b), 6.11(b) and 6.12(b) we show generalized λ-persistence diagrams. Figures 6.9(a), 6.10(a), 6.11(a) and 6.12(a) show λ=persistence diagrams.

In each case, there is one interval in 1-persistence diagrams, and two intervals in generalized 1-persistence diagrams. Moreover the diagrams of eigenspaces are subsets of the generalized eigenspace which is the consequence of the fact that eigenvectors are also generalized eigenvectors. No significant intervals are visible for $\lambda \neq 1$.

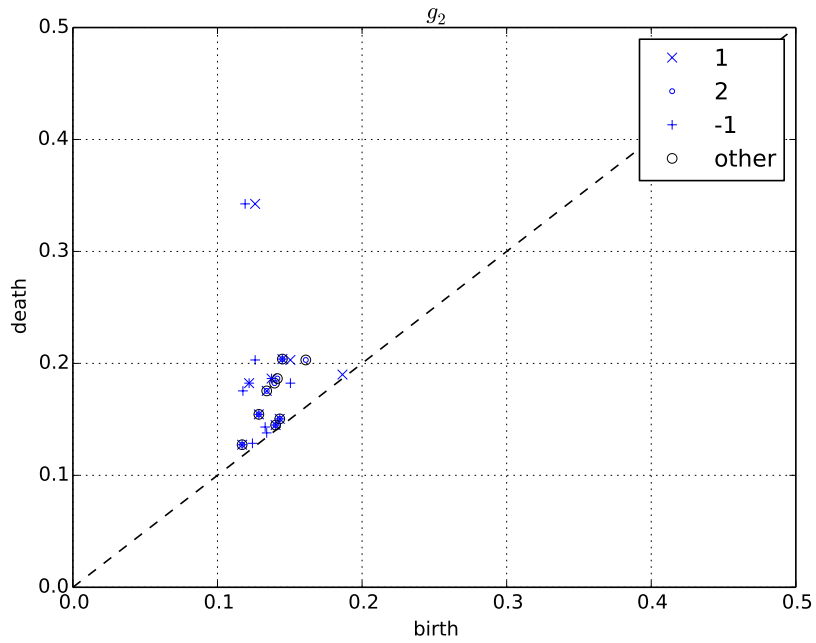


(a) λ-persistence diagrams

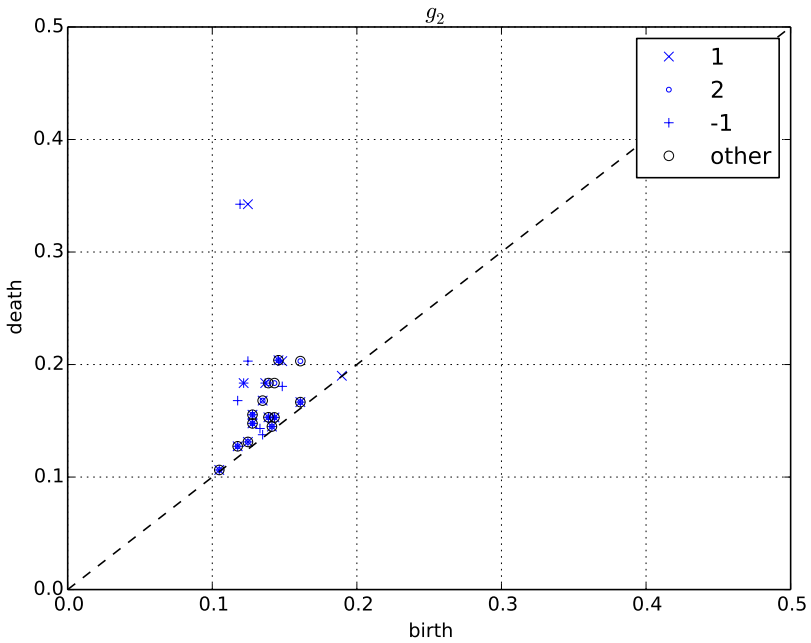


(b) Generalized λ-persistence diagrams

Figure 6.4: The 1-dimensional persistence diagrams of towers of eigenspaces of g_1.

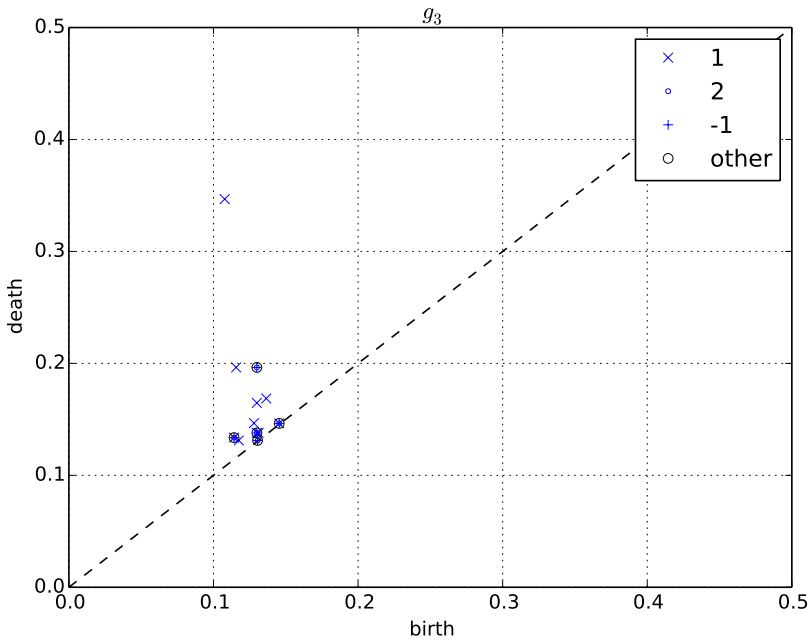


(a) λ-persistence diagrams

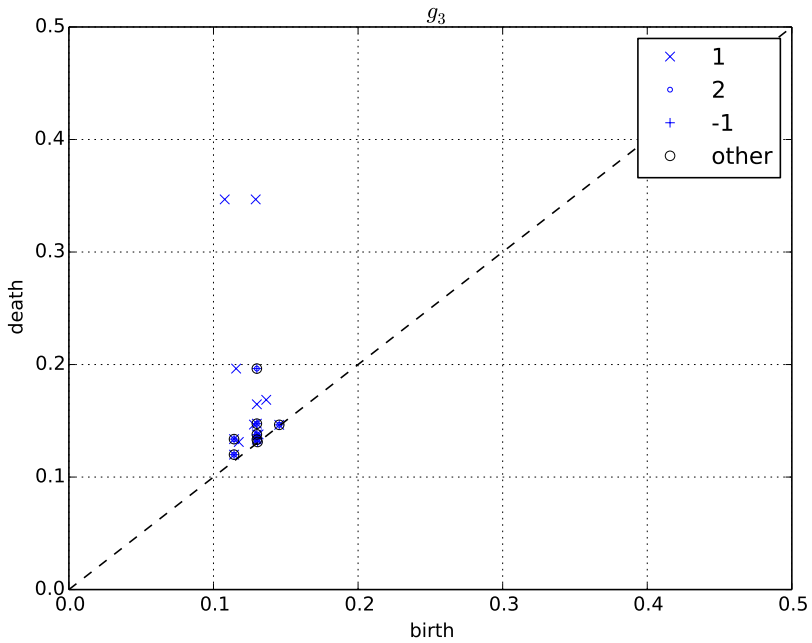


(b) Generalized λ-persistence diagrams

Figure 6.5: The 1-dimensional persistence diagrams of towers of eigenspaces of g_2.



(a) λ-persistence diagrams



(b) Generalized λ-persistence diagrams

Figure 6.6: The 1-dimensional persistence diagrams of towers of eigenspaces of g_3.

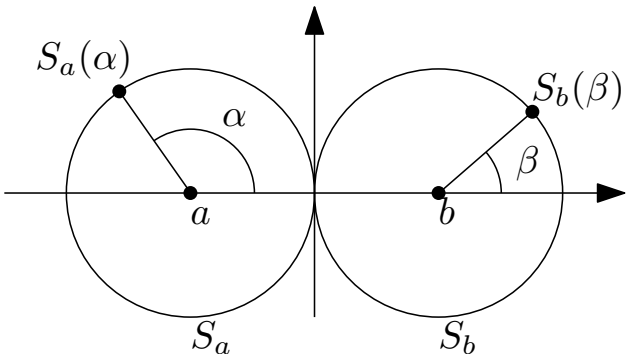


Figure 6.7: A plot of the space $\mathbb{E}$. The points $S_a(\alpha)$ and $S_b(\beta)$ are presented together with the defining angles α and β.

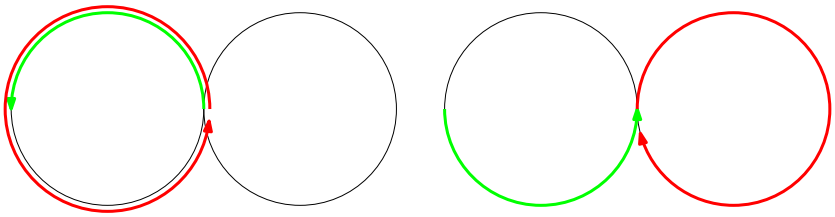
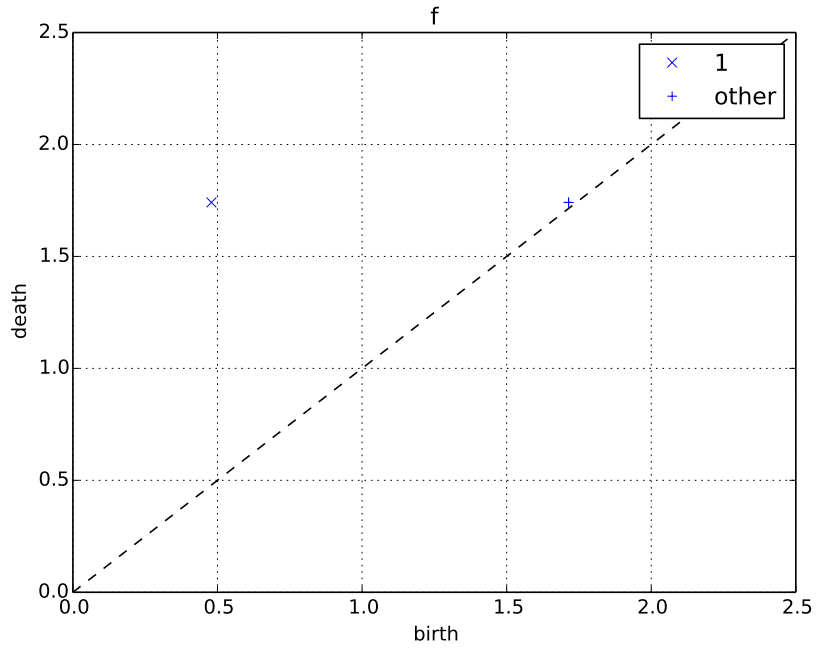
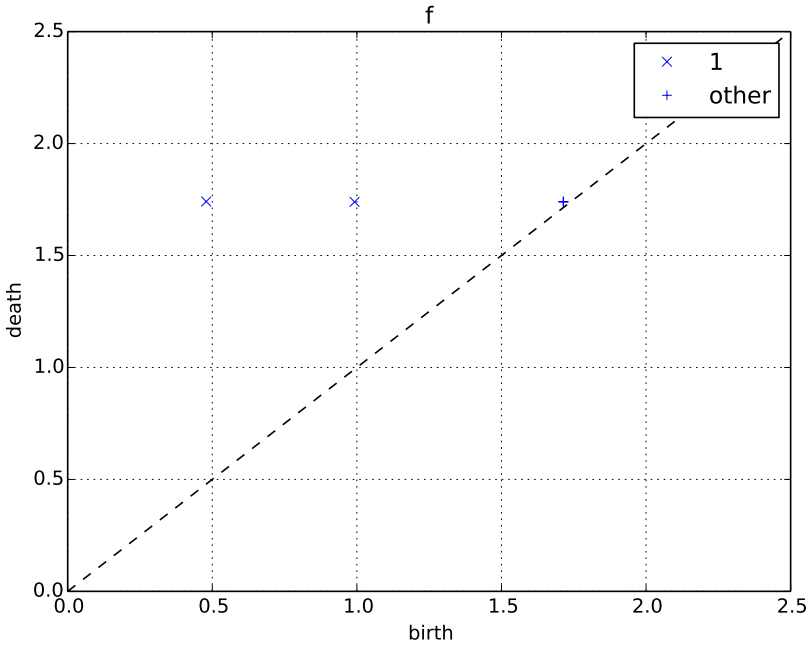


Figure 6.8: Green part of the circle is sent to the red one. Direction of arrows represents how the map rotates circles.

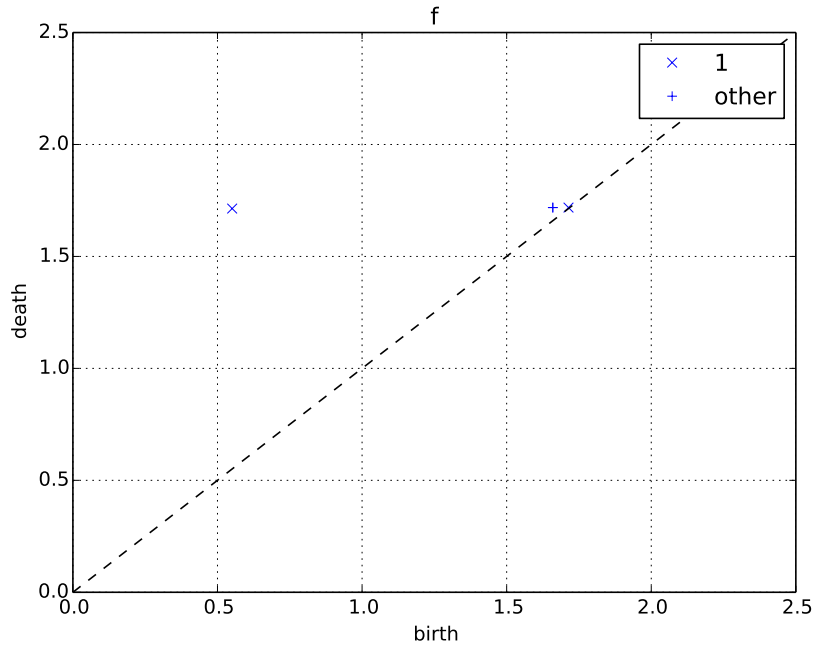


(a) λ-persistence diagrams

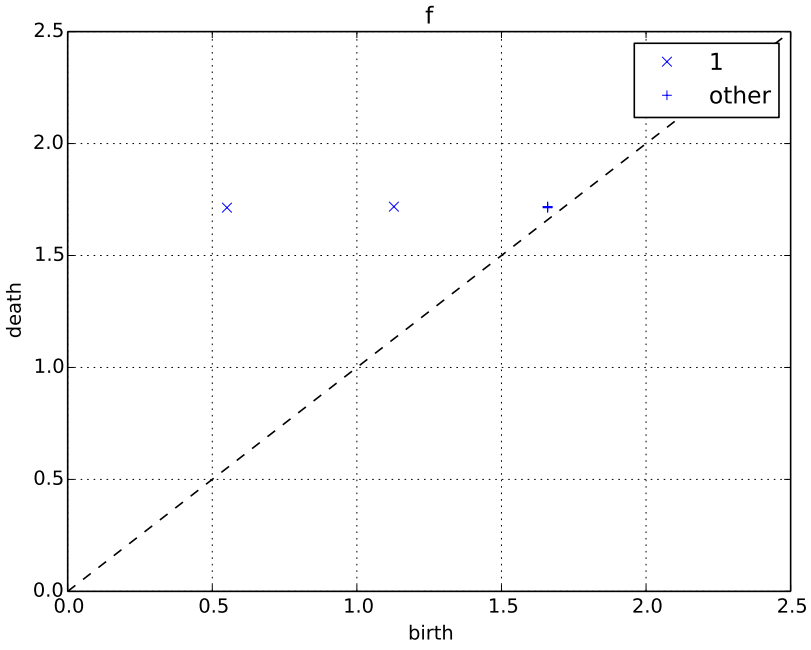


(b) Generalized λ-persistence diagrams

Figure 6.9: The 1-dimensional λ-persistence diagrams of the map $f : \mathbb{E} \to \mathbb{E}$ without noise. Dots represent persistence intervals for eigenvalue 1, crosses persistence intervals for other eigenvalues.

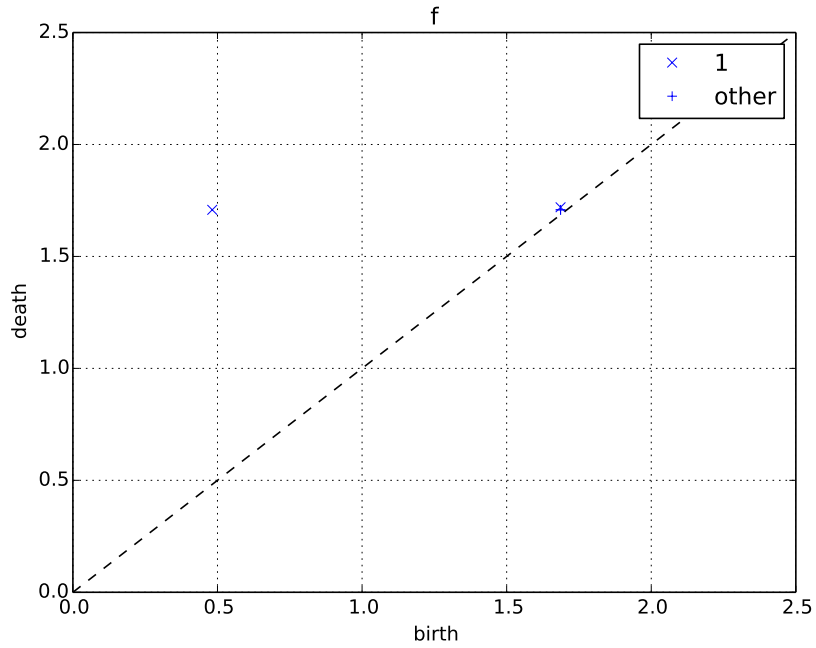


(a) λ-persistence diagrams

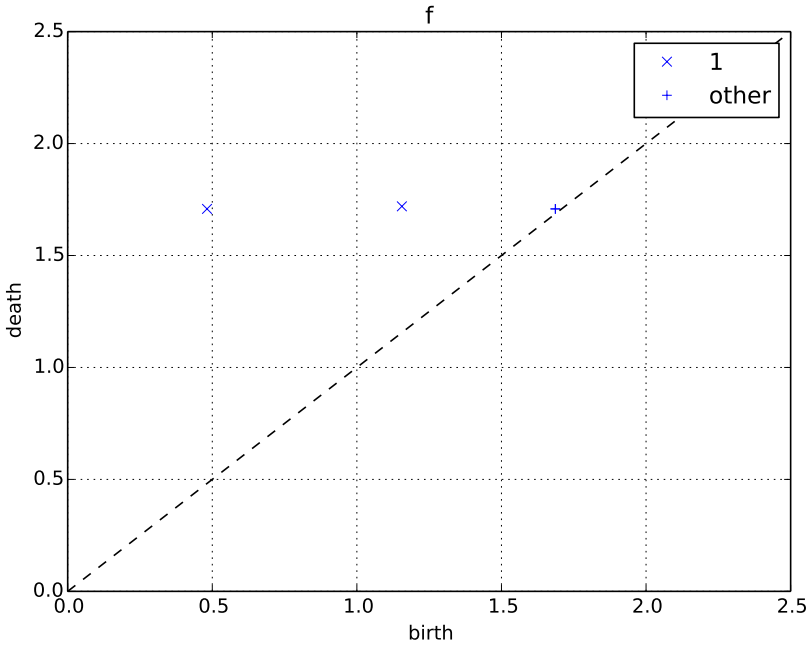


(b) Generalized λ-persistence diagrams

Figure 6.10: The 1-dimensional λ-persistence diagrams of the map $f : \mathbb{E} \to \mathbb{E}$. Gaussian noise with variance $\sigma = 0.03$

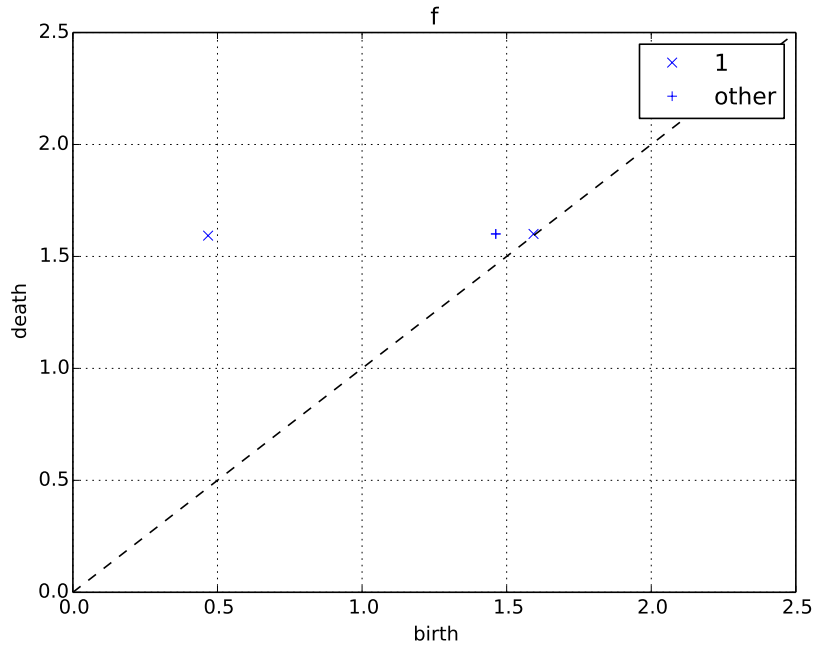


(a) λ-persistence diagrams

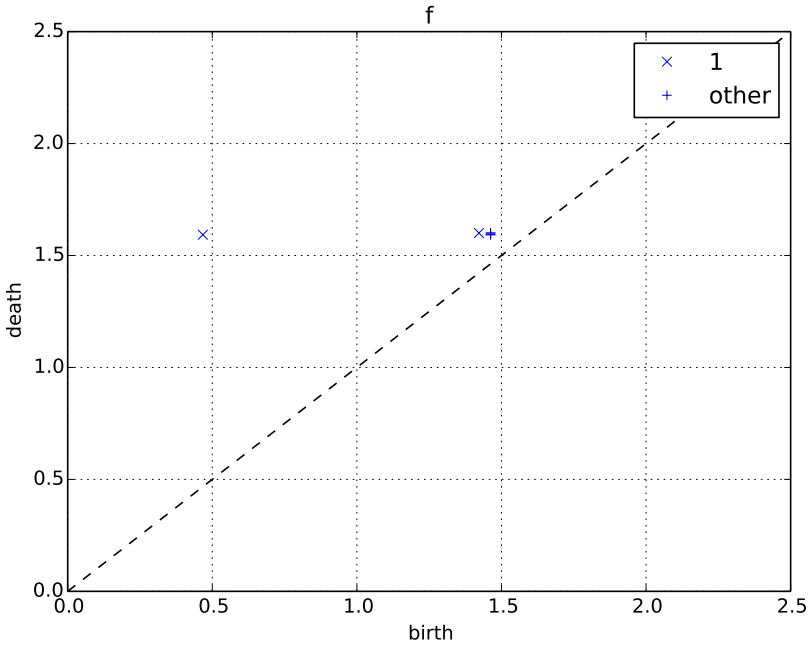


(b) Generalized λ-persistence diagrams

Figure 6.11: The 1-dimensional λ-persistence diagrams of the map $f : \mathbb{E} \to \mathbb{E}$. Gaussian noise with variance $\sigma = 0.06$



(a) λ-persistence diagrams



(b) Generalized λ-persistence diagrams

Figure 6.12: The 1-dimensional λ-persistence diagrams of the map $f : \mathbb{E} \to \mathbb{E}$. Gaussian noise with variance $\sigma = 0.09$

Chapter 7

Usage

All algorithms are implemented in C++ language. We use functions and data structures from STL [30], Boost [28] and CAPD [29] library.

7.1 Installation

The software was compiled and tested on Ubuntu 14.04 operating system. After unpacking the package execute in the main folder the command:

```
$ cmake .  && make
```

After successful configuration and compilation there should be a binary file named `vrmp` in the same folder.

7.2 Usage

To compute λ-persistence diagrams execute:

```
$ ./vrmp -p file_with_points -l file_with_candidates
```

where file_with_points contains a set of points $S \subset \mathbb{R}^n$ together with a map $g : S \to S$ in the following format:

- the first line contains one integer number n
- the following lines contain $2n$ decimal numbers. In every line we define a map g on a point $p_1 \in \mathbb{R}^n$ by setting $g(p_1) = p_2$. The first n numbers are coordinates of a point p_1, the following n numbers are coordinates of a point p_2.

The file `file_with_candidates` should contain candidates for eigenvalues in separate lines.

To compute λ-generalized persistence diagrams execute:

```
$ ./vrmp -p file_with_points -l file_with_candidates -g
```

7.3 Results

Results are stored in the file named `diagram.txt`. Persistence intervals are listed first by dimension (0th, 1st dimension) and inside dimension by the ordering of the candidates in the file `file_with_candidates`. Every persistence interval is described in one line by two real numbers birth and death. Dimensions are separated by a line containing `#####`. The persistence diagrams for eigenvalue 1 and 2 containing intervals $[0; 2.10257]$, $[0; 0.151285]$ for eigenvalue 1 in dimension 1, and interval $[0.0710019; 0.080787]$ for eigenvalue 2 in dimension 1 is shown below:

```
1
1
0 2.10257
0 0.151285
2
0.0710019 0.080787
#####
```

The first line denotes the dimension of homology.

7.4 Plots

The python script `diagram.py` in the folder `scripts` is provided to generate plots of diagrams from text files. To generate persistence diagram run:

```
$ ./diagram.py diagram.txt
```

The script `diagram.py` uses Matplotlib[19] library.

7.5 Examples

The input data used to generate plots in Fig. 6.1 is in the folder `examples/z2`. Every file contains the σ parameter of the Gaussian noise in the name. Similarly the input data used to compute diagram in Fig. 6.2 is in the folder `examples/ref`. The data used to generate Figs. 6.4 to 6.6 is stored respectively in the files `rand1.in`, `rand2.in` and `rand3.in` in the folder `examples/torus`. Input data for the figure of eight is stored in the folder `examples/eight`.

List of Algorithms

List of Figures

List of Tables

Index

Bibliography

[1] Madjid Allili, Konstantin Mischaikow, and Allen R Tannenbaum. Cubical homology and the topological classification of 2d and 3d imagery. In *Proceedings. 2001 International Conference on Image Processing*, volume 2, pages 173–176, 2001.

[2] Zhaojun Bai, James Demmel, Jack Dongarra, Axel Ruhe, and Henk van der Vorst. *Templates for the solution of algebraic eigenvalue problems: a practical guide*, volume 11 of *Software, Environments and Tools*. Siam, 2000.

[3] Ulrich Bauer, Michael Kerber, Jan Reininghaus, and Hubert Wagner. Phat–persistent homology algorithms toolbox. In *Mathematical Software–ICMS 2014*, pages 137–143. Springer, 2014.

[4] Peter Bubenik and Jonathan A Scott. Categorification of persistent homology. *Discrete & Computational Geometry*, 51(3):600–627, 2014.

[5] Gunnar Carlsson and Vin De Silva. Zigzag persistence. *Foundations of computational mathematics*, 10(4):367–405, 2010.

[6] Frédéric Chazal, David Cohen-Steiner, Marc Glisse, Leonidas J Guibas, and Steve Y Oudot. Proximity of persistence modules and their diagrams. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, pages 237–246. ACM, 2009.

[7] Chao Chen and Michael Kerber. Persistent homology computation with a twist. In *Proceedings 27th European Workshop on Computational Geometry*, volume 11. Citeseer, 2011.

[8] Scott Cohen and Carlo Tomasi. *Systems of bilinear equations*. Stanford University, Department of Computer Science, 1997.

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

[10] Vin De Silva and Robert Ghrist. Homological sensor networks. *Notices of the American mathematical society*, 54(1), 2007.

[11] Vin De Silva, Robert Ghrist, and Abubakr Muhammad. Blind swarms for coverage in 2-d. In *Robotics: Science and Systems*, pages 335–342, 2005.

[12] Harem Derksen and Jerzy Weyman. Quiver representations. *Notices of the AMS*, 52(2):200–206, 2005.

[13] Herbert Edelsbrunner and John Harer. *Computational topology: an introduction*. American Mathematical Soc., 2010.

[14] Herbert Edelsbrunner, Grzegorz Jabłonski, and Marian Mrozek. The persistent homology of a self-map. *Foundations of Computational Mathematics, online first*, 2014.

[15] Herbert Edelsbrunner, David Kirkpatrick, and Raimund Seidel. On the shape of a set of points in the plane. *Information Theory, IEEE Transactions on*, 29(4):551–559, 1983.

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

[17] Peter Gabriel. Unzerlegbare darstellungen i. *Manuscripta Mathematica*, 6(1):71–103, 1972.

[18] Allen Hatcher. *Algebraic topology*. Tsinghua University Press, 2002.

[19] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95, 2007.

[20] Tomasz Kaczynski, Konstantin Mischaikow, and Marian Mrozek. *Computational homology*, volume 157 of *Applied Mathematical Sciences*. Springer, 2004.

[21] Aleksei Ivanovich Kostrikin, Yu I Manin, and Michael E Alferieff. *Linear algebra and geometry*. Gordon and Breach Science Publishers, 1997.

[22] Dimitry Kozlov. *Combinatorial algebraic topology*, volume 21 of *Algorithms and Computation in Mathematics*. Springer Science & Business Media, 2008.

[23] Konstantin Mischaikow and Marian Mrozek. Chaos in the lorenz equations: a computer-assisted proof. *Bulletin of the American Mathematical Society*, 32(1):66–72, 1995.

[24] Konstantin Mischaikow, Marian Mrozek, and Pawel Pilarczyk. Graph approach to the computation of the homology of continuous maps. *Foundations of Computational Mathematics*, 5(2):199–229, 2005.

[25] Dmitriy Morozov. Persistence algorithm takes cubic time in worst case. *BioGeometry News*, 2005.

[26] Edwin H Spanier. *Algebraic topology*. Springer Science & Business Media, 1994.

[27] Afra Zomorodian. Fast construction of the vietoris-rips complex. *Computers & Graphics*, 34(3):263–271, 2010.

[28] Boost library. `http://www.boost.org/`.

[29] Computer assisted proofs in dynamics. `http://capd.ii.uj.edu.pl/`.

[30] Standard template library. `http://www.sgi.com/tech/stl/`.