FP7-ICT-2011.8
Contract no.: 318493

www.toposys.org

Deliverable D1.3

Progress and Activity Report on WP1

Deliverable Nature:	Report (R)
Dissemination Level: (Confidentiality)	Public (PU)
Contractual Delivery Date:	M36
Actual Delivery Date:	M36
Version:	1.0

Contents

1 Summary

In this deliverable, we outline work which was done for Work Package 1 in the final year of the project. We made progress in understanding the persistence of maps. There has been progress made in this year in the computation of both the eigenvalues as well as the generalized eigenvectors of a sampled system.

There has also been algorithmic progress in the form of an incremental algorithm for the persistence of a pair of maps. In addition to a proof-of-concept code, we are currently preparing a paper on this new algorithm.

We have also begun work on a better representation for chain maps building on the work reported last year on the equivalence of Cech, Delaunay and Wrap complexes through simple homotopies. This defines a way to "flow" chain structures through the filtration. This will provide tighter approximations of maps than are currently possible.

The work on cohomology of recurrent systems is reported on in the Delvierable for applications (4.3), since it the techniques have been applied as a feature for the epidemiology of flu seasons. Finally, the majority of work on a Unified Theory as well as tree-like persistence is reported on in the Deliverable for Categorical Systems (Deliverable 3.3), since it is an direct extension of the topoidal foundations for persistence. Here, we do, however, outline the current state of progress torwards a stability theorem for generalized persistence - which is also currently being prepared for submission.

This report is based on the following papers and reports:

- G. Jablonski, "Persistent homology of a self-map," PhD Thesis
- M. Ethier, G. Jablonski, M. Mrozek, "Computing the persistence of a self-map with the Kronecker canonical form," submitted
- M. Juda, P. Skraba, "An Incremental Algorithm for Computing the Persistence of a Pair of Maps," in prepration
- U. Bauer, H. Edlesbrunner, G. Jablonski, M. Mrozek, "Chain Maps on Delauanay Complexes from Discrete Morse Theory and Persistence Flow," in preparation
- J. Pita Costa, M. Vejdemo-Johansson, P. Skraba, "The Persistence Topos," in preparation.

We do not explictly report on it here, hwoever there has also been work on discretization strategies for Poincare maps which can be found in the paper "Discretization strategies for computing conley indices and Morse decompositions of flows," by K. Mischaikow, M. Mrozek, F. Weilandt, (submitted). Also we have investigated the computation of the fundamental group in "Fundamental Group Algorithm for Low-Dimensional Tesselated CW Complexes," by P. Brendel, G. Ellis, M. Juda and M. Mrozek (submitted). Finally, we also mention the following paper, which looks at the lattice structure which exists in directed spaces - "On the hierarchy of d-structures," by J. Pita Costa and L. Fajstrup (submitted).

2 Generalized Eigenfunctors

Here we present the algorithm for computing the generalized eignfunctor of a self-map. Recall that for a fixed value $\lambda \in \mathbb{F}$ and an endomorphism $\gamma : V \to V$ we define the set:

$$E_\lambda(\gamma) := \{v \in V \mid \gamma(v) = \lambda v\} \tag{1}$$

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.

For a pair of maps, we extend the definition of an eigenspace to a pair of linear maps:

Definition 2.1. *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.

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:

$$\psi(u) = \lambda\varphi(u) \tag{2}$$

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 (2) still makes sense.

Given $\lambda \in \mathbb{F}$ and an endomorphism $\gamma : U \to U$ one can generalize equation (1) to:

$$E_{\lambda,k}(\gamma) := \{v \in V \mid (\gamma - \lambda\mathrm{id})^k u = 0\}, \tag{3}$$

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 (1). For $k \geqslant 2$ 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 2.2. *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:*

$$\begin{aligned} v_{k-1} &= (\gamma - \lambda\mathrm{id})(v_k), \\ v_{k-2} &= (\gamma - \lambda\mathrm{id})(v_{k-1}), \\ &\ldots, \\ v_2 &= (\gamma - \lambda\mathrm{id})(v_3), \\ v_1 &= (\gamma - \lambda\mathrm{id})(v_2). \end{aligned} \tag{4}$$

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

Theorem 2.1. *[1][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.*

Here we present the algorithm to compute the generalized eigenvectors of a self-map.

2.1 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 $\varnothing = K_0 \subset K_1 \subset K_2 \subset \cdots \subset K_n = K$. 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 1 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:     return (K̄, w_K̄)
```

Proposition 2.1. *1 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 1 is linear with respect to the number of simplices in $\mathcal{K}$. 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 $\mathcal{K}$:

$$D[k, l] := \langle \sigma_k, \partial(\sigma_l) \rangle \tag{5}$$

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 2.1. *Let $\mathcal{K}$ be a filtration presented in 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 1.

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 2).

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.

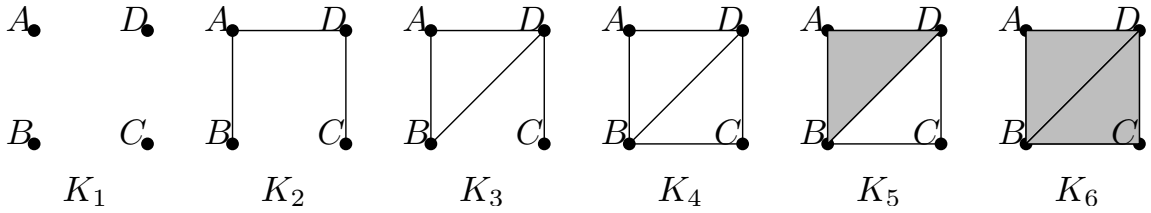


Figure 1: Filtration $\mathcal{K}$ consisting of seven levels $K_1, \ldots, K_7$.

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

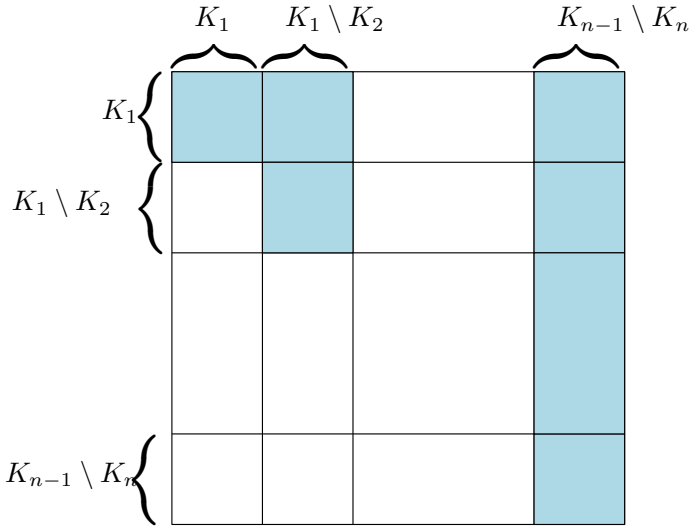


Figure 2: The structure of the matrix D: simplices are arranged into blocks, s.t. i-th block corresponds to new simplices in K_i.

The classical persistent homology algorithm [12][p. 153] consist in bringing the boundary matrix to the reduced form R. Its generalization to the case of matrices over a general field $\mathbb{F}$ is presented as 2. Note that by performing the same sequence of column additions as in 2 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 2 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 j_0 < j with low(j_0) = low(j) do
6:             α := R[low(j), j]/R[low(j), j_0]
7:             R[., j] := R[., j] − α · R[., j_0]
8:     extract persistence diagram dgm from R
9:     return dgm, R
```

Theorem 2.2. *[16] The worst-case running time of 2 is $\Omega(n^3)$ where n is the number of simplices in the filtration.*

Note that the practically observed running time of 2 is significantly better than the worst case [13, 16]. 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 2.2. *[12][Ch. VII.1] Addition of a positive simplex σ_j into a filtration gives birth to a new homology class.*

Proposition 2.3. *[12][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 2.4. *[12][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 2.2 to 2.4 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.

Our implementation of the peristent homology algorithm uses a modified version known as the persistence algorithm with a twist [15]. This version in practice is much faster in most cases [13].

	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 2: The reduced form of the boundary matrix of 2.1. 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 $\mathrm{Dgm}(\mathcal{K})$. The simplex ABD kills the cycle created when BD was added.

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 2.5. *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 2.1.

Proposition 2.6. *2.1 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 2.7. *4 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 3 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:    return v
```

Algorithm 4 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:    return G_1, ..., G_n
```

2.2 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. 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 $\bar{\mathcal{K}}$.

Algorithm 5 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 contain k then
11:                        w[i] := 0
12:                let j be the index of the cycle c in basis of U_k
13:                Φ_k[., j] := w
14:    return Φ_1, ..., Φ_n
```

Proposition 2.8. *5 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 $\bar{\mathcal{K}}$ 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 2.5.*

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 2.1. *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 6. 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 6 Construction of a tower in **Pairs(Vect)**

```
1: function TOWER(diagram dgm_K, dgm_K̄, matrix R_K, R_K̄, map h, g)
2:     matrix Φ_1, ..., Φ_n := SEQUENCE(dgm_K, dgm_K̄, R_K, R_K̄, h)
3:     matrix Ψ_1, ..., Ψ_n := SEQUENCE(dgm_K, dgm_K̄, R_K, R_K̄, g)
4:     matrix G_1 ..., G_n := INCLUSIONSEQUENCE(dgm_K̄)
5:     return Ψ_1, ..., Ψ_n, Φ_1, ..., Φ_n, G_1 ..., G_n
```

2.3 Eigenfunctors

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

$$\begin{array}{ccccccc}
\dots & \xrightarrow{w_{i-1}} & V_i & \xrightarrow{w_i} & V_{i+1} & \xrightarrow{w_{i+1}} & \dots \\
 & & \varphi_i \uparrow & & \varphi_{i+1} \uparrow & & \\
\dots & \xrightarrow{v_{i-1}} & U_i & \xrightarrow{v_i} & U_{i+1} & \xrightarrow{v_{i+1}} & \dots \\
 & & \psi_i \downarrow & & \psi_{i+1} \downarrow & & \\
\dots & \xrightarrow{w_{i-1}} & V_i & \xrightarrow{w_i} & V_{i+1} & \xrightarrow{w_{i+1}} & \dots
\end{array} \tag{6}$$

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

Algorithm 7 Construction of an eigenspace tower in **Vect** from a tower in **Pairs(Vect)**

1: **function** EIGENFUNCTOR(`matrix` $\Psi_1, \dots, \Psi_n, \Phi_1, \dots, \Phi_n, G_1 \dots, G_{n-1}$; `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: **return** $E_1, \dots, E_n, H_1, \dots, H_{n-1}$

Proposition 2.9. *7 applied to sequences $\Psi_1, \dots, \Psi_n$ of matrix representations of $\psi_1, \dots \psi_n$, $\Phi_1, \dots, \Phi_n$ of $\varphi_1, \dots, \varphi_n$ and $G_1 \dots, G_{n-1}$ of maps $v_1, \dots, v_{n-1}$ returns the tower of eigenspaces*

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

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

2.3.1 Generalized eigenfunctor

The algorithm presented in the last section may be extended to compute the tower of generalized eigenspaces. We formulate the algorithm (8 to compute the second level generalized eigenfunctor $E_{\lambda,2}$. The main difference with respect to 7 is in solving the equation:

$$\psi_i(u_2) - \lambda\varphi_i(u_2) - \varphi_i(u_1) = 0 \tag{7}$$

for u_2, where u_1 is one of the solutions of

$$\psi_i(u_1) - \lambda\varphi_i(u_1) = 0. \tag{8}$$

The solution of (7) 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 $K = \ker \Psi_i - \lambda \cdot \Phi_i$.

Algorithm 8 Computation of the 2nd level generalized eigenfuctor a tower in **Pairs(Vect)**

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

2.4 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:

Definition 2.3. *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 2.4. *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 2.2. *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 2.10. *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 2.3. *Any tower of vector spaces admits a matching basis.*

Let $\mathcal{X} = (X_i, \xi_i)$ be a tower in **Vect**. Without loss of generality we assume that for $i \leqslant 0$ 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 2.3 it is sufficient to show that a suitable change of bases B_i brings matrices A_i to matching form. This is achieved by 9 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 9 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: **for** $i := 1$ to $(n-1)$ **do**
5: 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.
6: **return** $B_1, \ldots, B_n; A_1, \ldots, A_{n-1}$

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.

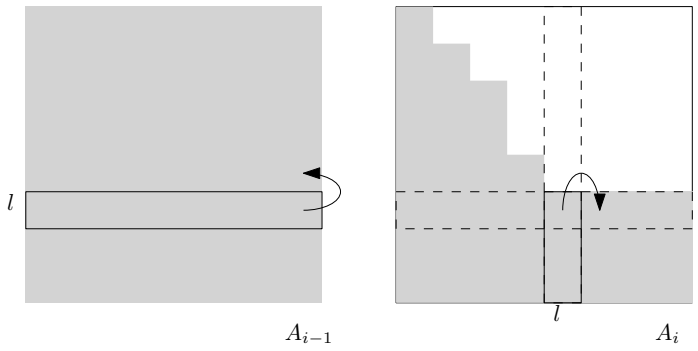


Figure 3: Whenever we do a left-to-right column operations on a matrix A_i, we do the same operation on columns of B_i. However in A_{i-1} we apply a corresponding top-to-bottom row operation.

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 2.10 we have that column echelon form of A_{i+1} is not violated.

The overview of Phase 1 and Phase 2 is presented in 5.

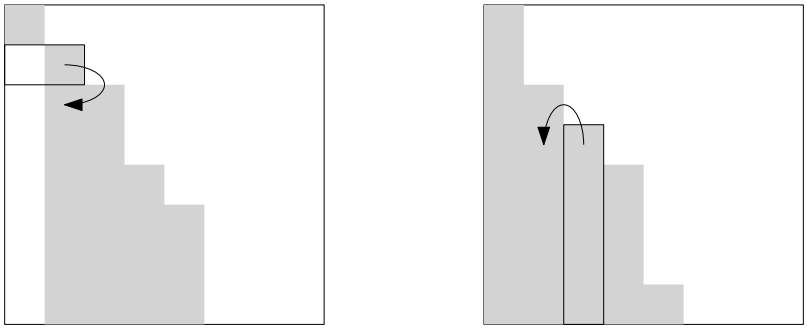


Figure 4: In Phase 2 when we perform a bottom-to-down row operation in A_i for example we add the second row to the third one, we have to modify the matrix A_{i+1} by subtracting the third column from the second column. In consequence, the column echelon form is not violated for A_{i+1}.

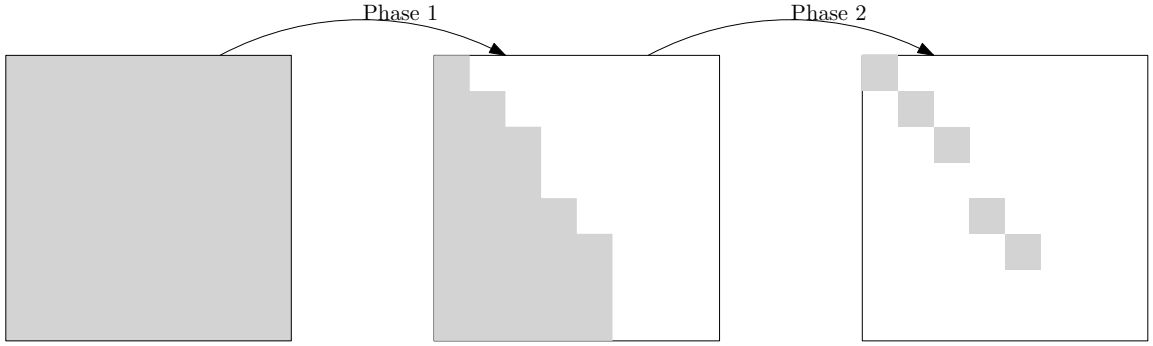


Figure 5: Two phases of the matching algorithm shown on a matrix A_i. First we bring the matrix A_i into the column echelon form, and then to the matching form.

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 2.1. *10 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 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 6-12 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. □

Algorithm 10 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:                let i be such that A_l[i, j] = 1
11:                j := i
12:                l := l + 1
13:            add [k, l] to dgm
14:    return dgm
```

Example 2.3. *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 2.11. *The running time of 9 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. □

2.5 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 11.

Theorem 2.4. *Algorithm (11) 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.*

Computation of generalized λ-persistence diagram is similar to 11, but with the call to the function GENERALIZEDEIGENFUNCTOR in line 8.

Algorithm 11 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

3 Numerical Examples

Here we present four numerical experiments which were computed using the algorithms desceibed in the previous section. The first two examples are self-maps of the unit circle $\S^1$ 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 12.

Algorithm 12 Approximation algorithm for a function f on points S

1: **function** PREPAREDATA(f, S)
2: Let $g : S \to S$ be a map
3: **for** each $p \in S$ **do**
4: $q := f(p)$
5: let $\bar{q}$ be a randomly chosen point in S
6: **for** each $r \in S$ **do** ▷ Find nearest point to q in S
7: **if** `dist`$(r, q) <$ `dist`$(\bar{q}, q)$ **then**
8: $\bar{q} := r$
9: $g(p) := \bar{q}$
10: **return** g

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.

3.1 Expansion map

As our first example we study the map $f : \S^1 \ni z \mapsto z^2 \in \S^1$. We represent points in $\S^1$ 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 order to be able to inspect several cases. We then use 12 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 11.

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 3. 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 3: Running time in seconds of 11 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 6. 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[14][8.3]. It is caused by the high noise.

3.2 Reflection map

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

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

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.

3.3 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 8. Then, every point on $\mathbb{T}$ can be described as a pair of

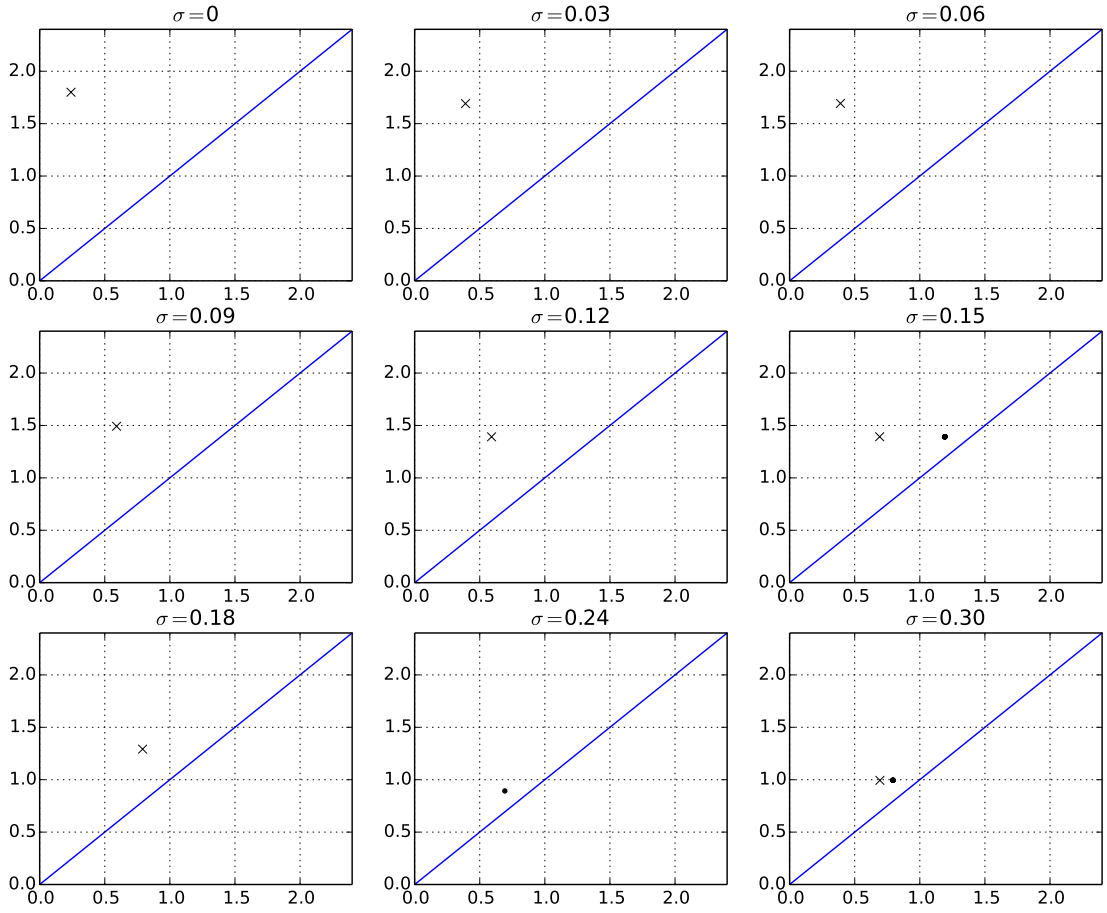


Figure 6: The persistence diagrams of $f(z) : \S^1 \ni z \mapsto z^2 \in \S^1$ in the first homology level, with varying noise σ. Crosses show the persistence intervals for eigenvalue $\lambda = 2$ and dots represent intervals visible for all values of $\mathbb{Z}_{1009}$. The x-axis is the birth of a homology generator and y-axis is the death.

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 $y_1 - y_2 \in \mathbb{Z}$.

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

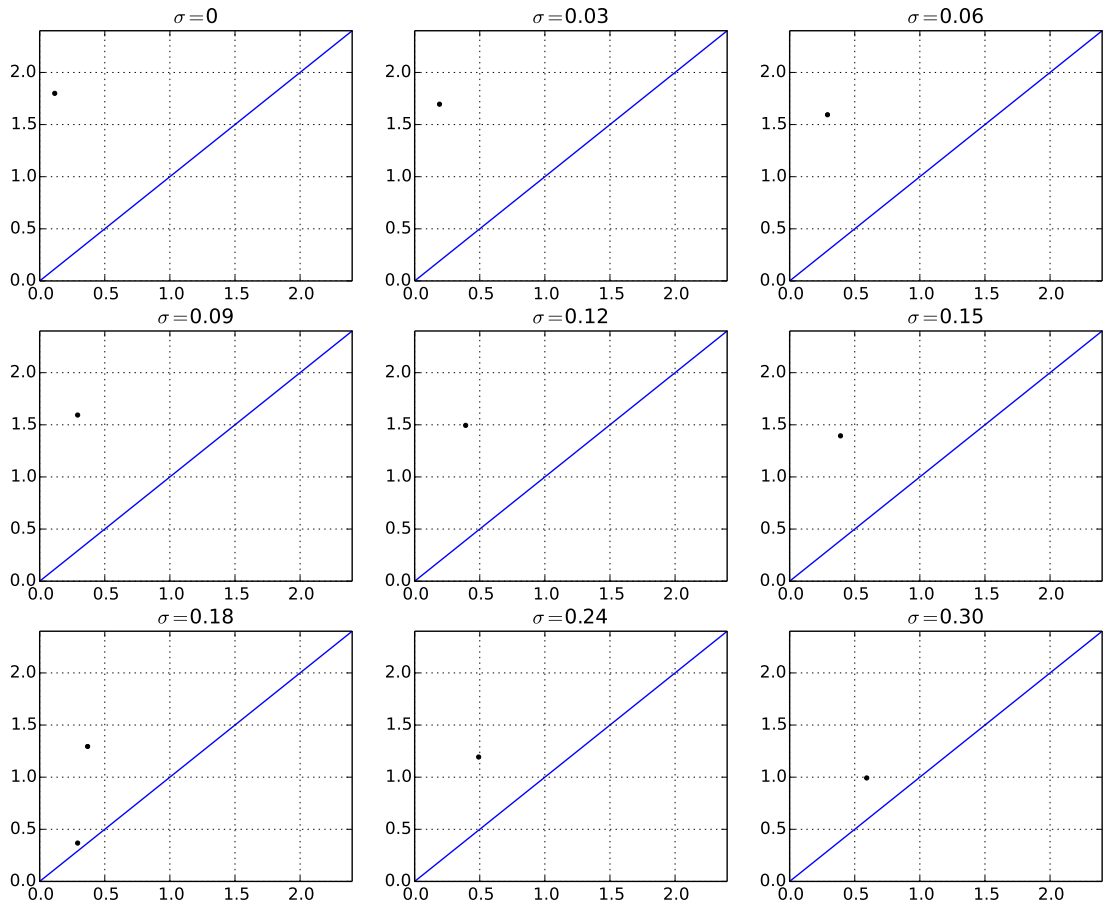


Figure 7: λ-persistence diagrams of the map $f(z) := \bar{z}$ for different noise levels. Dots represent intervals of the $\lambda = -1$ eigenavlue.

the eigenspace towers for maps induced by the following three matrices:

$$A_1 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, A_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, A_3 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}. \tag{9}$$

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

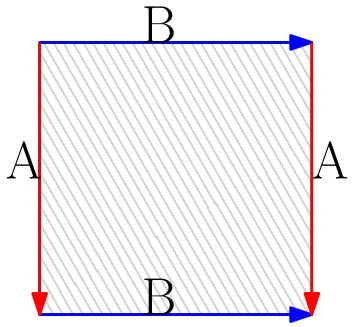


Figure 8: Torus representation as a square with opposite edges A and B identified. Arrows show the direction of glueing.

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 11 applied to S and g_i is presented in 9, 10 and 11. 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.

3.4 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 12. Let the map $f : \mathbb{E} \to \mathbb{E}$ be defined as follows:

$$f(x) := \begin{cases} S_a(2\alpha) \text{ if } x = S_a(\alpha) \text{ and } \alpha \in [0, \pi) \\ S_b(3\pi - 2\alpha) \text{ if } x = S_a(\alpha) \text{ and } \alpha \in [\pi, 2\pi) \\ x \text{ if } x \in S_b \end{cases} \tag{10}$$

Intuitively the map f:

1. 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(13)
2. stretches the lower left half to the right circle (13)
3. 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 14(b), 15(b), 16(b) and 17(b) we show generalized λ-persistence diagrams. Figures 14(a), 15(a), 16(a) and 17(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 $\lambda \neq 1$.

4 Single Pass Algorithm

We have developed a single pass algorithm for computing the persistence of a self map. Our initial implementationis still being tested and a more comprehensive writeup is currently being prepared to be submitted.

To illustrate the idea, we consider two chain maps given by ϕ and ψ and let $\phi, \psi : X \to Y$. Recall that the we would like to compute the persistence of

$$E = \ker(\phi - \psi)/(\ker\phi \cap \ker\psi)$$

If we consider these as graded modules (or more generally towers of spaces), we can compute the Smith Normal Form and obtain the barcode. The idea behind the incremental algorithm is use the fact that at any one parameter value the above are vector spaces. Therefore, by keeping track of the ranks of images and kernels (and their respective spans), we can formulate an algorithm based on positive and negative simplices - much like the original persistence algorithm.

We first make the assumption that for simplices with the same values, we insert the simplices in the target space Y before the simplices in the source space X. We keep a basis for the image of the boundary matrices of X and Y, which we denote ∂_X and ∂_Y respectively. Likewise, we keep a basis for $\mathrm{im}(\phi-\psi)$, $\mathrm{im}(\phi)$ and $\mathrm{im}(\psi)$. Finaally we also keep a basis for the generators and relations of E, which we denote G and R respectively.

Below we have a simplified version of the algorithm – with many details omitted as the full pseudocode is a full two pages of boundary cases.

We first consider what happens when we insert a simplex σ into the target space Y. The above bases are assumed to be up to date up to point before the insertion of the simplex.

1. We reduce the boundary of σ by ∂_Y - if it is a cycle (i.e. in the span of ∂_Y, we are finished)
2. If the chain is not in the span of ∂_Y, the reduced chain is added to the span.
3. The reduced chain is also used to reduce $\mathrm{im}(\phi-\psi)$, $\mathrm{im}(\phi)$ and $\mathrm{im}(\psi)$. If a column in $\mathrm{im}(\phi-\psi)$ becomes zero, this consistitutes a birth (this follows since the rank of the image has decreased). The corresponding element of $\mathrm{coim}(\phi-\psi)$ is added to G. If a column in $\mathrm{im}(\phi)$ or/and $\mathrm{im}(\psi)$, this potentially constitutes a death, and the corresponding elements from $\mathrm{coim}(\phi)$ or $\mathrm{coim}(\psi)$, are reduced by G. If they are in the span, then this consistutes a death, since the quotient has increased in rank.

When the boundary of a simplex σ is added to X,

1. Reduce the chain with respect to ∂_X. If it is not a cycle (i.e. a boundary), then add it to ∂_X.
2. If it is a cycle, then compute the $\mathrm{im}(\phi - \psi)(c)$, where c represents a cycle.
3. Reduce the image with respect to $span(\partial_Y \oplus \mathrm{im}(\phi - \psi))$. If it is non-zero, then add it to $\mathrm{im}(\phi - \psi)$.
4. If it is in the span, then check that it is not in the span of $\mathrm{im}(\phi)$ and $\mathrm{im}(\psi)$. Note that only one needs to be checked. If it is not in the span (can be reduced by either basis, this is homology class which is born and immediately dies). Otherwise, c is added to G.

The key idea is to perform reductions only when necessary. When adding a simplex to Y we can get a death if the rank of $(\ker\phi \cap \ker\psi)$ increases (or if both images decrease in rank). There can likewise be a birth if $(\ker(\phi - \psi)$ increases in rank (or the image decreases in rank). When a simplex is added to X, only a birth can occur. Tghis follows from the condition that these are required to be chain maps and that we have added simplices to Y first. The remaining work is book-keeping to ensure we have the correct spans after the imsertion of a simplex.

This algorithm has been implemented in Python and SAGE and will be next rewritten to C++ once it is further tested and optimized. This suggests that most variants of persistence should have an incremental version. Therefore, there is a clear pipeline from algebraic formulation to incresasingly optimized algorithms.

5 Chain Maps on Delauany Complexes and Persistence Flow

The work reported on in Deliverable 1.2 described an equvialence between Cech, Delauanay and Wrap complexes through simple homotopies. This extends this work to the representation of chain maps. In the persistence of a self map, we are required to consider a subset of the complex at a parameter value, so that the map is well defined (i.e. its image is in the complex). The main idea of the work here is to try to "flow" the image chain backwards in "time," so that we can consider a larger subset of the source complex.

Suppose $\mathbb{M}$ is a compact subset of $\mathbb{R}^n$ and $f\colon \mathbb{M} \to \mathbb{M}$ is a continuous self-map. We are interested in studying the dynamical system in the setting in which f is known only through a *sample*, by which we mean a finite subset $X \subseteq \mathbb{M}$, a function $g\colon X \to X$, and an $\varepsilon \geqslant 0$ such that $d(g(x), f(x)) \leqslant \varepsilon$ for all $x \in X$. We call ε the *approximation constant* of the sample. To construct such a sample, we may choose a finite set X such that the ε-neighborhoods of the points cover $\mathbb{M}$. For every $x \in X$, we then define $g(x) = y$ such that the ε-neighborhood of y contains $f(x)$. The concept of a *sampled dynamical system* appears already in [5], and our work relates to that paper by providing the combinatorial and computational technology to run the algorithm in [5] on Delaunay complexes instead of the much larger Čech or Vietoris–Rips complexes. The concept itself is less demanding than the traditional *discrete dynamical system*, which samples time but not space. We believe that this difference is essential to reach experimental settings in which pairs $(x, f(x))$ can be observed while the self-map remains otherwise unknown. To see that a vanishing approximation constant is not always possible, consider the map $f\colon [0,1] \to [0,1]$ defined by $f(u) = \frac{u}{2}$. Letting x be the smallest positive value in a finite set $X \subseteq [0,1]$, its image, $f(x)$, does not belong to X. We call

$$\lambda = \max_{x \neq y \in X} \frac{d(g(x), g(y))}{d(x,y)} \tag{11}$$

the *Lipschitz constant* of g. It is not necessarily the same or even close to the Lipschitz constant of f. However, Kirszbraun proved that for every $g\colon X \to X$ there is a continuous $h\colon \mathbb{M} \to \mathbb{M}$ such that $h(x) = g(x)$, for all $x \in X$, and g and h have the same Lipschitz constant; see [9, 10]. The Stability Theorem in [5] implies a connection between the dynamics of f and h, namely that for every eigenvalue t, the bottleneck distance between the towers of eigenspaces induced by f and by h is at most ε. Furthermore, the Inference Theorem in the same paper implies that for small enough ε and every eigenvalue, g gives the correct dimension of the corresponding eigenspace of the endomorphism between the homology groups of $\mathbb{M}$ induced by f.

5.1 Prior work and results

We continue the program started in [5] whose goal is the embedding of persistent homology in various steps of the computational approach to dynamical systems. In comparison, the contribution of this paper is modest, but it facilitates faster algorithms and therefore computational solutions to problems much larger than handles in [5]. Specifically, we construct a partial chain map on the Delaunay complexes in three steps:

$$\begin{array}{ccc} \mathrm{Del}_r(X) & \xleftarrow{3} & \mathrm{Del}_s(X) \\ & \searrow^{1} \quad \uparrow^{2} & \\ & \mathrm{Cech}_s(X), & \end{array}$$

in which $s = \lambda r$ and we assume the more difficult case of $\lambda > 1$. Step 1 is easy because of the Kirszbraun Intersection Property for balls established by Gromov [8]. In our context, it asserts that if a collection of balls with radius r in $\mathbb{R}^n$ has a non-empty common intersection and we move the balls to new locations such that the distance between any two centers is at most λ times the original distance, then expanding the balls to radius $s = \lambda r$ guarantees again a non-empty common intersection. This implies that $\mathrm{Del}_r(X)$ is a subcomplex of $\mathrm{Cech}_s(X)$. Step 2 is the most difficult of the three and relies on the recent proof that $\mathrm{Cech}_s(X)$ collapses to $\mathrm{Del}_s(X)$ [2]. To achieve our declared goal of fast software, we describe an implementation of this collapse that avoids the explicit construction of the Čech complex. Transporting chains within the filtration of Delaunay complexes, Step 3 is comparably easy. It has an elementary description in terms of the persistence barcode, but we prefer the equivalent construction using what we refer to as the *persistence gradient flow* of the Delaunay triangulation, which we feel is of independent interest.

6 Stability of Generalized Persistence

In this section, we outline the general proof for stability of generalized persistence. In Deliverable 3.3 (and the corresponding papers), we discuss the interleaving stability which follows from a chain level interleaving. The next natural question is an interleaving on the modules correspond to an equivalent bound on bottleneck distance (i.e. a matching or some equivalent notion for indecomposables of the module).

While we have worked most of the pieces for this work, we are still working on a more formal writeup. The key idea is to follow the original stability of persistence proof (as well as the algebraic stability proof). We recall the key four ingredients:

1. Quandrant Lemma - This is the initial step, which says that if we have an ϵ-interleaving between two modules F and G on the chain level, then there is an injective map from $F^2\epsilon \to G$. That is, if we consider a class in F which is at least 2ϵ-persistent, then there is a corresponding class in G. In the case of standard persistence, this means that each bar in F or length at least 2ϵ has a corresponding bar in G (e.g. it cannot map to the diagonal). This follows in our generalized setting by functoriality.

2. Box Lemma - This is the extension of the quadrant lemma to boxes. In the case of persistence, this is done by considering inclusion and exclusion. In our setting however, this is a consequence of the quadrant lemma and the Heyting algebra structure. In a sense, this is due to the fact that the thickening functor gives us an analogue of boxes. We note however that this must still be made more precise.

3. Easy Bijection Lemma - The first is the observation if the maps in the Box Lemma are of rank one, then the Box Lemma is sufficient for bottleneck distance (since if there is only one choice, then the Box Lemma implies a matching). This is equivalent in the case generalized modules - if the maps are one-to-one across indecompasables then a matching follows.

4. Interpolation - If we can interpolate between modules, then under appropriate tameness assumptions, repeated application of the Easy Bijection Lemma achieves bottleneck distance. Essentially, we discretize finely enough that at any one step, the Lemma applies and then repeat to achieve a matching between then original modules.

Here we make several remarks. First, we note that the Quadrant Lemma is a consequence of functoriality. Likewise, interpolation is a general operation which can be adopted from the algebraic stability proof in [17], slightly reformulated in terms of pushouts (i.e. colimits) and/or pullbacks(i.e. limits). Using these universal constructions, given F and G which are ϵ-interleaved, it is relatively straightforward to find an interpolated module H which is ϵ_1-interleaved with F and ϵ_2-interleaved with G for any $\epsilon_1 + \epsilon_2 \geqslant \epsilon$.

On a side note, if we take the image from limit to colimit, we also obtain an interpolated module with all the "small bars" removed. Considering this interpolation may result in a space of persistence diagrams/barcodes with nicer geometric properties (again, with some genericity assumptions). However, we have not explored sufficiently yet.

The remaining steps are in the process of being checked. The key remaining obstacle is to define an analogue of bottleneck distance more precisely. In the case of one-dimensional persistence, the indecomposables are of rank one making the definition obvious. In our generalized setting, this is not the case, as indecomposables may be much more complicated. It is clear that a notion such

as the support will be stable, but we are currently investigating more refined notions of matchings between indecomposables which are stable.

Finally, while our notion of interleaving does agree in the one dimensional case, it is in some sense more restrictive than considering notions which have been considered for other case such as zig-zag persistence. That is, there is a fundamental difference between thickening the topological space and the thickening of the interval (e.g. a point maps to an interval, an interval maps to a bigger interval, etc.).

We close by noting that we believe this will be a weaker stability theorem than what is known in the one-dimensional case. It is however much more general and shoudl provide an insight into directions for proving stronger results about persistence in more specific cases.

References

[1] A.I. Kostrikin, Y.I Manin, M.E. Alferieff. Linear algebra and geometry. *Gordon and Breach Science Publishers, 1997.*

[2] U. Bauer and H. Edelsbrunner. The Morse theory of Čech and Delaunay complexes. Manuscript, IST Austria, Klosterneuburg, Austria, 2015.

[3] K. Borsuk. On the imbedding of systems of compacta in simplicial complexes. *Fund. Math.* **35** (1948), 217–234.

[4] H. Edelsbrunner and J. L. Harer. *Computational Topology. An Introduction.* Amer. Math. Soc., Providence, Rhode Island, 2010.

[5] H. Edelsbrunner, G. Jabłoński and M. Mrozek. The persistent homology of a self-map. *Found. Comput. Math.*, to appear.

[6] H. Edelsbrunner, D. Letscher and A. Zomorodian. Topological persistence and simplification. *Discrete Comput. Geom.* **28** (2002), 511–533.

[7] R. Forman. Morse theory for cell complexes. *Adv. Math.* **134** (1998), 90–145.

[8] M. Gromov. Monotonicity of the volume of intersections of balls. In *Geometrical Aspects of Functional Analysis*, Springer Lecture Notes **1267** (1987), 1–4.

[9] M. D. Kirszbraun. Über die zusammenziehenden und Lipschitzsche Transformationen. *Fund. Math.* **22** (1934), 77–108.

[10] J. H. Wells and L. R. Williams. *Embeddings and Extensions in Analysis.* Ergebnisse der Mathematik und ihrer Grenzgebiete, Band 84, Springer-Verlag, 1975.

[11] E. Welzl. Smallest enclosing disks (balls and ellipsoids). In *New Results and New Trends in Computer Science*, H. A. Maurer (ed.), Springer, LNCS **555** (1991), 359–370.

[12] H. Edelsbrunner and J. Harer Computational Topology: An Introduction *AMS Press, 2010*,

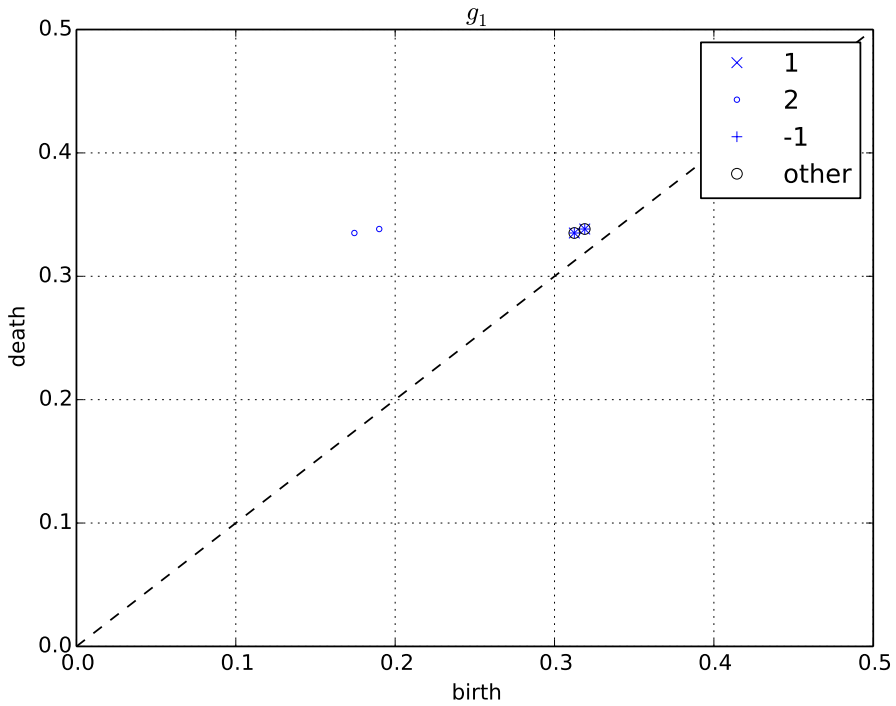
[13] U.Bauer, M.Kerber, J.Reininghaus, H.Wagner PHAT Persistent Homology Algorithms Toolbox. *Mathematical Software ICMS 2014, Lecture Notes in Computer Science Volume 8592, 2014, pp 137-143*

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

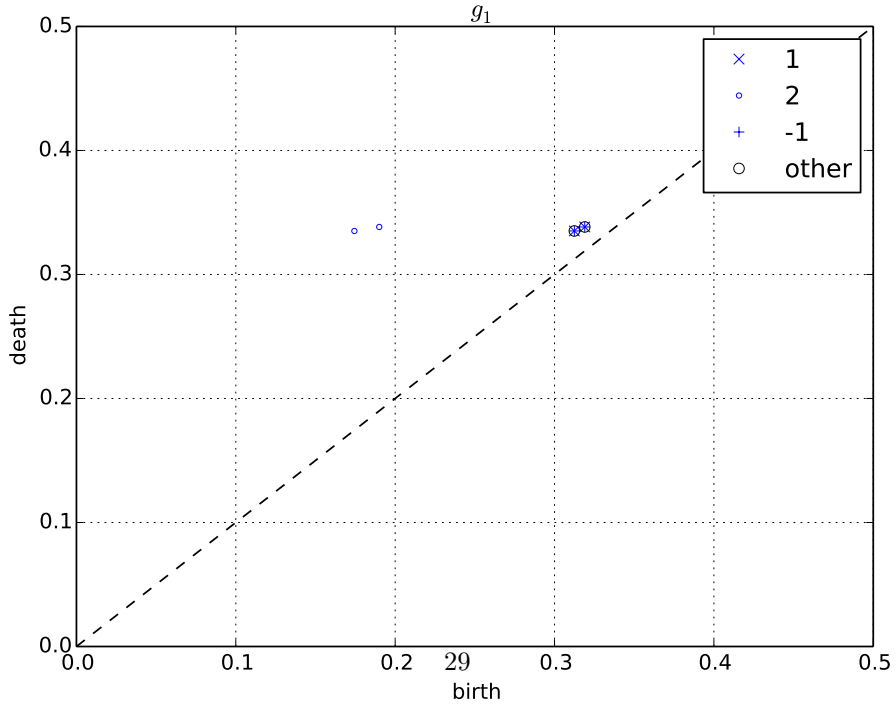
[15] C.Chen, M.Kerber Persistent Homology Computation With a Twist *27th European Workshop on Computational Geometry, 2011.*

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

[17] F. Chazal, V. de Silva, M. Glisse, S. Oudot. The Structure and Stability of Persistence Modules. *arXiv:1207.3674v1*

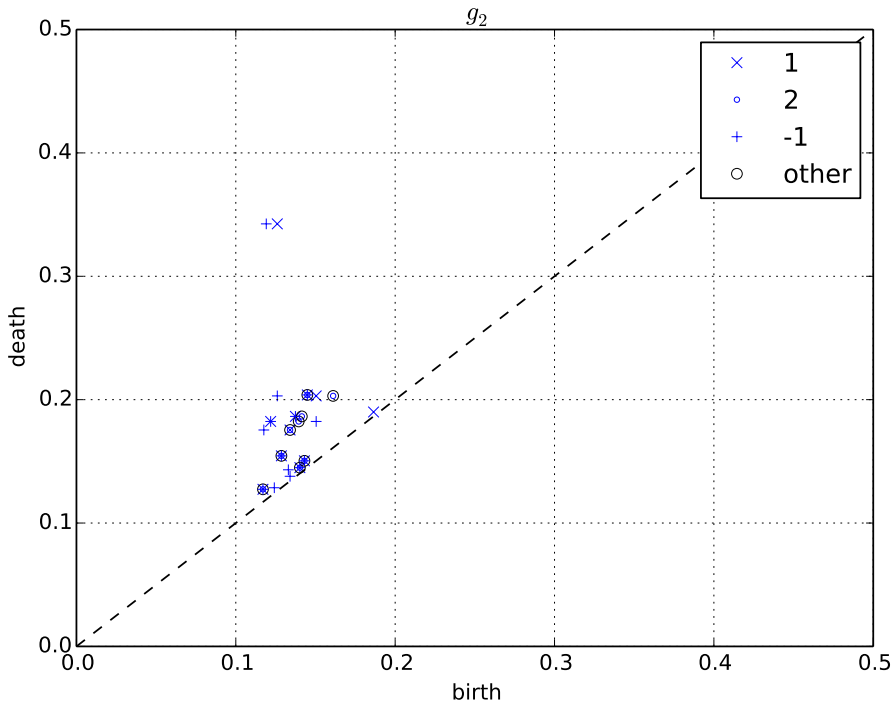


(a) λ-persistence diagrams

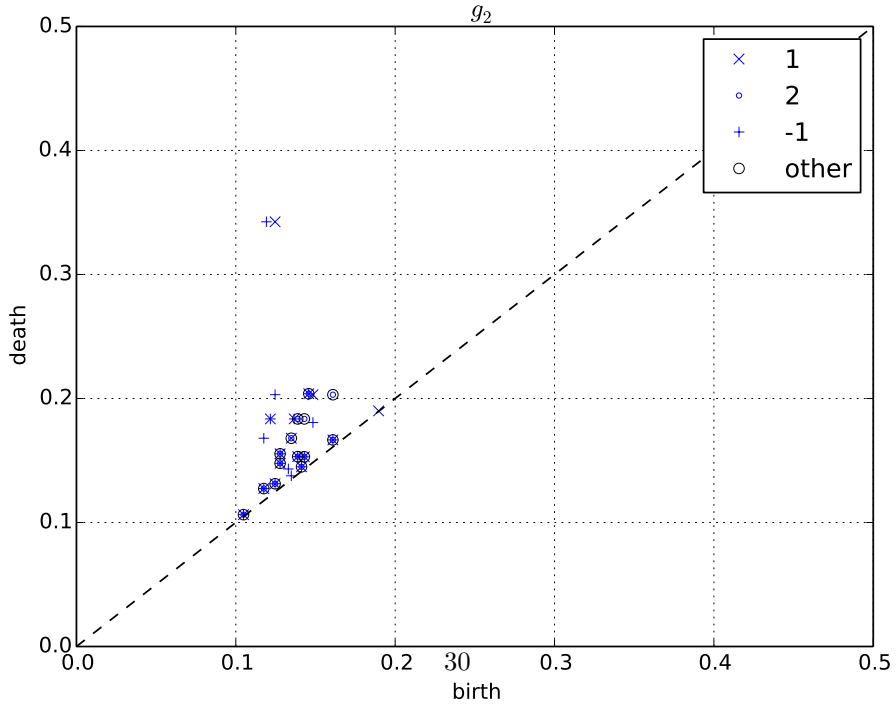


(b) Generalized λ-persistence diagrams

Figure 9: The 1-dimensional persistence diagrams of towers of eigenspaces of g_1.

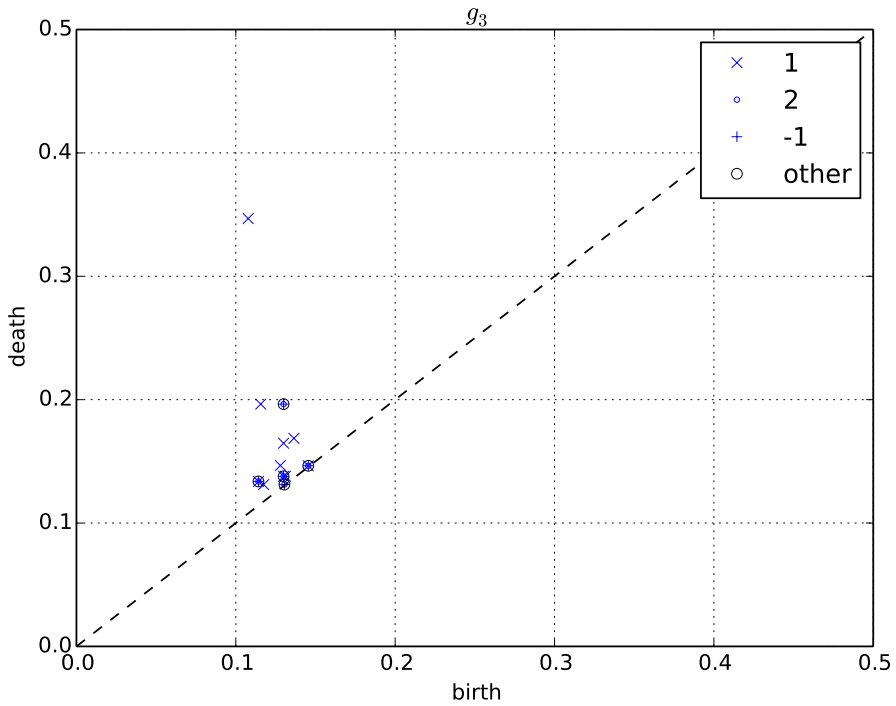


(a) λ-persistence diagrams

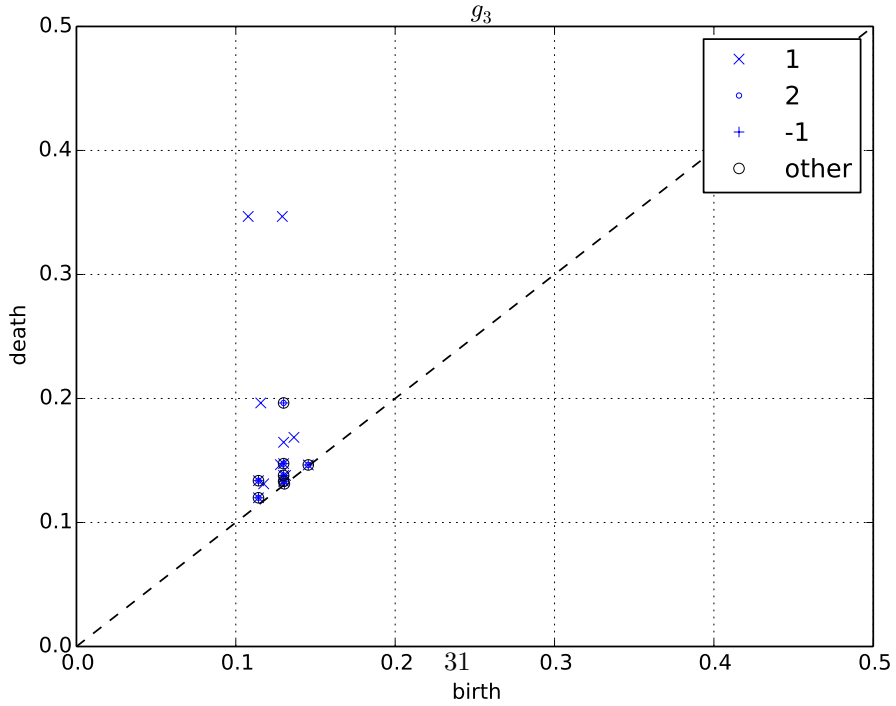


(b) Generalized λ-persistence diagrams

Figure 10: The 1-dimensional persistence diagrams of towers of eigenspaces of g_2.



(a) λ-persistence diagrams



(b) Generalized λ-persistence diagrams

Figure 11: The 1-dimensional persistence diagrams of towers of eigenspaces of g_3.

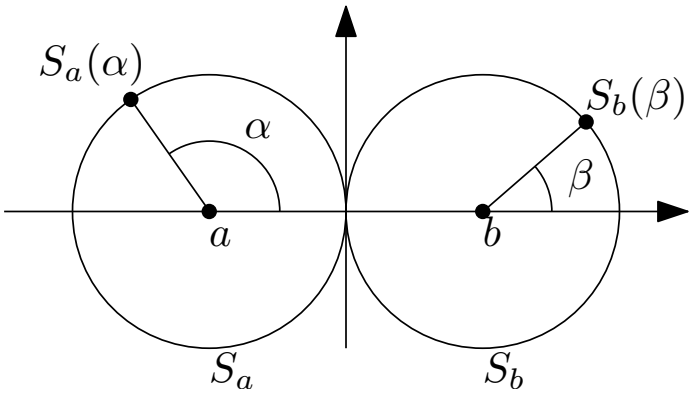


Figure 12: A plot of the space $\mathbb{E}$. The points $S_a(\alpha)$ and $S_b(\beta)$ are presented together with the defining angles α and β.

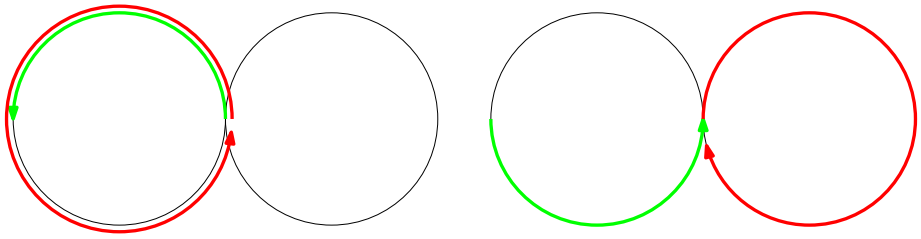
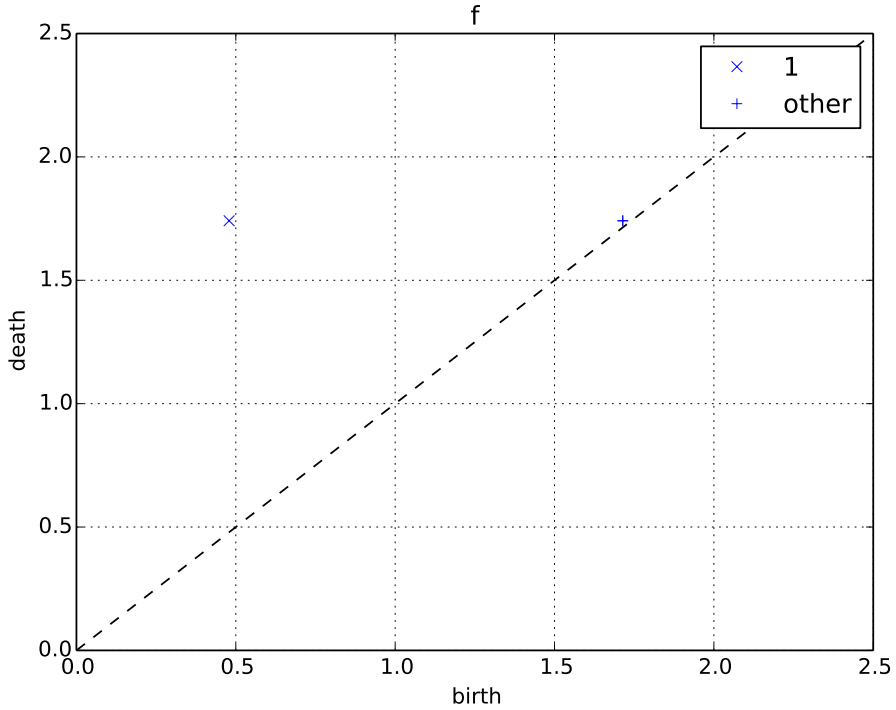
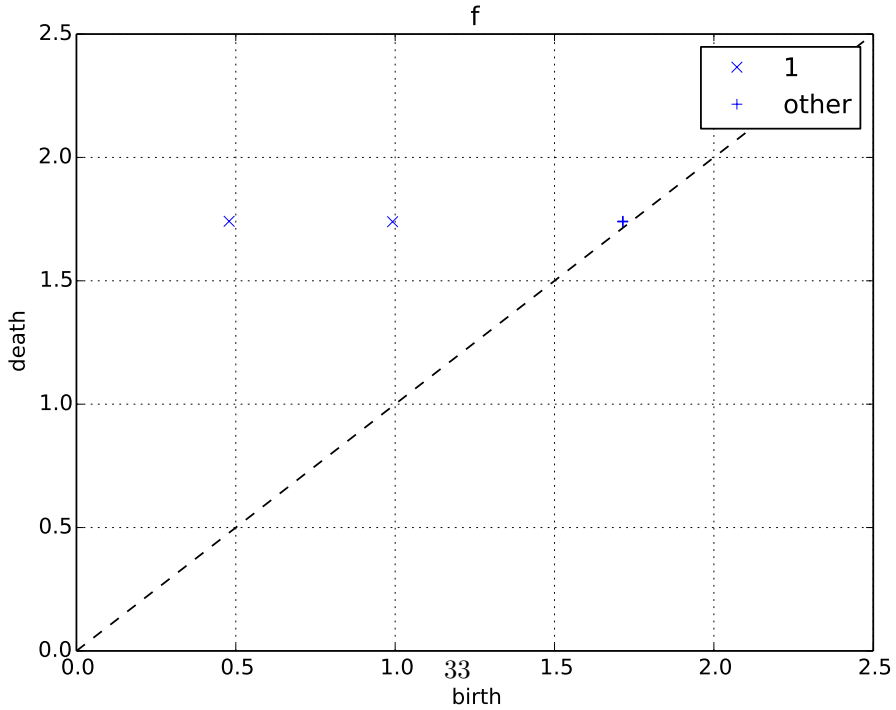


Figure 13: Green part of the circle is sent to the red one. Direction of arrows represents how the map rotates circles.

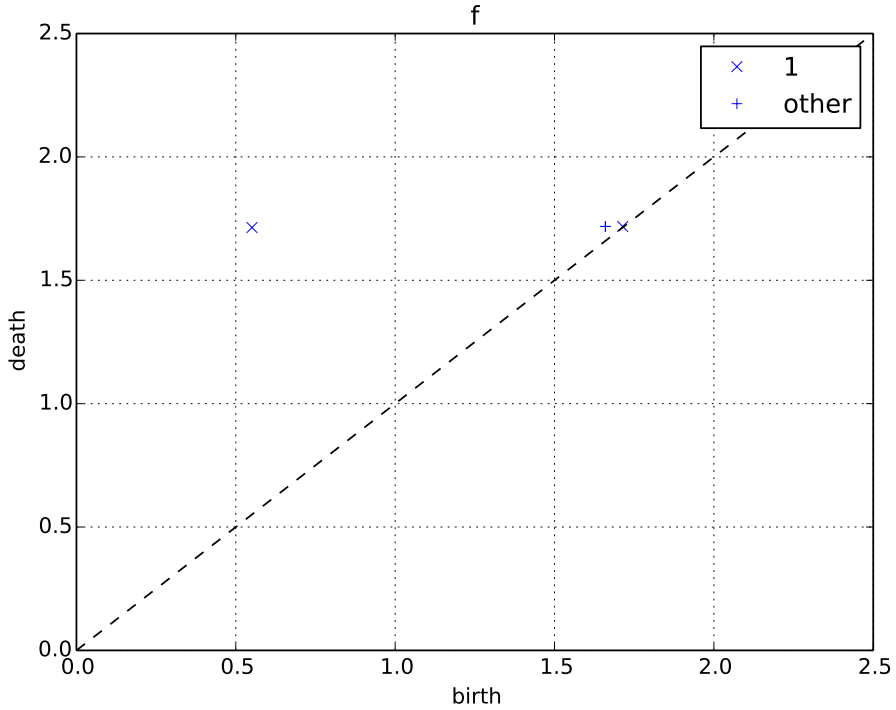


(a) λ-persistence diagrams

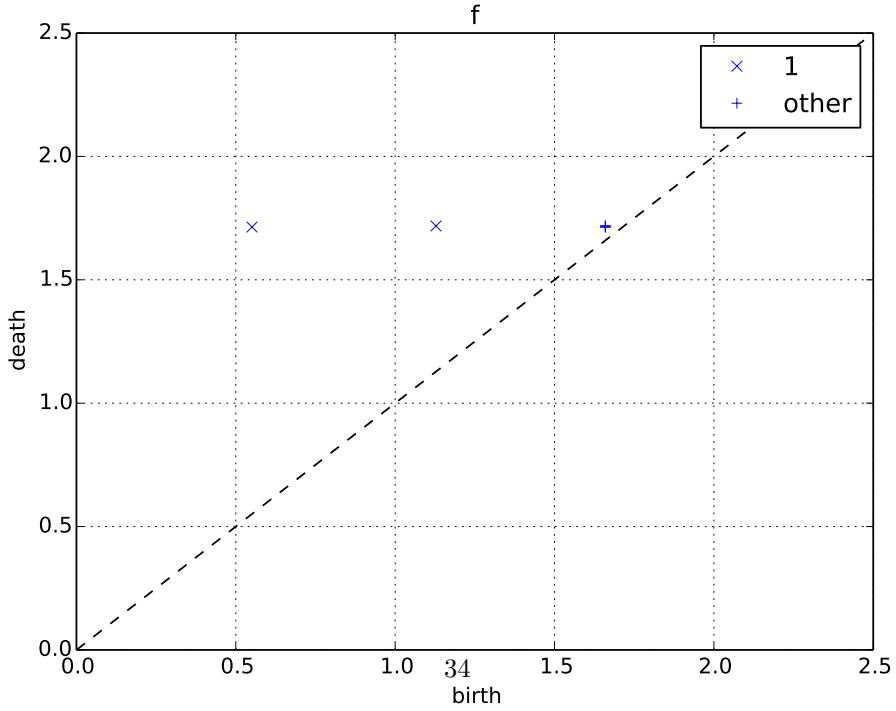


(b) Generalized λ-persistence diagrams

Figure 14: 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.

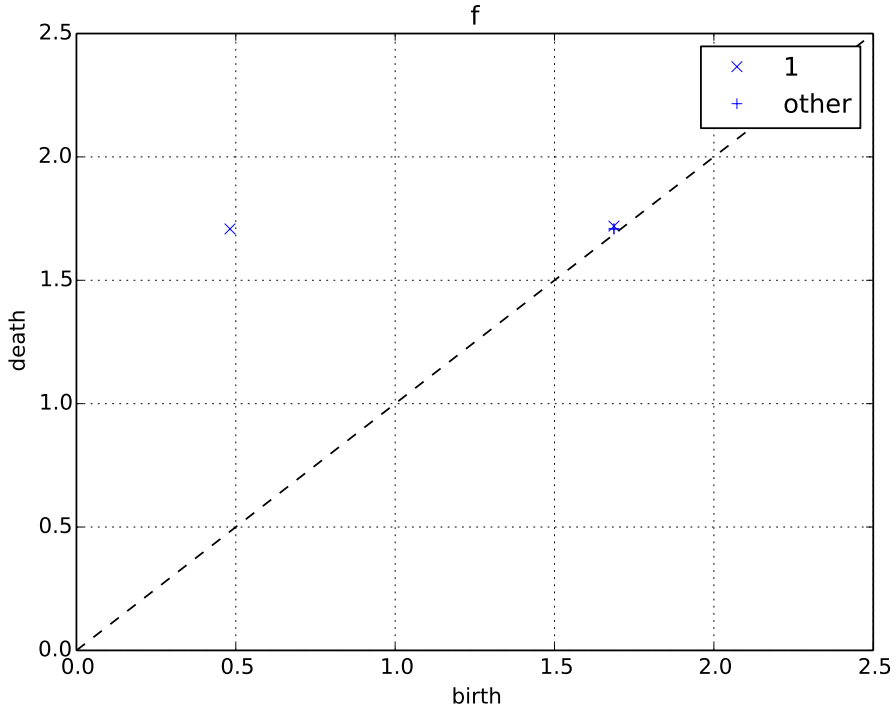


(a) λ-persistence diagrams

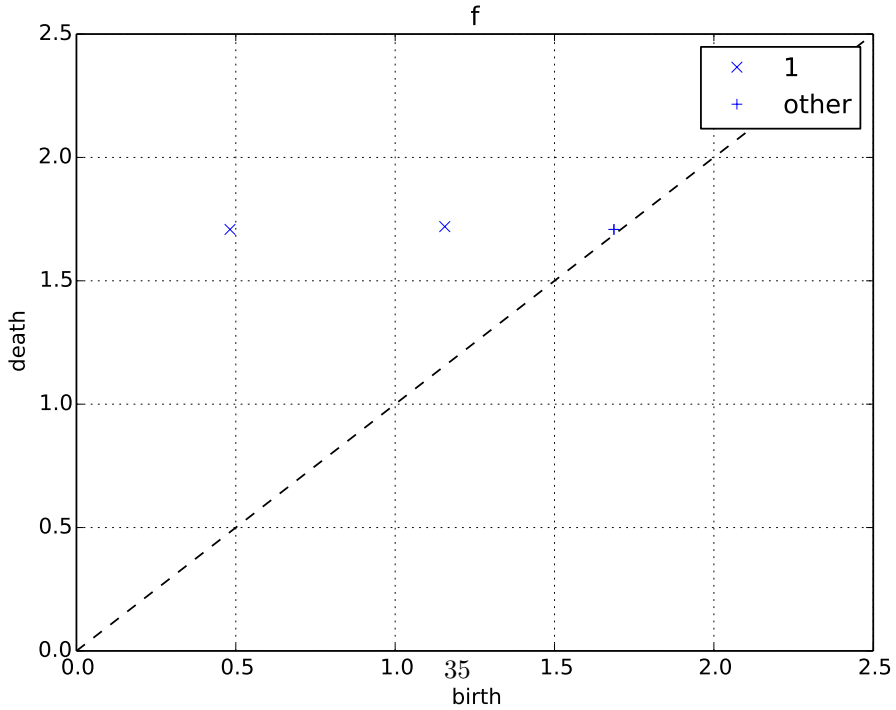


(b) Generalized λ-persistence diagrams

Figure 15: The 1-dimensional λ-persistence diagrams of the map $f : \mathbb{E} \to \mathbb{E}$. Gaussian noise with variance $\sigma = 0.03$

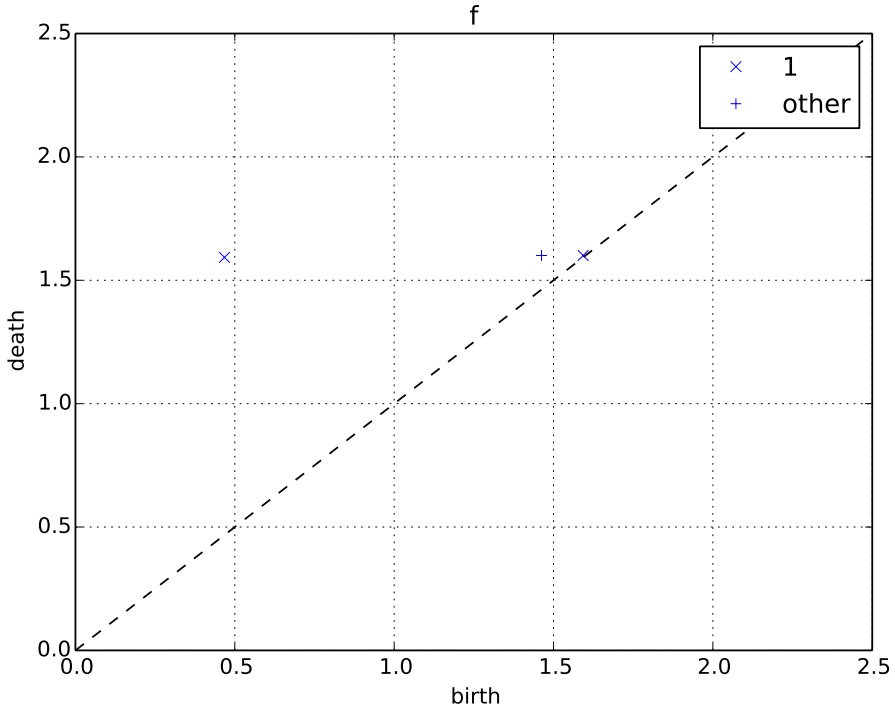


(a) λ-persistence diagrams

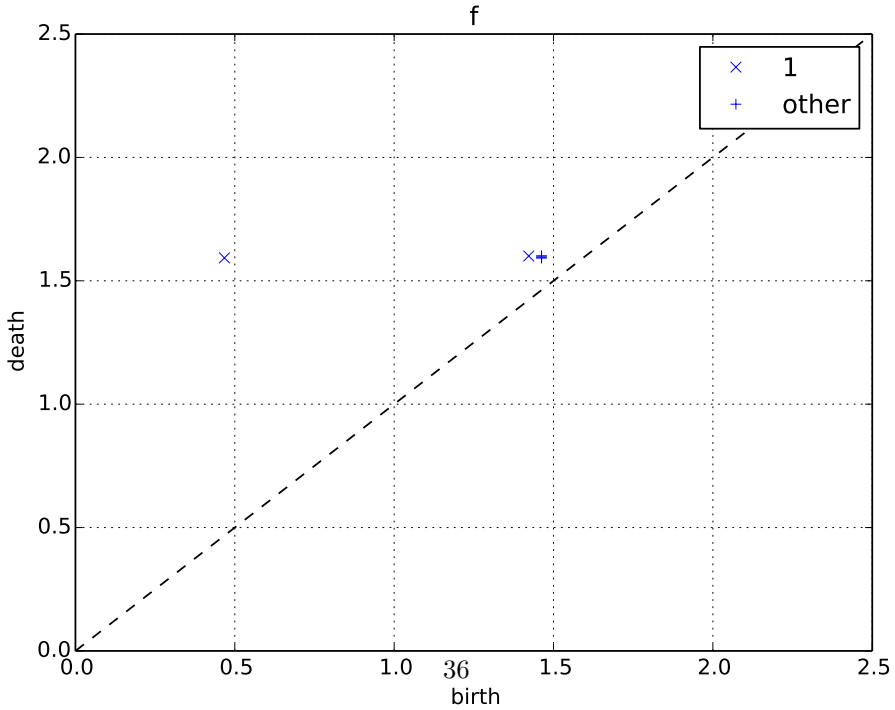


(b) Generalized λ-persistence diagrams

Figure 16: 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 17: The 1-dimensional λ-persistence diagrams of the map $f : \mathbb{E} \to \mathbb{E}$. Gaussian noise with variance $\sigma = 0.09$